Research Square

Preprints are preliminary reports that have not undergone peer review.
They should not be considered conclusive, used to inform clinical practice,
or referenced by the media as validated information.

Reconfiguring Authenticity in the Age of Artificial Intelligence: A Critical Code Analysis of Deepfake Generation and Detection Systems

Ruixin Hu

912383134@qq.com

Xi'an Jiaotong University

Article

Keywords: Deepfake, Authenticity, Science and Technology Studies (STS), Critical Code Analysis, Technological Systems

Posted Date: November 17th, 2026

DOI: https://doi.org/10.21203/rs.3.rs-10519755/v1

License: This work is licensed under a Creative Commons Attribution 4.0 International License.
Read Full License

Additional Declarations: No competing interests reported.

Reconfiguring Authenticity in the Age of Artificial Intelligence: A Critical Code Analysis of Deepfake Generation and Detection Systems

Abstract

This study examines how authenticity is constructed and reconfigured within deepfake generation and detection systems. Drawing on the perspective of Science and Technology Studies (STS) and the methodological framework of Critical Technical Practice (CTP), it conducts a critical analysis of the core code of two representative open-source projects, FaceFusion and DeepfakeBench. Through the syntactic, semantic, and pragmatic dimensions of code, the study investigates how authenticity judgments are technically operationalized within deepfake systems. The findings reveal three interrelated processes through which authenticity is reconfigured in deepfake technological systems. First, authenticity undergoes Proxyization, as direct judgment of authenticity itself is increasingly suspended and verisimilitude comes to function as its technological proxy. Second, authenticity undergoes Riskification, as authenticity judgment shifts from factual verification to risk calculation and risk balancing. Third, authenticity undergoes Residualization, as technological systems establish authenticity by continuously identifying and excluding falsity. Consequently, authenticity no longer functions as a positive category requiring direct confirmation. Instead, it becomes a negative category established through the negation of falsity and exists only as the residual state left after falsity has been continuously identified and excluded. More importantly, these processes demonstrate that deepfake technologies are driving a profound transformation of the digital information environment: credibility is increasingly established through the exclusion of falsity rather than the confirmation of truth, thereby reshaping the fundamental logic through which authenticity is established and trust is sustained within the broader social information environment.

Keywords: Deepfake; Authenticity; Science and Technology Studies (STS); Critical Code Analysis; Technological Systems

Introduction

Technological systems have never been value-neutral tools. Rather, they actively participate in shaping social order and the production of knowledge. From photography and film to digital imaging, each transformation in media technology has reshaped how authenticity is understood and judged. Photography was once regarded as objective evidence of reality, whereas digital imaging gradually weakened the presumed correspondence between images and the world they represent. The rise of generative AI has accelerated this process. In particular, deepfake technologies can generate highly realistic audiovisual content at extremely low cost, making the boundary between the authentic and the fake increasingly difficult to discern.

In recent years, deepfake incidents have attracted global attention. From the fabricated "Zelenskyy surrender" video in 2022 and the fake "Pentagon explosion" image in 2023 to the growing number of AI-generated videos depicting politicians and public figures since 2024, deepfakes have not only generated significant communication risks but have also intensified the crisis of authenticity in digital media environments (Verma, 2025). An increasing body of research argues that generative AI is undermining the credibility of digital images as evidence and challenging the foundations of authenticity upon which journalism, public communication, and social trust have long depended (Tulin et al., 2025).

Unlike earlier media technologies, however, deepfake is itself a technological system organized around authenticity. On the one hand, generation systems continuously attempt to make fabricated content appear authentic. On the other hand, detection systems continuously seek to identify such fabrications. The ongoing interaction between generation and detection turns authenticity into the central object of technical organization, computation, and optimization. For this reason, deepfake generation and detection systems provide a particularly revealing case for understanding how authenticity is produced within technological systems.

Existing research on authenticity in the context of deepfakes has mainly developed along three strands. The first examines the crises of authenticity, social trust, and risks to the digital public sphere brought about by deepfakes (Verma, 2025). The second focuses on fact-checking, platform governance, and regulatory mechanisms to explore how authenticity can be restored (Birrer & Just, 2025; Hynek, 2025). The third seeks to improve the performance of generation and detection models by enhancing generation quality and detection accuracy (Gong & Li, 2024).

These studies have provided important insights into deepfake technologies. Overall, however, they tend to treat authenticity as an object to be restored, governed, or detected, while paying less attention to a more fundamental question: how is authenticity itself constructed within deepfake technological systems? In journalism and communication research in particular, discussions of deepfakes have largely focused on their communication effects and governance mechanisms. Far less attention has been paid to the technical infrastructure itself—to how authenticity judgments are encoded into code, models, and algorithms.

Science and Technology Studies (STS) has long argued that technologies are not value-neutral tools

but active participants in the construction of social order (Hackett et al., 2008). Technological systems not only shape how people act but also influence how they perceive the world, understand facts, and make judgments. From this perspective, the authenticity crisis associated with deepfakes stems not only from the proliferation of difficult-to-distinguish synthetic content, but also from the capacity of technological systems to redefine authenticity itself.

Understanding the authenticity crisis brought about by deepfakes therefore requires more than examining their communicative consequences. It also requires opening the technological "black box" to investigate how authenticity is organized and computed through the ongoing interaction between generation and detection systems.

Building on this perspective, this study draws on Philip Agre's (1997) Critical Technical Practice and adopts Critical Code Analysis (Marino, 2020) within an STS framework. It examines two representative open-source deepfake projects: the generation system FaceFusion and the detection benchmark DeepfakeBench. Unlike previous studies that primarily evaluate algorithmic performance, this study investigates how code organizes authenticity judgments and how authenticity is continuously negotiated through the interaction between generation and detection systems.

The findings suggest that the authenticity crisis associated with deepfake technologies stems not only from the proliferation of difficult-to-distinguish synthetic content but also from a fundamental reconfiguration of the technological mechanisms through which authenticity is produced.

More specifically, the study identifies three interrelated transformations of authenticity within deepfake technological systems. First, authenticity undergoes Proxyization, as verisimilitude gradually becomes the technological proxy for authenticity. Second, authenticity undergoes Riskification, as authenticity judgments increasingly rely on risk calculation and risk balancing rather than factual verification. Third, authenticity undergoes Residualization, as authenticity ceases to be the direct object of computation and is progressively reduced to the residual that remains after falsity has been continuously identified and excluded. Taken together, these three transformations demonstrate how deepfake technological systems are redefining how authenticity comes to be established in digital media environments.

Literature review

Technological systems as products of social negotiation

Since the 1960s and 1970s, the rapid development of nuclear technology, military technologies, industrialization, and large-scale technological systems has generated a range of environmental crises, societal risks, and governance challenges. These consequences prompted growing recognition that science and technology are not neutral forces operating independently of society but are deeply embedded in political, economic, cultural, and institutional relations. Against this backdrop, **Science and Technology Studies (STS)** emerged as an interdisciplinary field that situates scientific knowledge and technological systems within their social contexts, examining the power relations, value choices, and processes of social negotiation underlying technological development

(Hackett et al., 2008).

The critique of technological neutrality can be traced back to philosophical challenges to the instrumental view of technology. Heidegger (1977) argued that technology is not merely a tool but a mode of revealing through which the world is disclosed. Building on this insight, Winner (1980) further proposed that technological artefacts can embody particular forms of political power and social order. Pinch and Bijker (1984) subsequently moved this discussion into empirical inquiry by demonstrating that technological artefacts are not the products of autonomous technological evolution but are socially constructed through processes of interpretation, negotiation, and stabilization among relevant social groups. Actor–Network Theory (ANT) further challenged the binary distinction between humans and technology by conceptualizing technological systems as heterogeneous networks composed of human actors, technologies, institutions, texts, and standards. From this perspective, technological development is understood as an ongoing process of negotiation and co-construction among heterogeneous actors (Callon, 1986). Closely related to this line of inquiry, infrastructure studies have further shown that once technological systems become infrastructures, their embedded classifications, technical standards, and operational rules gradually sediment into taken-for-granted norms that continue to shape social practices in everyday life (Star and Ruhleder, 1996).

More recently, the rapid development of generative AI has further stimulated sociotechnical research on AI. Rather than being understood merely as a model or an algorithm, AI is increasingly conceptualized as a sociotechnical system composed of data, models, infrastructures, organizations, and social practices (Kudina & Van de Poel, 2024).

On the one hand, from the perspective of AI's technical properties, the data, models, and infrastructures that constitute AI systems all exhibit distinct characteristics of social construction. AI relies on large-scale training data, yet data themselves are not neutral materials but products of social relations and power structures (D'Ignazio & Klein, 2020). The extensive processes of data annotation, human feedback, and safety alignment involved in model training further embed social judgments into AI systems (Domínguez Hernández et al., 2026). Moreover, model architectures, algorithmic designs, and training objectives are continuously negotiated among technical performance, commercial interests, and application demands (Gillespie, 2014; Taeihagh, 2025). Beyond software, the development of AI also depends heavily on hardware infrastructures. Domínguez Hernández et al. (2026) conceptualize AI as a sociotechnical assemblage composed of computational resources, data infrastructures, data curation labor, and material resources. The concentration of computational resources not only shapes the development of AI capabilities but also exacerbates technological inequalities across countries and regions.

On the other hand, AI possesses not only technical properties but also distinctive product characteristics. Large generative AI models are essentially products developed and continuously operated by technology companies. As such, their training objectives and value orientations are inevitably shaped by corporate goals, regulatory requirements, and broader social responsibilities (Pierson et al., 2023). Meanwhile, the same AI model may assume different institutional roles and produce different social consequences when deployed in domains such as journalism, the judiciary,

education, and healthcare (Selgas-Cors, 2025). AI, therefore, should not be understood solely in terms of its technical properties. Rather, as a sociotechnical system, its meaning and functions are continuously defined and redefined through organizational contexts and social practices.

Empirical research on AI as a sociotechnical system can be broadly divided into two strands. The first strand takes humans as its primary unit of analysis, focusing on AI practitioners, data annotators, educators, AI governance practitioners, and users in specific domains. It examines issues such as AI's reflexive influence on human actors (Mahnke & Bagger, 2024), labor and value embedded in data annotation (Fu et al., 2026), and the challenges of AI responsibility and governance (Madaio et al., 2024). Studies within this strand predominantly employ qualitative methods, including ethnography and in-depth interviews.

The second strand moves beyond treating AI as a "black box" and instead turns to the AI system itself. It takes training data, models, training documentation, and model outputs as its objects of analysis, investigating issues such as the provenance of training data, data visibility (Baack, 2024), model transparency (Burkhardt & Rieder, 2024), and value alignment (Hadar-Shoval et al., 2024). Drawing on Critical Technical Practice as a methodological orientation, this line of research commonly adopts approaches such as governance audits, model audits, application audits (Mökander et al., 2024), and the walkthrough method (Light et al., 2018) to examine the operational logic of AI systems and the values embedded within them. The present study builds on this line of inquiry.

The construction of authenticity in technological systems

After recognizing that technological systems are not neutral, scholars soon began to examine how they shape authenticity. With the advent of photography, images entered the recording of reality through mechanical reproduction, giving rise to enduring debates on the relationship between media technologies and authenticity. Benjamin (2008) argued that mechanical reproduction transformed the mode of existence of artworks and gradually dissolved their traditional *aura*. Kittler (1999) further maintained that media technologies are not merely tools for recording reality but technological systems that reconfigure the conditions of perception, storage, and knowledge production. Subsequently, Baudrillard (1994) proposed the theory of *simulacra*, arguing that modern media no longer simply represent reality but instead produce it through mechanisms of simulation, rendering the boundary between the real and the fake increasingly unstable.

Generative AI has created unprecedented technological opportunities while simultaneously intensifying the impact of technological systems on authenticity. It enables the large-scale production of highly convincing synthetic content, giving rise to AI hallucinations, misinformation, and unpredictable behaviors (Williamson & Prybutok, 2024). Among these developments, deepfakes have emerged as one of the most pressing challenges to authenticity. They can generate both benign synthetic content and deceptive media intended to mislead audiences, posing unprecedented challenges to the verification of media authenticity (Farouk & Fahmi, 2024). As a consequence, deepfakes have further intensified information disorder in the post-truth era, triggered a broader crisis of authenticity (Verma, 2026), and undermined journalistic truth, public knowledge

production, and institutional trust (Tulin et al., 2025).

However, the significance of deepfake technologies extends beyond the increasing volume of false content and the governance challenges they pose (Birrer & Just, 2025). More fundamentally, they make images and videos—long regarded as relatively reliable forms of evidence—generable, manipulable, and highly realistic. In doing so, they undermine the epistemic status of visual media as evidence and repositories of public memory (Habgood-Coote, 2023) and destabilize existing regimes of authenticity and the boundary between the real and the fake at the epistemological level (Fallis, 2021).

Furthermore, if both deepfake generation systems and detection systems are technological systems shaped through social negotiation, then the authenticity judgments jointly produced by these systems likewise possess distinctive social characteristics. In other words, both the generation of deepfake content and its detection and verification are embedded with particular value orientations and together contribute to the construction of new boundaries between the real and the fake. However, existing research has focused primarily on the authenticity crisis caused by deepfakes and its governance (Birrer & Just, 2025; Hynek, 2025), while paying relatively little attention to how generation and detection systems jointly participate in the construction of authenticity. Both the evolving realities of generative AI and the STS perspective on the social nature of technological systems suggest the need to return to the technological systems themselves and examine how authenticity is continuously produced and reconfigured through the processes of generation and detection.

Accordingly, drawing on the STS framework, this study takes deepfake generation and detection systems as its objects of analysis to examine *how these technological systems reconfigure the boundary between the real and the fake and, in doing so, redefine authenticity.*

Methods

Methodological framework

Agre (1997) proposed Critical Technical Practice (CTP) as a representative methodological orientation within Science and Technology Studies (STS). At its core, CTP advocates integrating social, cultural, and political reflection into technical practice, moving beyond an engineering-oriented view that treats technology as a neutral tool and instead examining technological systems as sociotechnical assemblages shaped by values, power relations, and knowledge. Inspired by CTP's emphasis on critically engaging with technology from within, recent years have witnessed the emergence of a range of approaches that examine code, models, and algorithms as technical texts. Among these, Critical Code Studies (CCS) has become one of the most representative research traditions (Marino, 2020).

This line of research argues that code is not merely a technical tool for executing instructions but also a language that can be interpreted (Hua & Raley, 2023). As a form of media infrastructure, code itself invites critical reflection on technology, culture, and power (Levin & Brain, 2021). Through

its operation, it naturalizes particular abstract logics and knowledge structures, thereby producing the ordering of everyday social spaces (Dhulipalla, 2025). In other words, reading and interpreting code is itself an essential means of unpacking sociotechnical structures. Accordingly, Critical Code Analysis (CCA) approaches code as a cultural text and seeks to reveal its extra-functional significance (Marino & Douglass, 2023). In recent years, this method has been widely applied to studies of video games, feminist programming, and racial ethics in algorithms (Toupin, 2024; Ciston et al., 2025). For this study, authenticity judgments in deepfake technological systems are ultimately implemented through code. It is therefore necessary to return to the code itself to examine the technological construction of authenticity. Critical Code Analysis provides an appropriate methodological tool for this purpose.

To further examine how authenticity judgments are encoded within deepfake technological systems, this study adopts the methodological orientation of Critical Technical Practice and conceptualizes deepfake generation and detection systems as sites where technical practice intersects with social values. At the operational level, Critical Code Analysis (CCA) is employed as the primary research method to investigate how deepfake technological systems encode authenticity and, in doing so, construct the boundary between the real and the fake.

Kauffmann (2025) argues that Critical Code Analysis should first enable non-technical readers to understand what the code does before moving to cultural interpretation. Building on the work of Marino, he proposes a two-stage analytical procedure: technical interpretation followed by critical interpretation. He further develops a three-level analytical framework consisting *of performative analysis, cultural analysis, and materialist analysis*.

The above framework is primarily designed to interpret a single research object from multiple perspectives. In contrast, this study focuses on the interaction between two types of code within deepfake technological systems: the generation system and the detection system. Authenticity is not determined by either system alone. Instead, it is continuously constructed through the ongoing contest between the two. Applying this framework directly would separate the generation and detection processes into parallel analyses and make it difficult to capture their dialectical relationship in the construction of authenticity. At the same time, authenticity judgments in deepfake technological systems are ultimately implemented through code, and code itself constitutes a symbolic system. Accordingly, this study understands authenticity judgment as a process of symbolic construction. It therefore draws on Morris (1938) and his three-dimensional classification of sign systems—syntactics, semantics, and pragmatics. Based on this framework, Critical Code Analysis is organized into three analytical levels: syntactic, semantic, and pragmatic. At each level, Kauffmann's principle of technical interpretation followed by critical interpretation is retained. This framework makes it possible to better capture the relational construction of authenticity within the adversarial interaction between generation and detection systems.

At the syntactic level, this study examines code syntax, module interactions, and the organization of key functions. The aim is to explain how technological systems organize authenticity. At the semantic level, the analysis focuses on model architectures, core variables, and parameter design. The aim is to explain how technological systems define authenticity. At the pragmatic level, the

analysis examines the specific contexts of authenticity judgment, operational procedures, and threshold-setting strategies. The aim is to explain how technological systems determine authenticity.

Research objects and analytical procedures

To enable a systematic and evidence-based analysis of source code, this study adopted two criteria for selecting the research objects. First, the system should be mature, widely adopted, and capable of stable implementation in practice. Second, it should be open source and available as a complete project, allowing full access to the codebase for examination and analysis.

Based on these criteria, FaceFusion[1], one of the most widely used open-source face-swapping projects, was selected as the generation system. The project has received substantial community attention on GitHub (approximately 26.3K stars). It has remained actively maintained through 2026 and has developed a mature deployment ecosystem[2]. FaceFusion consists of four main modules: jobs, processors, uis, and workflows. Together, these modules comprise 234 source code files and support the complete generation pipeline. Specifically, the jobs module manages task scheduling, the processors module performs face analysis and image generation, the uis module handles user interaction, and the workflows module supports continuous integration and automated testing.

In analyzing the FaceFusion project, this study focuses on authenticity, which is the central concern of this research. Particular attention is given to the core source code involved in the face-swapping process, including face_analyser.py, face_classifier.py, and face_masker.py. The analysis also covers code related to authenticity detection, such as content_analyser.py. In addition, task management modules and the README documentation are examined, as they specify the project's architecture, functions, and configuration. Together, these materials provide important evidence for understanding how authenticity is encoded within the generation system. A complete list of the analyzed files is provided in Appendix A.

For the detection system, DeepfakeBench was selected to provide a directly comparable counterpart to the generation system and to ensure the comparability and reproducibility of the analysis[3]. DeepfakeBench is a unified benchmarking framework for deepfake detection. It standardizes data management, model implementation, and evaluation protocols, thereby reducing inconsistencies in data processing, experimental settings, and evaluation metrics across different detection methods. The project also has strong academic recognition. It consists of five main modules—analysis, datasets, figures, preprocessing, and training—which together contain 242 source code files. The analysis module evaluates model performance and generates visualizations. The datasets module manages the benchmark datasets. The figures module contains visual outputs. The preprocessing module prepares the input data. The training module includes the core code for model training and evaluation. This study focuses primarily on the code in the analysis and training modules that is directly related to authenticity detection and judgment. A complete list of the analyzed files is provided in Appendix B.

---

[1] Open-source project repository: https://github.com/facefusion/facefusion
[2] Official project website: https://facefusion.io/
[3] Open-source project repository: https://github.com/SCLBD/DeepfakeBench

The analysis was conducted independently by two researchers with experience in source code interpretation. They translated the technical syntax of the generation and detection systems into structured and accessible descriptions while preserving the original computational logic as accurately as possible. Their analyses were highly consistent. The results were then compared and jointly discussed in light of the research questions. Based on this process, a critical interpretation was subsequently developed to enhance the credibility and rigor of the findings.

Results and discussion

The proxyization of authenticity

In a truth-default environment, people generally presume that the information they receive is true rather than questioning its authenticity at the outset. Levine (2014) argues that this default assumption remains in place unless sufficient cues trigger suspicion. Deepfake technological systems, however, alter the technical logic of authenticity judgment. Through the continuous interaction between generation and detection systems, authenticity is no longer defined primarily by its correspondence with objective reality. Instead, it is increasingly mediated by verisimilitude. In other words, technological systems do not determine whether content is authentic. They determine whether it appears authentic.

From the perspective of the generation system, authenticity is not something to be verified but a visual effect to be simulated. The source code continuously optimizes the generated content to eliminate detectable flaws and align it with the authenticity cues embedded in everyday experience. More specifically, this process of verisimilitude is organized around four dimensions: boundary, motion, identity, and context. Representative variables are presented in Table 1. Typical operations include identity anchoring, multimodal consistency, temporal continuity, contextual plausibility, and artifact management.

Table 1. Representative Variables Supporting Verisimilitude in the Generation System

Coding Category	Representative Variable	Function
Boundary Plausibility	*face_mask_blur*	Controls the feathering intensity of the face mask boundary.
	face_mask_padding	Controls the padding around the mask (top, bottom, left, and right).
Motion Plausibility	*expression_restorer_factor*	Controls the blending ratio between the target expression and the current facial expression.
	source_voice_frame	Specifies the key frame for lip-sync alignment.
Identity Plausibility	*lip_syncer_weight*	Controls the strength of lip synchronization.
	reference_face_position	Specifies the position of the reference face in multi-face scenes.
	current_face_distance	Measures the vector distance between the current face and the reference face.
Contextual Plausibility	*color_difference_vision_frame*	Generates a pixel-level color difference map.
	output_video_fps	Ensures consistency between the output and source video frame rates.

	output_video_scale	Maintains consistency between the output resolution, scaling ratio, and the source video.

In contrast to the generation system, the detection system does not directly verify whether content is authentic. Instead, it determines whether the generated content contains suspicious traces. Accordingly, what the detection system understands as **"authentic"** is, in practice, **"not suspicious."** If the generated content maintains sufficient consistency in terms of **boundary, motion, identity,** and **context**, it is likely to pass authenticity detection. Representative variables are presented in **Table 2**.

Frequency-domain analysis provides a typical example. **High-frequency filters** examine fine-grained textures, such as skin details, pores, and boundary continuity. Their primary function is to assess **boundary plausibility**. **Mid-frequency filters** inspect meso-level structures, including facial contours and shadow transitions, to evaluate **identity plausibility**. **Low-frequency filters** analyze the overall silhouette, illumination distribution, color tone, and other global imaging characteristics. They are mainly used to assess **contextual plausibility**.

Table 2. Representative Variables in the Detection System

Coding Category	Representative Variable	Function
Boundary Plausibility	*mask_pred*	Predicts the boundary mask for the manipulated region.
	mask_gt	Provides the ground-truth supervision signal for the transition region, enabling the model to detect inconsistencies between the manipulated area and the original image.
	high_filter	Applies high-frequency filtering to detect anomalies in high-frequency components.
Motion Plausibility	*video_level_features*	Evaluates the stability and credibility of overall motion patterns.
	time_size	Models whether frame-to-frame temporal changes are natural.
Identity Plausibility	*loss_eic*	Defines the identity consistency loss used during model optimization.
	explicit_extractor	Extracts explicit identity features as references for identity consistency detection.
	middle_filter	Applies mid-frequency filtering to detect anomalies in medium-frequency components.
Contextual Plausibility	*high_filter*	Applies low-frequency filtering to assess the consistency of global contextual information.

It is evident that both the generation and detection systems operate around **verisimilitude**, rather than authenticity itself. The generation system continuously improves the degree of verisimilitude, whereas the detection system continuously searches for imperfections in it. The two systems are engaged in an ongoing contest based on the same standards of verisimilitude. Throughout this process, authenticity is no longer the direct object of technical operations. Instead, it is progressively displaced by verisimilitude.

This transformation fundamentally changes the logic of authenticity judgment in deepfake technological systems. In traditional media environments, authenticity is primarily established

through correspondence between content and objective reality. Authenticity judgments therefore rely on fact-checking, source verification, and evidence comparison. Deepfake-generated content, however, often lacks an original reference against which it can be directly verified. At the same time, technological systems cannot algorithmically verify factual reality itself. They can only compute the degree of verisimilitude.

As a result, within deepfake technological systems, authenticity no longer primarily refers to correspondence with reality. Instead, it comes to mean **being sufficiently verisimilar**. Authenticity gradually recedes into the background, while verisimilitude becomes its direct proxy. The escalating technological competition between generation and detection systems is therefore not a competition over authenticity itself, but over the continuous optimization of verisimilitude. Authenticity judgment thus shifts from **authenticity** to **verisimilitude**. Authenticity becomes represented through verisimilitude, a process that this study conceptualizes as the **Proxyization of Authenticity**.

The riskification of authenticity

Analysis of the source code for both the generation and detection systems shows that authenticity judgment in deepfake technological systems has, in effect, been transformed into a form of risk calculation. The generation system focuses on minimizing the risk of exposure, whereas the detection system focuses on balancing different types of error risks.

From the perspective of the generation system, the code does not seek to maximize the extent of face replacement. Instead, it prioritizes the control of exposure risk. To achieve this, FaceFusion constructs four types of masks simultaneously: a box mask, an occlusion mask, an area mask, and a region mask. The box mask smooths boundary transitions to prevent jagged edges. The occlusion mask preserves occlusion relationships by maintaining the spatial hierarchy between facial regions and objects such as hair, hands, and glasses, thereby preventing artifacts caused by incorrect occlusions. The area mask uses facial landmarks to define a geometrically constrained replacement region, limiting manipulation to more controllable areas. The region mask relies on a face parsing model to identify facial components, such as skin, eyes, lips, and the nose, allowing the system to avoid regions that are more likely to reveal manipulation or to concentrate replacement on relatively safe regions.

How does the algorithm choose among these four masks? The code shows that the generation system neither selects the optimal mask nor seeks a compromise among them. Instead, it adopts a most conservative masking strategy. As shown in Figure 1, the system first generates the four masks according to the configuration specified in face_mask_types. It then applies the numpy.minimum.reduce function to compute the pixel-wise minimum value across all masks. These values can be understood as replacement permissions. Values closer to 1 allow more extensive replacement, whereas values closer to 0 preserve the original content. Taking the minimum value means that if any one of the four masks identifies a region as being more likely to expose manipulation, the replacement weight for that region is reduced according to the strictest criterion, even if the other masks permit replacement. The region may even be excluded from manipulation altogether. This implements a veto-based decision logic within the generation system. Once a region is assessed as posing a higher exposure risk, the system tends to replace

less, replace more conservatively, or avoid replacement entirely. In other words, the generation system does not pursue the best possible generation result. Instead, it prioritizes avoiding the worst possible outcome.

```
        if 'box' in state_manager.get_item('face_mask_types'):
            box_mask = create_box_mask(crop_vision_frame, state_manager.get_item('face_mask_blur'), state_manager.get_item('face_mask_padding'))
            crop_masks.append(box_mask)

        if 'occlusion' in state_manager.get_item('face_mask_types'):
            occlusion_mask = create_occlusion_mask(crop_vision_frame)
            crop_masks.append(occlusion_mask)

        pixel_boost_vision_frames = implode_pixel_boost(crop_vision_frame, pixel_boost_total, model_size)
        for pixel_boost_vision_frame in pixel_boost_vision_frames:
            pixel_boost_vision_frame = prepare_crop_frame(pixel_boost_vision_frame)
            pixel_boost_vision_frame = forward_swap_face(source_face, target_face, pixel_boost_vision_frame)
            pixel_boost_vision_frame = normalize_crop_frame(pixel_boost_vision_frame)
            temp_vision_frames.append(pixel_boost_vision_frame)
        crop_vision_frame = explode_pixel_boost(temp_vision_frames, pixel_boost_total, model_size, pixel_boost_size)

        if 'area' in state_manager.get_item('face_mask_types'):
            face_landmark_68 = cv2.transform(target_face.landmark_set.get('68').reshape(1, -1, 2), affine_matrix).reshape(-1, 2)
            area_mask = create_area_mask(crop_vision_frame, face_landmark_68, state_manager.get_item('face_mask_areas'))
            crop_masks.append(area_mask)

        if 'region' in state_manager.get_item('face_mask_types'):
            region_mask = create_region_mask(crop_vision_frame, state_manager.get_item('face_mask_regions'))
            crop_masks.append(region_mask)

        crop_mask = numpy.minimum.reduce(crop_masks).clip(0, 1)
        paste_vision_frame = paste_back(temp_vision_frame, crop_vision_frame, crop_mask, affine_matrix)
        return paste_vision_frame
```

Figure 1. Core Code Implementing the Four Mask Types and the Veto-Based Decision Logic[4]

This conservative strategy can be understood as a form of exposure risk management. Rather than pursuing more aggressive manipulations at the cost of a higher probability of detection, the algorithm is designed to minimize the likelihood of being detected. Research on deception has long argued that being detected is not merely an external consequence that occurs after deception. Instead, it is a risk cost that must be anticipated during strategic planning and taken into account before action is undertaken in order to maintain credibility (Greene et al., 1985). Accordingly, for the generation system, authenticity is not the objective of generation but the outcome of exposure risk control. What the code actually computes is not truth, but the probability of exposure.

How, then, does the detection system determine whether content is authentic or fake? Authenticity judgment is first translated into a probability of suspiciousness. A classification decision is then made by applying a decision threshold. If the suspiciousness probability exceeds the threshold, the sample is classified as fake. Otherwise, it is classified as authentic.

Changing the threshold inevitably changes the balance between false positives and false negatives. This raises an important question: what constitutes an optimal threshold? In practice, every threshold embodies a particular ordering of error costs. It reflects which type of error the detection system is designed to avoid: falsely classifying authentic content as fake, or failing to detect manipulated content.

Following the logic of the generation system, one might expect that the detection system would

[4] **Source:** The code is from *core.py* in the *face_swapper* module of the **FaceFusion** project.

prioritize avoiding false negatives, since the generation system is primarily concerned with minimizing exposure risk. However, the pragmatic analysis suggests otherwise. At least in this project, the evaluation framework does not assign a higher priority to false negatives. Instead, false negatives and false positives are treated as equally important types of error.

Specifically, the project adopts Equal Error Rate (EER) as an evaluation metric. EER is calculated by searching across all possible decision thresholds to identify the point at which the false negative rate (FNR) and the false positive rate (FPR) are as close as possible. The error rate at this point is recorded as the EER, as illustrated in Figure 2. This procedure indicates that the detection system does not assume that either type of error is inherently more unacceptable. Rather, it seeks a balance between the two and uses this balance as a symmetrical technical criterion for threshold selection.

```
64            fpr, tpr, thresholds = metrics.roc_curve(new_label, new_pred)
65            v_auc = metrics.auc(fpr, tpr)
66            fnr = 1 - tpr
67            v_eer = fpr[np.nanargmin(np.absolute((fnr - fpr)))]
68            return v_auc, v_eer
```

Figure 2. Core Code for Equal Error Rate (EER) Calculation[5]

Threshold setting is fundamentally a process of risk allocation. By selecting different decision thresholds, algorithm designers assign different weights to different types of errors and determine the relative importance of false positives and false negatives. For example, in scenarios such as public opinion governance and fact-checking during public events, falsely classifying authentic content as manipulated may directly harm individual rights and undermine institutional credibility. In such cases, false positives are typically assigned a higher cost. The system therefore tolerates a certain number of suspicious samples in order to avoid misclassifying authentic content. By contrast, in high-risk scenarios such as financial fraud or large-scale AI-enabled content manipulation, undetected fake content may enter the communication chain and produce substantial real-world harm. In these situations, false negatives are usually assigned a higher cost to improve the detection rate, even at the expense of increasing false positives. Consequently, there is no single optimal decision threshold. Its selection ultimately depends on how different risks are weighed and prioritized.

Taken together, the analyses of the generation and detection systems reveal that, although the two systems pursue opposite objectives, both operate according to a common logic of risk management. The generation system improves verisimilitude by minimizing exposure risk, whereas the detection system reaches authenticity judgments by managing the risks associated with false positives and false negatives. Whether in the form of exposure risk or error risk, these uncertainties are ultimately translated into a set of computable and allocable risk variables. As a result, authenticity judgment is no longer grounded primarily in factual verification, but in risk calculation and risk allocation. Accordingly, judging whether content is authentic or fake no longer primarily concerns whether it corresponds to reality, but whether the associated risk is considered acceptable.

[5] **Source:** The code excerpt is taken from *training/metrics/utils.py* in the **DeepfakeBench** project.

The residualization of authenticity

Further analysis of deepfake technological systems reveals that authenticity is not only **proxied by verisimilitude** and **reconfigured through risk calculation**, but also gradually recedes into a **residual** category that exists only in relation to falsity. From a dialectical perspective, authenticity and falsity are mutually opposed yet mutually constitutive. Each derives its meaning through its relation to the other. Within deepfake technological systems, however, this relatively balanced relationship is transformed. Authenticity is no longer affirmed directly. Instead, it is continuously established through the identification and exclusion of falsity. In this sense, authenticity is confirmed only as **"not fake,"** becoming the residue that remains after falsity has been filtered out.

More specifically, the generation system continuously computes how to produce content that appears sufficiently authentic. Yet authenticity here is not measured by correspondence with factual reality. Instead, it is evaluated by the absence of detectable flaws and the appearance of **not being fake**. The generation system therefore continuously removes traces that might reveal manipulation, keeping the generated content below the threshold of suspicion. The logic is even more explicit in the detection system, whose primary task is to detect fake content. Rather than positively demonstrating why content is authentic, the system continuously searches for indicators of falsity and classifies content accordingly.

Taking the detection system as an example, when the model processes a sample, it first computes two prediction scores (the variable **pred**), corresponding to the tendencies of being **more authentic** and **more fake**, respectively. These scores are then converted into probabilities through the **Softmax** function, which constrains their sum to equal one, as shown in **Figure 3.**

```
109      def forward(self, data_dict: dict, inference=False) -> dict:
110          # get the features by backbone
111          features = self.features(data_dict)
112          # get the core_feat for loss
113          core_feat = nn.ReLU(inplace=False)(features)
114          core_feat= F.adaptive_avg_pool2d(core_feat, (1, 1))
115          core_feat = core_feat.view(core_feat.size(0), -1)
116          # get the prediction by classifier
117          pred = self.classifier(features)
118          # get the probability of the pred
119          prob = torch.softmax(pred, dim=1)[:, 1]
120          # build the prediction dict for each output
121          pred_dict = {'cls': pred, 'prob': prob, 'feat': features, 'core_feat': core_feat}
122
123          return pred_dict
```

Figure 3. Core Execution Flow of the Core Detector (core_detector.py)

As shown in the code, **"more authentic"** and **"more fake"** are encoded as two interdependent variables within the same probabilistic structure. An increase in one necessarily corresponds to a decrease in the other. Within this balanced structure, the system typically takes the probability of **"being fake"** as the variable **prob**, which is interpreted as the **probability of suspiciousness**. A

decision threshold is then applied to classify the sample. If the suspiciousness probability falls below the threshold, the sample is provisionally classified as authentic. Once it exceeds the threshold, the sample is classified as fake. It is important to note that the threshold is not a natural constant. Rather, it is a technical parameter that can be continuously configured and updated. As the threshold changes, content previously classified as authentic may subsequently be reclassified as fake. Authenticity therefore does not exist as an independently established positive category. Instead, it is preserved only as the residual that remains outside suspiciousness. Particularly within the detection system, where the primary objective is to identify fake content, authenticity is not established through positive verification. Rather, it is confirmed through the continuous exclusion of falsity.

The notion of the **residual** occupies a distinctive position in **Baudrillard's** theoretical framework. Baudrillard argues that the residual is not simply what is left over. Rather, it is produced through processes of negation. As he writes, *"It is a positivity, but only negativity gives it the force of reality… When the sexual act becomes the product of sexual relations, the sexual act becomes residual. When society becomes the product of social relations, society itself becomes residual"* (Baudrillard, 1981/2025, pp. 270–276). The residual, therefore, is not something that naturally remains. It is the outcome of abstraction and institutionalized operations.

The same logic applies to deepfake technological systems. Authenticity increasingly assumes the status of a **residual**. As deepfake generation and detection technologies continue to evolve around **falsity**, the object that technological systems organize, compute, and optimize is consistently **falsity** itself. Whether the generation system attempts to eliminate traces of manipulation or the detection system continuously identifies the risk of falsity, the systems never compute **authenticity** directly. Instead, they continuously compute **falsity**. What counts as authentic is simply the content that has **not yet been classified as fake**.

The relationship between authenticity and falsity is therefore reversed. In conventional authenticity judgments, authenticity serves as the point of departure, while falsity is understood as a deviation from authenticity. Within deepfake technological systems, however, **falsity** becomes the central object of technical computation and governance, whereas **authenticity** is redefined as **"not fake."** Authenticity can no longer exist as an independent and self-evident state. Instead, it is continuously confirmed through the identification, exclusion, and negation of falsity. Authenticity thus ceases to be an object directly produced or verified by technological systems. Instead, it becomes the **residual** left behind after falsity has been continuously identified and filtered out. **This is what this study conceptualizes as the *Residualization of Authenticity*.**

Conclusion

This study examines deepfake generation and detection systems from the perspective of Science and Technology Studies (STS). Drawing on Critical Code Analysis, it analyzes the core source code of two representative deepfake projects, FaceFusion and DeepfakeBench, to investigate how authenticity is organized within technological systems. The findings suggest that the authenticity crisis brought about by deepfake technologies stems not only from the proliferation of difficult-to-

distinguish fake content, but also from a fundamental reconfiguration of the technological logic through which authenticity is produced. Rather than computing authenticity directly, deepfake technological systems continuously compute falsity. Authenticity is established only indirectly through the identification, risk assessment, and exclusion of falsity. As a result, authenticity is proxied by verisimilitude, validated through risk calculation, and ultimately reduced to the residual left behind after falsity has been continuously filtered out. Deepfake technological systems therefore do more than transform the technical process of distinguishing the authentic from the fake. They also redefine how authenticity comes to be established.

More specifically, the study identifies three interrelated transformations of authenticity. First, authenticity undergoes Proxyization. Both the generation and detection systems continuously optimize and evaluate verisimilitude rather than authenticity itself. The object of technical operation is not authenticity, but the degree of verisimilitude. Accordingly, authenticity judgment gradually shifts from verifying factual authenticity to assessing verisimilitude, which becomes the technological proxy for authenticity.

Second, authenticity undergoes Riskification. Whether through the generation system's management of exposure risk using masking strategies or the detection system's management of error risk through thresholds, false positives, and false negatives, authenticity judgment no longer centers on factual verification. Instead, it becomes a decision-making process based on risk calculation and risk prioritization. Judging whether content is authentic or fake thus shifts from factual correspondence to risk assessment.

Third, authenticity undergoes Residualization. Within deepfake technological systems, computation is continuously organized around falsity, while authenticity itself is never directly computed. Authenticity increasingly comes to mean "not fake," becoming the residual that remains after falsity has been continuously identified and filtered out. Authenticity therefore ceases to be an object directly produced or verified by technological systems. Instead, it becomes a relational concept that exists only through its dependence on falsity.

Taken together, these three findings demonstrate that deepfake technological systems reconfigure the very mechanism through which authenticity is produced. Authenticity is transformed from an epistemological concept grounded in correspondence with reality into a sociotechnical outcome jointly organized by code, models, and algorithms. This transformation also reshapes the logic through which authenticity is established in digital media environments. In traditional communication contexts, authenticity is primarily established through fact-checking, source verification, and evidence comparison, all of which seek to verify correspondence between content and objective reality. In the context of deepfake technologies, however, authenticity increasingly takes the form of confirmatory exclusion. Content is regarded as authentic not primarily because it has been verified as true, but because it has not yet been identified as fake. Emerging technical practices—including AI detection, content provenance, digital watermarking, and Content Credentials—are all fundamentally oriented toward excluding falsity rather than proving authenticity. Consequently, the confirmation of authenticity is gradually shifting from verification to the exclusion of falsity. Authenticity is no longer treated as an intrinsic property of content, but as a sociotechnical condition that requires continuous confirmation by technological systems.

This study contributes to the literature in two respects.

First, at the theoretical level, it repositions authenticity within the Science and Technology Studies (STS) framework of sociotechnical systems. Rather than approaching authenticity through the external governance perspectives that dominate existing research—such as authenticity crises, fact-checking, and platform governance—it explains how authenticity is constructed from within technological systems. In doing so, it extends sociotechnical approaches to the study of authenticity.

Second, at the methodological level, this study brings Critical Technical Practice and Critical Code Analysis into deepfake research. By examining code, models, and algorithms, it opens the technological "black box" and demonstrates how authenticity judgments are encoded into technological systems. This approach provides a replicable analytical framework for studying generative AI systems and offers methodological guidance for future communication research that investigates large language models, AI agents, and other emerging sociotechnical systems at the level of code.

Data availability

The materials underpinning this study consist of publicly available source code and technical documentation from the FaceFusion and DeepfakeBench repositories. The original source materials are available at https://github.com/facefusion/facefusion and https://github.com/SCLBD/DeepfakeBench. The versions and repository snapshots analysed in this study are specified in the Supplementary Information, which also includes the selected code-file inventories, analytical protocol, evidence-to-finding matrix, and documentation of the analytical procedures. No new datasets were generated, and no human-participant, personal, confidential, or proprietary data were used in this study.

References

Agre, P. E. (1997). Toward a critical technical practice: Lessons learned in trying to reform AI. In G. C. Bowker, L. Gasser, S. L. Star, & W. Turner (Eds.), *Social science, technical systems, and cooperative work: Beyond the great divide* (pp. 131–157). Lawrence Erlbaum Associates.

Baack, S. (2024, June). A critical analysis of the largest source for generative AI training data: Common Crawl. In *Proceedings of the 2024 ACM Conference on Fairness, Accountability, and Transparency* (pp. 2199–2208). ACM. https://doi.org/10.1145/3630106.3659033.

Baudrillard, J. (1994). *Simulacra and simulation* (S. F. Glaser, Trans.). University of Michigan Press.

Benjamin, W. (2008). *The work of art in the age of mechanical reproduction* (J. A. Underwood, Trans.). Penguin Books.

Birrer, A., & Just, N. (2025). What we know and don't know about deepfakes: An investigation into the state of the research and regulatory landscape. *New media & society, 27*(12), 6819-6838. https://doi.org/10.1177/14614448241253138

Burkhardt, S., & Rieder, B. (2024). Foundation models are platform models: Prompting and the political economy of AI. *Big Data & Society, 11*(2), 20539517241247839. https://doi.org/10.1177/20539517241247839.

Callon, M. (1986). Some elements of a sociology of translation: Domestication of the scallops and the fishermen of St. Brieuc Bay. In J. Law (Ed.), *Power, action and belief: A new sociology of knowledge?* (pp. 196–233). Routledge & Kegan Paul.

Ciston, S., Mann, Z., Marino, M. C., & Douglass, J. (2025). Can open-source fix predictive policing? Anti-racist critical code studies approach to contemporary AI policing software. *Digital Humanities Quarterly, 19*(1).

Dhulipalla, K. (2025). Vernacular computing as encoded aesthetics for decolonial code intervention. *AI & Society, 41*, 2683–2695. https://doi.org/10.1007/s00146-025-02689-w

Domínguez Hernández, A., Perini, A. M., Hadjiloizou, S., Borda, A., Mahomed, S., & Leslie, D. (2026). Towards a sociotechnical ecology of artificial intelligence: power, accountability, and governance in a global context. *AI and Ethics, 6*(1), 43. https://doi.org/10.1007/s43681-025-00902-6

D'Ignazio, C., & Klein, L. F. (2020). *Data Feminism*. MIT Press.

Fallis, D. (2021). The epistemic threat of deepfakes. *Philosophy & technology, 34*(4), 623-643. https://doi.org/10.1007/s13347-020-00419-2

Farouk, M. A., & Fahmi, B. M. (2024). Deepfakes and media integrity: Navigating the new reality of synthetic content. *Journal of Media and Interdisciplinary Studies, 3*(9), 47-94.

Fu, P., Lin, Z., & Wang, W. Y. (2026). Operationalizing AI governance: data annotation, La Qi and manual alignment in China. Information, *Communication & Society, 29*(6), 1813-1835. https://doi.org/10.1080/1369118X.2025.2526162

Gillespie, T. (2014). The relevance of algorithms. In T. Gillespie, P. J. Boczkowski, & K. A. Foot (Eds.), *Media technologies: Essays on communication, materiality, and society* (pp. 167–194). MIT Press.

Gong, L. Y., & Li, X. J. (2024). A contemporary survey on deepfake detection: Datasets, algorithms, and challenges. *Electronics, 13*(3), 585. https://doi.org/10.3390/electronics13030585

Habgood-Coote, J. (2023). Deepfakes and the epistemic apocalypse. *Synthese, 201*(3), 103. https://doi.org/10.1007/s11229-023-04097-3

Hackett, E. J., Amsterdamska, O., Lynch, M., & Wajcman, J. (Eds.). (2008). *The Handbook of Science and Technology Studies*. MIT Press.

Hadar-Shoval, D., Asraf, K., Mizrachi, Y., Haber, Y., & Elyoseph, Z. (2024). Assessing the alignment of large language models with human values for mental health integration: Cross-sectional study using Schwartz's theory of basic values. *JMIR Mental Health*, 11, e55988. https://doi.org/10.2196/55988

Heidegger, M. (1977). The question concerning technology. In *The question concerning technology and other essays* (W. Lovitt, Trans., pp. 3–35). Harper & Row. (Original work published 1954).

Hua, M., & Raley, R. (2023). *How to do things with deep learning code* [Preprint]. arXiv. https://doi.org/10.48550/arXiv.2304.09406

Hynek, N., Gavurova, B., & Kubak, M. (2025). Risks and benefits of artificial intelligence deepfakes: Systematic review and comparison of public attitudes in seven European Countries. *Journal of Innovation & Knowledge, 10*(5), 100782. https://doi.org/10.1016/j.jik.2025.100782

Kauffmann, T. (2025). Leveraging LLMs for interpreting historical source code: A case study of the Apple Lisa through critical code studies. *AI & Society*. Advance online publication. https://doi.org/10.1007/s00146-025-02786-w.

Kittler, F. A. (1999). *Gramophone, film, typewriter* (G. Winthrop-Young & M. Wutz, Trans.). Stanford University Press.

Kudina, O., & Van De Poel, I. (2024). A sociotechnical system perspective on AI. *Minds and Machines, 34*(3), 21. https://doi.org/10.1007/s11023-024-09680-2

Levin, G., & Brain, T. (2021). *Code as creative medium: A handbook for computational art and design*. MIT Press..

Levine, T. R. (2014). Truth-default theory (TDT): A theory of human deception and deception detection. *Journal of Language and Social Psychology, 33*(4), 378–392. https://doi.org/10.1177/0261927X14535916

Light, B., Burgess, J., & Duguay, S. (2018). The walkthrough method: An approach to the study of apps. *New Media & Society, 20*(3), 881–900. https://doi.org/10.1177/1461444816675438

Madaio, M., Kapania, S., Qadri, R., Wang, D., Zaldivar, A., Denton, R., & Wilcox, L. (2024, June). Learning about responsible AI on-the-job: Learning pathways, orientations, and aspirations. In *Proceedings of the 2024 ACM Conference on Fairness, Accountability, and Transparency* (pp. 1544–1558). ACM. https://doi.org/10.1145/3630106.3658988

Mahnke, M. S., & Bagger, C. (2024). Navigating platformized generative AI: Examining early adopters' experiences through the lens of data reflectivity. *Convergence, 30*(6), 1974–1991. https://doi.org/10.1177/13548565241300085

Marino, M. C. (2020). *Critical code studies*. MIT Press.

Marino, M. C., & Douglass, J. (2023). Introduction: Situating critical code studies in the digital humanities. *Digital Humanities Quarterly, 17*(2).

Morris, C. W. (1938). Foundations of the theory of signs. In *International encyclopedia of unified science* (Vol. 1, No. 2, pp. 1–59). University of Chicago Press.

Mökander, J., Schuett, J., Kirk, H. R., & Floridi, L. (2024). Auditing large language models: A three-layered approach. *AI and Ethics, 4*(4), 1085–1115. https://doi.org/10.1007/s43681-023-00289-2

Pierson, J., Kerr, A., Robinson, S. C., Fanni, R., Steinkogler, V. E., Milan, S., & Zampedri, G. (2023). Governing artificial intelligence in the media and communications sector. *Internet Policy Review, 12*(1), 1–28. https://doi.org/10.14763/2023.1.1683

Pinch, T. J., & Bijker, W. E. (1984). The social construction of facts and artefacts: Or how the sociology of science and the sociology of technology might benefit each other. *Social Studies of Science, 14*(3), 399–441. https://doi.org/10.1177/030631284014003004

Selgas-Cors, M. (2025). Sociotechnical transformation: A systematic review on the impact of artificial intelligence on society and organizations. *FinTech and Sustainable Innovation*. https://doi.org/10.47852/bonviewFSI52026076

Star, S. L., & Ruhleder, K. (1996). Steps toward an ecology of infrastructure: Design and access for large information spaces. *Information Systems Research, 7*(1), 111–134. https://doi.org/10.1287/isre.7.1.111

Taeihagh, A. (2025). Governance of generative AI. *Policy & Society, 44*(1), 1–22. https://doi.org/10.1093/polsoc/puaf001

Toupin, S. (2024). Shaping feminist artificial intelligence. *New Media & Society, 26*(1), 580–595. https://doi.org/10.1177/14614448221150776

Tulin, M., Pantazi, M., Starke, C., Sivolap, M., & Dobber, T. (2025). Generative AI and disinformation: Echoes of doubt: Exposure to information about generative AI decreases believability of news. *International Journal of Communication, 19*, 3712–3735.

Verma, A. (2026). Deepfakes and the crisis of digital authenticity: Ethical challenges in the age of synthetic media. *Journal of Information, Communication and Ethics in Society, 24*(1), 59–76. https://doi.org/10.1108/JICES-04-2025-0083

Williamson, S. M., & Prybutok, V. (2024). The era of artificial intelligence deception: Unraveling the complexities of false realities and emerging threats of misinformation. *Information, 15*(6), 299. https://doi.org/10.3390/info15060299

Winner, L. (2017). Do artifacts have politics? In D. G. Johnson & H. T. Nissenbaum (Eds.), *Computer ethics* (4th ed., pp. 177–192). Routledge.

Appendix A. Selected Key Code Files in the Generation Module

File Name	Module	Primary Function	Notes
README	Documentation	Introduces the project and provides usage instructions.	Contains basic project information and installation guidelines.
facefusion.py	Main Program	Serves as the project entry point, containing the main execution logic and module imports.	Includes core configurations and startup functions.
core.py	Core Module	Provides core functionalities and utility functions for basic system logic.	Includes commonly used helper functions and classes.
camera_manager.py	Camera Management	Manages local and remote cameras and provides video stream interfaces.	Handles camera input and video stream management.
choices.py	Configuration Module	Defines configurable options for models, face detection methods, audio formats, and other settings.	Provides selectable configuration options.
common_helper.py	Utility Module	Provides common helper functions, including platform checks and data conversion utilities.	Supports cross-platform functions such as memory limitation and range generation.
content_analyser.py	Content Analysis	Analyzes input content for authenticity-related characteristics.	Used for content analysis and authenticity assessment.
execution.py	Execution Module	Manages inference sessions and execution workflows.	Handles model execution and task scheduling.
face_analyser.py	Face Analysis	Extracts facial features from input images.	Depends on external image processing libraries.
face_classifier.py	Face Classification	Classifies extracted facial features to ensure they satisfy predefined conditions.	Relies on pretrained face classification models.
face_detector.py	Face Detection	Detects faces in input images and identifies their locations.	Supports multiple detection models (e.g., RetinaFace).
face_helper.py	Face Utilities	Provides helper functions for facial processing, such as landmark extraction.	Supports face detection and feature processing.
face_landmarker.py	Face Landmark Detection	Detects facial landmarks and generates key-point information.	Depends on facial landmark detection algorithms.
face_masker.py	Face Mask Generation	Generates facial masks for subsequent face-swapping operations.	Computationally intensive.
face_recognizer.py	Face Recognition	Identifies and matches facial identities using facial embeddings.	Uses pretrained face recognition models.
face_selector.py	Face Selection	Selects one or multiple faces for subsequent processing.	Supports different selection strategies (e.g., left-to-right, best match).
face_store.py	Face Storage	Stores facial data, including embeddings and extracted features.	Manages persistent facial feature storage.
inference_manager.py	Inference Management	Coordinates inference sessions and manages inference tasks.	Responsible for session creation and destruction.
process_manager.py	Process Management	Manages task status	Controls task lifecycle,

		throughout execution.	including processing, checking, and termination.
program.py	Program Entry	Main executable file of the project.	Contains the overall execution logic and entry point.
program_helper.py	Program Utilities	Provides program control functions such as startup and shutdown.	Supports lifecycle management.
state_manager.py	State Management	Maintains system states, task status, and configuration information.	Responsible for state persistence and restoration.
types.py	Type Definitions	Defines commonly used data types and type aliases.	Improves type annotation and code readability.
time_helper.py	Time Utilities	Provides time-related helper functions.	Supports timestamp retrieval and time interval calculation.
translator.py	Translation	Provides multilingual support through localization and translation loading.	Used for internationalization (i18n).
video_manager.py	Video Management	Handles video reading, writing, conversion, and frame processing.	Manages video input/output operations.
vision.py	Computer Vision	Performs computer vision tasks, including image processing and feature extraction.	Responsible for image and video processing.
voice_extractor.py	Voice Extraction	Extracts speech signals from audio inputs.	Used for speech separation and subsequent audio processing.
jobs/	Job Management	Stores and manages task workflows and execution status.	Organizes and coordinates task execution.
processors/	Processing Modules	Implements different processing pipelines and operations.	Includes image, audio, and other processing components.

Appendix B. Selected Key Code Files in the Detection Module

Folder	File Name	Module	Primary Function
analysis	README	Documentation	Introduces the project and provides usage instructions.
	auc_fromaug.py	AUC Calculation and Visualization	Calculates and visualizes the effects of different methods on model performance using AUC.
	auc_table1_fromrecord.py	AUC Result Processing	Reads computed AUC values and summarizes detector performance across datasets in tabular form.
	auc_table2_fromrecord.py	AUC Result Processing	Organizes AUC results across training methods and test datasets and exports summary tables.
	curve_draw.py	ROC and Precision–Recall Curves	Generates ROC and precision–recall curves for model performance evaluation.

training	explore_pretrained.py	Pretrained Model Analysis	Evaluates pretrained models across datasets by comparing AUC performance.
	frequency.py	Frequency Spectrum Analysis	Analyzes and visualizes frequency-spectrum differences between real and manipulated images.
	heatmap_tab2.py	Heatmap Visualization	Generates heatmaps comparing model performance across multiple datasets.
	model_archi.py	Model Architecture Comparison	Visualizes and compares model architectures using radar charts.
	number_frames.py	Frame Number Analysis	Examines the relationship between the number of training frames and model performance.
	plot_curve.py	Performance Curve Visualization	Plots performance curves for different detection models.
	tsne.py	t-SNE Visualization	Projects high-dimensional features into low-dimensional space for visualization.
	config/	Configuration	Stores configuration files, model settings, and hyperparameters.
	dataset/	Dataset Management	Provides dataset storage, preprocessing, loading, and access functions.
	detectors/	Detector Models	Contains deepfake detection models for identifying manipulated images.
	lib/	Library	Provides reusable utility libraries and helper functions.
	loss/	Loss Functions	Implements loss functions used for model optimization.
	metrics/	Evaluation Metrics	Implements evaluation metrics such as AUC, accuracy, and precision.
	networks/	Neural Network Architectures	Defines neural network architectures for deepfake detection.
	optimizor/	Optimizer	Implements optimization algorithms such as SGD and Adam.
	pretrained/	Pretrained Models	Stores pretrained models for fine-tuning or direct inference.
	trainer/	Training Module	Implements training workflows, including data loading and parameter updates.
	logger/	Logging	Provides logging utilities for monitoring and debugging training.
	test/	Test Dataset	Stores datasets used for model evaluation.
	train/	Training Dataset	Stores datasets used for model training.

Supplementary Files

This is a list of supplementary files associated with this preprint. Click to download.

- SupplementaryInformationCodeCorpusAnalyticalProtocolandEvidenceAuditTrail.docx