MASARYK
UNIVERSITY

FACULTY OF INFORMATICS

An Approach to Digital Image Forgery Detection

Master's Thesis

ZUZANA PITSMAUSOVÁ

Brno, Fall 2025

MASARYK UNIVERSITY

FACULTY OF INFORMATICS

An Approach to Digital Image Forgery Detection

Master's Thesis

ZUZANA PITSMAUSOVÁ

Advisor: doc. RNDr. David Svoboda, Ph.D.

Department of Visual Computing

Brno, Fall 2025


FACULTAS ARTIS INFORMATICÆ
UNIVERSITAS MASARYKIANA

Declaration

Hereby I declare that this paper is my original authorial work, which I have worked out on my own. All sources, references, and literature used or excerpted during elaboration of this work are properly cited and listed in complete reference to the due source.

During the preparation of this thesis, I used the following AI tools:

- Grammarly for grammar checking.
- ChatGPT to improve the writing style and to assist with the preparation of LaTeX figure and table placeholders. During the development of the practical part of the thesis, it was used for code checking and troubleshooting, for discussing possible adjustments and improvements to the implementation, and for assistance with code documentation, explanatory text in the accompanying Jupyter Notebook, and README files.

I declare that I used these tools in accordance with the principles of academic integrity. I checked the content and took full responsibility for it.

Zuzana Pitsmausová

Advisor: doc. RNDr. David Svoboda, Ph.D.

Acknowledgements

I would like to express my sincere gratitude to my advisor, doc. RNDr. David Svoboda, Ph.D., for his guidance, feedback, valuable insights, and continuous support throughout this thesis. His patience, reliability, and the significant amount of time invested in consultations greatly contributed to shaping this work.

I am also grateful to Ing. Martin Spurný and doc. RNDr. Petr Matula, Ph.D., for taking the time to discuss the thesis and for sharing their perspectives.

Additionally, I would like to thank Jáchym Coufal for his contribution to the image splicing forgery community through his thesis.

Lastly, I would like to thank my family and friends, whose encouragement and understanding have meant a great deal to me during my academic journey.

Abstract

Image splicing is a common and challenging form of digital manipulation in which content from multiple sources is combined into a single frame. The ability to detect such tampering is important in fields like journalism, law, and scientific research, where manipulated material can mislead or influence decision-making, as well as on social media, where altered content spreads quickly and can affect how information is perceived.

This thesis explores pixel-level image splicing detection using deep learning. A lightweight hierarchical transformer-based framework is designed and developed, with the architecture and training strategy tailored to better address the characteristics of image manipulation localization. The method aims to guide the model toward subtle forensic cues rather than semantic scene content, addressing a common limitation of many deep learning approaches. Additionally, transparency and reproducibility are emphasized by providing full implementation details and experimental settings.

The proposed method is evaluated on publicly available datasets through qualitative and quantitative analysis. The results demonstrate effective localization of spliced regions and promising generalization to images outside the training domain, while maintaining a significantly simpler architecture compared to a subset of recent multi-stream or otherwise complex solutions.

Keywords

Forgery Detection, Image Splicing, Deep Learning, Image Forensics, Pixel-Level Detection, Transformer-Based Architecture

Contents

Introduction **1**

1 Related Work and Background **3**
1.1 Image Forgery Detection 3
1.2 Image Splicing Forgery Detection 4
1.3 Hand-Crafted Features-Based Methods 9
1.4 Deep Learning-Based Methods 12

2 Datasets **14**
2.1 Utilized Datasets 14
2.2 Data Preparation 18

3 Proposed Method **21**
3.1 Overall Workflow 21
3.2 Design Motivation 22
3.3 Model Architecture 24
3.4 Training Details 27
3.5 Model Inference 33
3.6 Postprocessing 33
3.7 Implementation and Environment 34

4 Evaluation and Results **37**
4.1 Evaluation Metrics 38
4.2 Quantitative Evaluation 40
4.3 Qualitative Evaluation 46

5 Conclusion and Future Work **52**
5.1 Future Work 53

Bibliography **55**

A An appendix **64**

Introduction

Digital images have become an important part of how we communicate, document events, and share information. Ensuring their authenticity is particularly important in fields such as journalism, law, and scientific research, as manipulated visuals can mislead and influence decision-making. The same problem appears on social media, where altered content spreads quickly and may shape opinions long before anyone questions its credibility.

Unfortunately, as image editing technologies improve, so does the ability to alter images in ways that are often invisible to the human eye. Splicing, the act of combining content from different images into a single frame, is a widely studied and challenging form of tampering. Although not all image manipulations are malicious, developing reliable tampering detection methods and advancing related research represent crucial and highly practical challenges within digital forensics.

Historically, splicing research focused mainly on image-level detection, deciding whether an image has been manipulated. Early approaches relied on hand-crafted forensic features designed to capture visual inconsistencies introduced by tampering. However, these methods struggle to generalize, since they only perform well when the manipulation produces the specific artifacts they were designed to detect.

Over time, research shifted toward deep learning-based approaches. Instead of manually defining which traces to look for, neural networks learn relevant patterns directly from data. With deep learning, pixel-level detection (localization) has become increasingly common. Instead of providing only a binary decision, the model predicts a pixel-wise manipulation mask that indicates where the splicing occurred. However, even though deep learning removes the need for hand-crafted features, it introduces new challenges. Deep models can unintentionally focus on image content rather than subtle forensic traces. This limitation motivates the methods to actively guide the network toward manipulation-specific cues.

This thesis aims to explore the detection of image forgery using deep learning techniques. To achieve this, a deep learning framework

for pixel-level splicing detection is designed and developed. The architecture and training strategy are adapted to the specific characteristics of image manipulation localization, attempting to guide the model toward subtle forensic cues instead of semantic content. The work also places strong emphasis on transparency and reproducibility, addressing a common issue in the field where many methods lack implementation details or a clear specification of experimental settings.

The rest of the thesis is organized as follows. **Chapter 1** reviews related work on image forgery detection, focusing on splicing detection and both hand-crafted and deep learning-based methods. It also discusses publicly available image datasets relevant to this field. **Chapter 2** describes the datasets used throughout the development and evaluation, along with the data preparation process. **Chapter 3** presents the proposed method, including the design motivation, model architecture, training procedure, inference strategy, postprocessing steps, and implementation details. The experimental results — quantitative and qualitative — are then provided in **Chapter 4**. Finally, **Chapter 5** concludes the thesis and outlines possible directions for future work.

1 Related Work and Background

Published in 2021, **Advances in Computing and Data Science** [1] presents post-conference proceedings from the 5th International Conference on Advances in Computing and Data Science. One of its contributions, **Image Splicing Forgery Detection Techniques: A Review** [2], analyzes recent advancements in image splicing forgery detection, offering insight into both theoretical foundations and state-of-the-art methods. This chapter builds upon that review and uses it as a reference to provide essential context.

Within the image splicing literature, the term detection is often used in a broad sense. It may refer either to determining whether an image has been manipulated (image-level detection) or to identifying where the manipulation occurred (pixel-level detection, i.e., localization). Although this thesis focuses on localization, much of the traditional splicing research targets image-level detection. However, understanding these works remains highly relevant, as they reveal which intrinsic image characteristics are altered by splicing. Localization relies on the same forensic cues, but uses them to identify the manipulated regions. In this sense, it is a natural extension of image-level detection, shifting from a binary verdict to a pixel-wise prediction. Ultimately, both tasks are grounded in the same forensic principles.

1.1 Image Forgery Detection

As addressed in the Introduction, altering digital images is now easier than ever. As a result, detecting these changes has naturally become much more difficult, especially for the naked eye.

In the field of image forensics, tamper detection methods are classified into two categories: active and passive. **Active forensic methods**, also known as non-blind approaches, utilize the information embedded in the image to determine the possibility of forgery [2]. These approaches rely on pre-inserted data, such as a digital watermark or a digital signature, that are added to the image before its distribution either by the device used for acquisition or by an authorized individual processing the image [3]. That being said, most consumer-

grade digital cameras do not have the ability to embed watermarks or signatures at the time of image capture. Although some specialized cameras used in professional or secure settings may offer this feature, such cameras are not typically found in everyday use. Consequently, active methods are often impractical for pictures obtained from the Internet or social media platforms, where such embedded information is usually missing [2].

To overcome the limitations of active techniques, **passive forensic methods** (blind approaches) have been developed. These techniques require neither prior knowledge of the query image nor a particular key for verification [4]. Given that only about 2–5% of the images are actively secured with such keys [4], passive methods are particularly valuable for examining the vast majority of query images.

Passive approaches focus on analyzing the intrinsic characteristics of images, including statistical or structural features, which are often altered during tampering. Natural, unaltered images tend to exhibit consistent patterns, such as uniform noise distribution, well-defined edges, or coherent illumination across the scene. When an image is tampered with, these patterns may be disrupted, leading to inconsistencies in different regions. These irregularities can serve as clues for detecting forgery [2].

According to Meena and Tyagi [2], the passive image forgery detection techniques can be broadly classified into two categories: computer-generated image detection and image tampering detection. Image tampering detection includes several subcategories: copy-move forgery detection, retouching detection, resampling detection, and, lastly, splicing detection. The last category, splicing detection, will be discussed in more detail in the following section. The complete categorization of passive image forgery detection techniques, as outlined in Meena and Tyagi's review [2], is presented in Figure 1.2. However, it is worth noting that the categorization may vary across articles and reviews, reflecting diverse perspectives in the field.

1.2 Image Splicing Forgery Detection

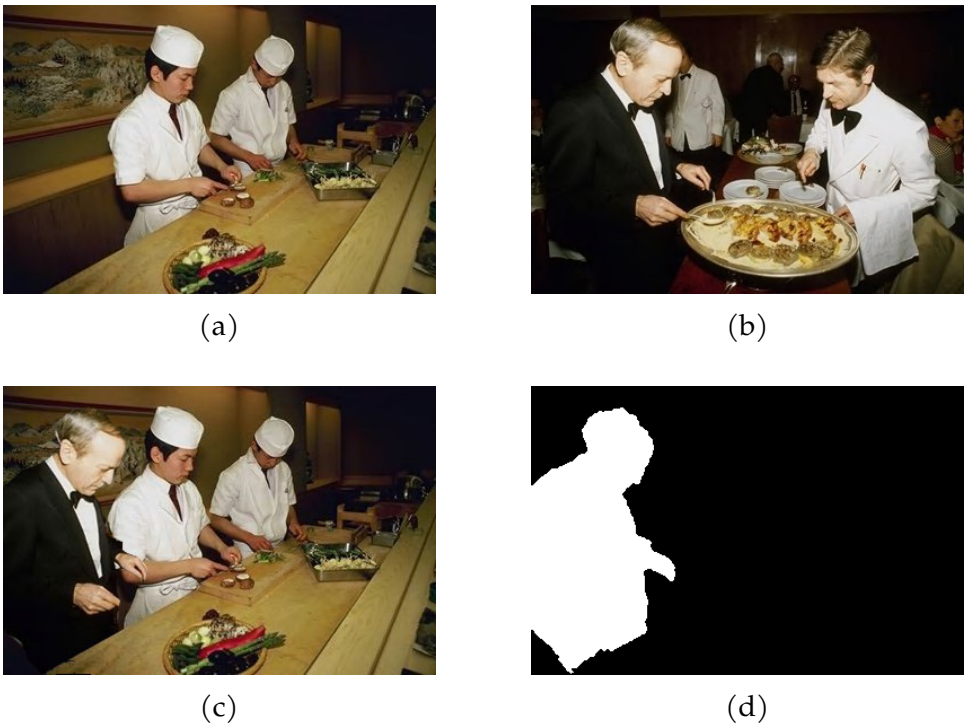
Image tampering can be viewed from two perspectives: legitimate and illegitimate [2].

Legitimate image tampering involves modifications that do not significantly change the semantic meaning of the image. For instance, retouching and resampling used for aesthetic adjustments without the intent of misrepresenting the image's content fall into this category. On the other hand, **illegitimate image tampering** involves intentional alterations that purposely change the meaning of the image, creating a false or misleading impression of the original scene. Examples of such techniques would be image splicing and copy-move forgery[1] [2].

Image splicing is the process of combining two or more different images (or parts of the images) to create a composite one [5]. It involves copying a portion of one image and pasting it into another image, as illustrated in Figure 1.1. The technique is commonly used to remove or replace specific objects or elements while maintaining the overall appearance intact [4]. The aim is to produce an image that appears authentic but alters the original content [4]. To create a convincing result, the forger might manipulate the source images by resizing, cropping, rotating, or retouching them to align with the visual context, proportions, or lighting of the host image [2]. Furthermore, the splicing operation is most of the time followed by various forms of postprocessing, such as average filtering or color adjustment, to ensure the spliced regions blend seamlessly with the host image [2]. These postprocessing operations, together with the fact that new information is inserted into a host image (due to the pasting of new region/s), introduce inconsistencies in the forged image, providing clues that can be used to expose that the image was spliced [2]. According to Itier, V. et al. [6], inconsistencies in tampered images can occur at various levels. These include coding-level artifacts (such as compression mismatches), macro-level issues (like content inconsistencies), and micro-level irregularities (such as variations in image noise).

A generalized workflow [2] to detect image splicing forgery comprises four main steps, two of which are optional. For cases focusing solely on classification, the input to such a workflow is a query image, and the output is typically a label indicating whether the image is spliced or not. For methods aiming at localization, the input is also a

1. Copy-move forgery refers to the manipulation of an image where a selected portion is copied and reinserted into another area within the same image [4]



(a) (b)

(c) (d)

Figure 1.1: An instance from the CASIA 2.0 dataset [7, 8]: (a) Host image serving as the base for splicing, (b) source image supplying the region pasted onto host image (a), (c) resulting spliced image, (d) ground truth mask (white = spliced, black = authentic).

query image. However, the output is usually a forgery detection map, highlighting specific tampered regions. The four steps involved are:

1. preprocessing (optional)
2. feature extraction
3. decision stage
4. postprocessing (optional)

The **preprocessing** step, while optional, is often the first step for many image splicing detection methods. When applied, it usually involves converting the input RGB image into an alternative color

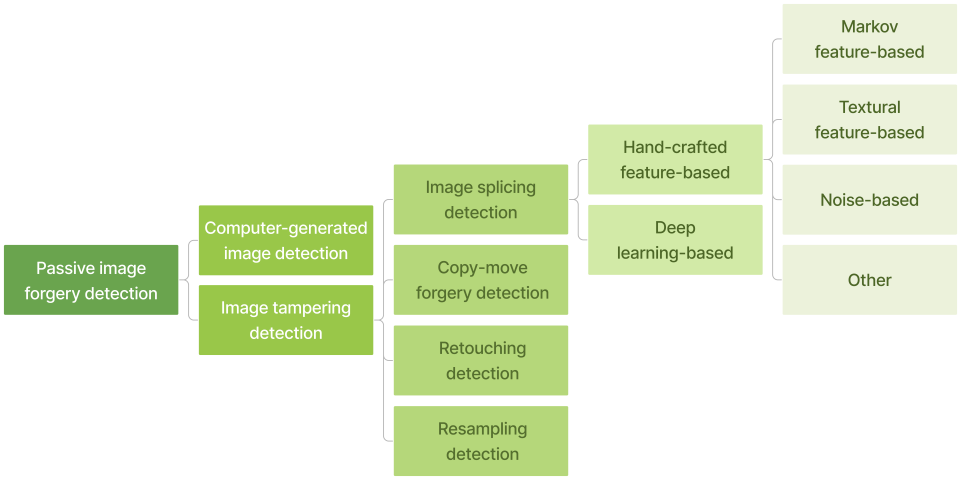


Figure 1.2: Division of passive image forgery detection approaches based on Meena and Tyagi's review [2].

space [2]. Among these, the YCbCr[2] color space has been shown to be particularly effective for a large number of methods [9, 10, 11]. Alahmadi, A. et al. [9] state that human vision perceives luminance more effectively than chrominance, and that is why many tampering traces, which are not detectable to the naked eye, are hidden within the chromatic channel. Although less common, some methods also benefit from converting to HSV [12] or grayscale [13, 14]. This step has the potential to enhance the performance, as some splicing artifacts may be more distinguishable in certain color models or channels [15]. Preprocessing may also include tasks such as resizing, denoising, or enhancing image quality [4].

Feature Extraction is intuitively a very important step in the image splicing forgery detection pipeline, as it enables the identification of discrepancies introduced during the splicing process. Depending on the type of extracted features, different types of inconsistencies caused by the splicing can be effectively highlighted [2]. A wide range of features is commonly utilized for this purpose, including patterns of

2. YCbCr is a color model that separates color information into three components: Y (luminance), representing brightness, and Cb and Cr (chrominance channels), which represent color differences — blue-difference and red-difference, respectively.

photo-response non-uniformity (PRNU) noise, chroma values, texture descriptors, block artifacts, noise statistics, and histogram properties [4]. In different methods, the feature extraction step is applied across various domains, such as the spatial domain [16], the discrete cosine transform (DCT) domain [16, 17, 18], and the discrete wavelet transform (DWT) domain [17]. As illustrated in Figure 1.2, Meena and Tyagi [2] categorize image splicing forgery methods based on the feature extraction mechanism into hand-crafted feature-based methods and deep learning-based methods, both of which are discussed more thoroughly in the following two sections.

In traditional splicing forensic approaches, image-level detection is more common. After the feature extraction step, the resulting features are passed to a classifier during the **decision stage**. Among these methods, Support Vector Machines (SVMs) are the most widely used **feature classification** technique for determining whether an image has been spliced [5, 9, 18]. Other approaches, such as Random Forest [6] and K-Means Clustering [19, 20], are applied less often. Techniques like logistic regression [13] are then relatively rare in this context.

With the advent of deep learning, the pipeline shifts from engineered features + classifier to end-to-end modeling, where a single network learns both feature extraction and the decision process. While many deep learning methods still perform image-level decision [21], more recent works increasingly aim for pixel-level detection, producing a manipulation mask on the output [22, 23].

Postprocessing is then the optional final step, typically applied to refine the output of the decision stage. Depending on the method, this may involve highlighting suspicious areas (in image-level detection) or improving the quality of a predicted manipulation mask (in pixel-level localization). Common techniques include filtering, smoothing, and morphological refinement to reduce false positives or negatives and improve overall accuracy [2, 4].

With the general workflow of image splicing detection outlined, the following two sections focus on principles and methods employed in this field, covering both hand-crafted feature-based approaches and deep learning-based techniques.

1.3 Hand-Crafted Features-Based Methods

Hand-crafted feature-based methods represent a traditional approach to image splicing forgery detection. These methods rely on manually designed features rather than learned representations [2, 4]. As discussed in Section 1.2, the majority of the hand-crafted approaches focus on image-level detection [9, 15, 18], whereas only a smaller subset performs localization [10, 19].

Figure 1.2 illustrates the classification of hand-crafted methods into four categories [2]:

- Markov feature-based
- Textural feature-based
- Noise-based
- Other

This section aims to introduce these categories, outlining the underlying principles behind each one and providing examples of related state-of-the-art methods.

Markov Feature-Based Methods

When an image undergoes splicing, the correlation between its pixels tends to change, which can be modeled using the Markov random process, a concept rooted in random process theory. As a result, Markov-based features, derived from this process, can capture the statistical changes introduced by splicing [24]. One of the key components in deriving these features is the transition probability matrix, thanks to its ability to effectively characterize the correlation between pixels or coefficients[3] [17]. The Markov features are frequently generated from the transition probability matrices in DCT [17, 18], occasionally also in the DWT [17] domain.

A typical workflow for extracting Markov features involves several steps [17, 25]: computing 2-D coefficient difference arrays (horizontal, vertical, diagonal, and minor diagonal), applying thresholding

3. Coefficients in this context refer to numerical values obtained after applying a transformation like DCT (usually to smaller 8x8 image blocks) or DWT.

to reduce the dimensionality of probability transition matrices, and generating transition probability matrices from the thresholded arrays. Different methods utilize various combinations of these difference arrays [18, 25]. The choice of the threshold is then particularly important, as it directly affects a tradeoff between computational complexity and performance [18].

A notable drawback of these approaches is that they typically generate a large number of features and thus need to be often followed by a dimensionality reduction technique [18]. Nevertheless, Markov features remain a popular choice for many state-of-the-art image splicing detection methods.

Textural Feature-Based Methods

During image splicing, structural changes occur in the affected regions, modifying the micro-texture patterns both within the spliced area and along its boundaries. These changes alter the local frequency distribution and introduce discontinuities, including sharp edges at the borders [9]. Such textural alterations and discontinuities are often leveraged in splicing detection methods that rely on various descriptors like Local Binary Patterns (LBP) [9], Gray Level Co-occurrence Matrix [14], or Local Directional Patterns [10].

Among these, LBP is particularly popular due to its simplicity and ability to capture and highlight tampering artifacts in manipulated images. By encoding micro-patterns such as edges into a binary format, LBP emphasizes textural inconsistencies caused by splicing, making the altered regions more pronounced [9].

Noise-Based Methods

Noise-based methods are widely explored in the field of splicing detection [6, 19, 26]. They are motivated by the assumption that spliced regions originate from a different source and, therefore, tend to exhibit distinct noise patterns compared to the authentic parts of the image [6, 19]. Noise is naturally introduced during the image acquisition process, and the inherent characteristics of each camera sensor lead to different noise variances across devices. In an unaltered image, noise is generally uniform. However, splicing disrupts this consistency, as the

spliced region typically reflects the noise characteristics of its original source [20]. Additionally, forgers may deliberately introduce noise to hide manipulation traces [19].

Typically, these methods follow a two-step process. First, noise variances are estimated across different regions of the image to identify potential inconsistencies. Next, clustering techniques are applied to distinguish spliced regions from authentic ones by analyzing the detected noise variance discrepancies [20]. The effectiveness of noise-based approaches, however, depends on the robustness of the noise level estimator [2]. To enhance accuracy, various noise level estimators have been explored [6, 19].

Despite these advancements, noise-based approaches face challenges in certain scenarios. Zhu, N. and Li, Z. [20] point out that regions with complex textures or sharp edges can lead to inaccurate noise variance estimations, as local blind noise estimation methods often struggle in such cases. As a result, this inaccuracy can reduce the effectiveness of subsequent splicing detection steps. Furthermore, the performance of noise-based techniques is often unsatisfactory when the noise differences between the host image and spliced regions are subtle [19].

Other Methods

In addition to texture-based, noise-based, and Markov feature-based methods, Meena and Tyagi [2] introduce a category for approaches whose features do not neatly fit into the previously mentioned groups [15, 27]. This category also includes hybrid methods [13] that mainly combine multiple features, attempting to capture a wider range of artifacts and inconsistencies introduced by the splicing.

A notable subset of these methods [27, 28] is based on JPEG compression features, which were prominent in earlier splicing detection approaches [29, 30]. These methods exploit two primary concepts [27]: blocking artifacts and inconsistencies in DCT coefficients. Blocking artifacts are grid-like patterns that arise during JPEG compression due to the independent processing of non-overlapping 8x8 pixel blocks [27]. These artifacts can be used as a "natural authentication code" to verify image integrity. When images are manipulated, such as by splicing from different sources or cameras, the blocking artifacts may become

inconsistent, signaling potential tampering [29]. Inconsistencies in DCT coefficients, on the other hand, are often linked to the double quantization effect. They occur when a spliced region, compressed separately (with a different quantization factor), is inserted into the image, leading to disruptions in compression patterns and histogram periodicity [30]. These foundational concepts inspired many subsequent methods [27, 28], and they still remain relevant for image splicing forgery research.

1.4 Deep Learning-Based Methods

Deep learning has demonstrated remarkable success across a variety of computer vision tasks, such as object detection [31], semantic segmentation [32], and image classification [33], largely due to its strong capability to represent features in high-dimensional data [34]. It has naturally been adopted for image splicing detection to address the limitations of hand-crafted feature-based methods. Traditional approaches often rely on domain-specific features and rigid assumptions about spliced regions and tampering patterns, making them less adaptable to different manipulation scenarios [5, 34]. Deep learning approaches attempt to overcome these challenges by automatically learning robust feature representations from large datasets [4, 5].

Deep learning-based methods for image splicing detection employ a variety of strategies as outlined in comprehensive reviews [2, 4]. Early approaches used neural networks mainly as feature extractors, combining them with traditional classifiers like SVMs to detect tampering [5, 35]. Later, research shifted toward an end-to-end approach, where a single network handles both feature extraction and the decision stage within a unified framework [21, 22].

With the rise of end-to-end architectures, pixel-level localization has become increasingly common. Instead of producing only a binary decision, many modern approaches formulate splicing detection as a segmentation problem and directly output a manipulation mask [22, 36].

Deep learning-based approaches vary in complexity — while some rely on relatively straightforward architectures [37], others explore more sophisticated designs [21, 22]. These include U-Net [22] and

FCN-derived models [36] adapted for manipulation localization, and Vision Transformer-based models [23, 38].

A subset of methods enhances the input space through domain-specific preprocessing techniques inspired by traditional splicing detection methods. For instance, Meena, K., and Tyagi, V. [5] employed noise residual maps, Wu, J. et al. used high-pass filtering [37], and Pomari, T. et al. leveraged illuminant maps [35], each aiming to emphasize tampering traces before feeding the data into neural networks. For pixel-level detection, this idea is often extended through multistream architectures [39, 40], where the RGB image is usually processed in one branch and forensic cue maps are provided to additional branches.

Although convolutional neural networks (CNNs) are very powerful tools, their standard application often results in models that focus on learning scene content from training data rather than capturing manipulation traces essential for forensic analysis [37]. To address this limitation, researchers have explored strategies that suppress image content while enhancing splicing traces, typically through modifications to the architectures [22, 23] or preprocessing techniques [5, 37, 40].

Altogether, the various strategies discussed in this section demonstrate the field's ongoing effort to balance complexity, performance, and adaptability to diverse manipulation scenarios, as researchers continue to tackle many challenges posed by splicing detection.

2 Datasets

The availability of high-quality datasets plays a significant role in advancing image splicing research. This area, like many others, relies on large, annotated datasets that are essential for training and evaluating machine learning algorithms. Recent image forgery detection literature makes use of several widely adopted resources, including CASIA v1.0 and v2.0 [7, 8], CUISDE [41], CARVALHO [42], Fantastic Reality [43], IMD2020 [44], NIST16 [45], and DEFACTO [46]. In addition, a number of studies employ self-created datasets to address specific research needs [26].

While this thesis does not aim to provide an exhaustive enumeration of all forgery datasets, it is important to acknowledge their diversity and availability. Comprehensive reviews, such as those by Meena and Tyagi [2] or Kumari and Garg [4], offer more detailed listings. As highlighted by these reviews, every dataset has its strengths and weaknesses, and researchers should carefully select the one that best aligns with their research goals and requirements.

Since a neural network can only learn manipulation cues present in the data it is exposed to, the quality and diversity of datasets play a crucial role in the development and evaluation of effective splicing detection approaches.

This chapter provides an overview of the datasets used throughout the training, validation, and testing stages of the proposed method. The first section introduces individual datasets, outlines their characteristics, and discusses their limitations and relevance to the splicing detection task. The second section then describes the data preparation steps applied to the datasets.

2.1 Utilized Datasets

In the development and evaluation of the proposed splicing detection method, multiple publicly available datasets were used, each offering distinct characteristics in terms of scene diversity, resolution, or multiplicity of spliced regions. Table 2.1 provides an overview of these datasets, indicating whether they were used throughout the training

pipeline (training, validation, and testing) or exclusively for testing the generalization capabilities of the model.

CARVALHO

The **CARVALHO** [42] dataset, also known as DSO-1, consists of 200 high-resolution images ($2,048 \times 1,536$ pixels), split evenly into 100 authentic and 100 spliced samples. The forgeries involve inserting one or more people from one image into another, followed by postprocessing to blend the tampered regions seamlessly. Adjustments to color, brightness, and contrast make the manipulation visually convincing and challenging to detect.

CASIA v1.0 and v2.0

CASIA v1.0 [7, 8] and CASIA v2.0 [7, 8] are among the most frequently used datasets in image splicing detection [18, 22]. While both also include tampered images produced via copy-move forgery, only the splicing subset is used. Neither of these two datasets includes official ground truth masks for the manipulated regions. Therefore, this work relies on binary masks provided by a third-party source [8]. Additionally, it is important to mention that many CASIA v1.0 and v2.0 samples exhibit visible compression artifacts, particularly block-like patterns.

CASIA v1.0 includes 800 authentic and 462 spliced `.jpg` color images, all at a resolution of 384×256 pixels and organized into eight thematic categories [1]. The splicing manipulations involve pasted regions that may be rotated, scaled, or otherwise distorted, without any postprocessing. The spliced regions vary in size and complexity.

CASIA v2.0 is an extended and more challenging version of v1.0, comprising 7,491 authentic and 1,828 spliced images in `.jpg`, `.bmp`, and `.tif` formats, with resolutions ranging from 320×240 to 800×600. Unlike in v1.0, postprocessing (e.g., blurring) may be applied. A ninth category, "indoor," was added to increase content diversity.

1. scene, animal, architecture, character, plant, article, nature, and texture

CUISDE

The Columbia Uncompressed Image Splicing Detection Evaluation (**CUISDE**) [41] dataset is a commonly used benchmark in the image forensics community, particularly for assessing cross-dataset generalization performance. It contains only splicing manipulations. The dataset comprises 183 authentic and 180 tampered high-resolution uncompressed images in `.tiff` and `.bmp` formats with resolutions ranging from 757×568 to $1,152 \times 768$ pixels. Most images depict indoor scenes, and approximately 15% show outdoor environments. The authentic images were captured using four different digital cameras, and each spliced image was created from the authentic image set by combining content from exactly two cameras. No postprocessing was applied to the manipulated images. Each tampered image is accompanied by two manually created ground truth masks, which mark the spliced regions using color-coded annotations. The specific usage of these masks in this thesis is described in Section 2.2.

DIS25k

The **DIS25k** [47] dataset contains nearly 25,000 automatically generated spliced images accompanied by corresponding ground-truth masks. It was introduced to demonstrate the capabilities of the authors' framework and offers high artifact diversity, a large sample size, and realistic manipulations.

Fantastic Reality

The **Fantastic Reality** [43] dataset is a large-scale collection containing exclusively splicing forgeries. It includes over 16,000 authentic and 19,000 manipulated images, primarily in JPEG format, each accompanied by a corresponding ground-truth mask. The dataset covers a wide variety of scenes and content.

IMD2020

The **IMD2020** [44] is a dataset of 2,010 manipulated images collected from online sources and designed to represent challenging, real-world

forgeries. It covers several types of forgeries, including splicing, removal, and copy-move. In this work, only the subset of samples involving splicing manipulations was manually selected and used. Ground-truth masks are available for all samples.

In-the-Wild

The **In-the-wild** [48] dataset contains 201 real-world splicing forgeries collected from two online sources: The Onion (parody news website) and the Reddit Photoshop Battles community, where users create and share manipulated images. The manipulations are typically well blended and often involve people, faces, or animals. Ground-truth masks are provided, even though in some cases they are rather coarse.

SP COCO

Dataset **SP COCO** [40, 49] was generated from the COCO 2017 dataset [50] by automatically selecting one or more arbitrary objects from a source image and pasting them onto a different host image at random positions, with random resizing and rotation. The resulting images are JPEG-compressed with randomly selected quality factors between 60 and 100, without any additional postprocessing. The dataset provides a large number of samples, comprising 199,999 tampered images, each accompanied by its corresponding ground truth mask.

MISD

The authors describe the **Multiple Image Splicing Dataset** (**MISD**) [51] as the first publicly available dataset containing samples with realistic multiple spliced regions. It comprises 620 authentic images (sourced from CASIA v1.0) and 303 tampered images, primarily in `.jpg` format, each with a fixed resolution of 384×256 pixels. Spliced regions are extracted from one or more authentic images and modified through transformations such as rotation or scaling. Each manipulated image is accompanied by separate binary ground truth masks, with their number corresponding to the number of spliced regions. However, the masks in this dataset are less precise, and in most cases,



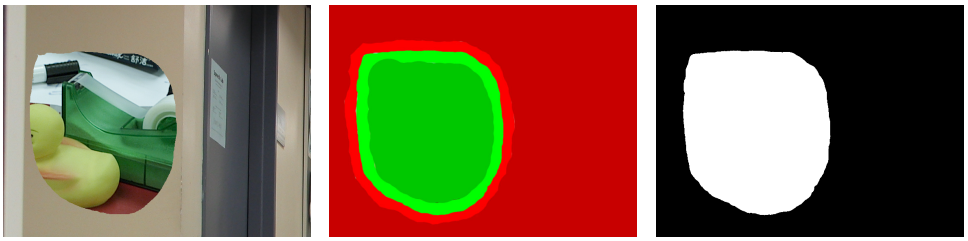
Figure 2.1: Sample from the MISD [51] dataset and the corresponding ground-truth mask (white denotes the spliced region, black the authentic background).

the spliced regions include pixels that do not actually belong to the manipulated area, as illustrated in Figure 2.1. The dataset is rather small, and the image resolution is also, as in CASIA v1.0 and v2.0, lower.

2.2 Data Preparation

All utilized datasets were carefully inspected to ensure consistency and alignment between the tampered images and their corresponding masks. Since CASIA v1.0 and CASIA v2.0 contain both splicing and copy-move manipulations, only samples related to the splicing task were extracted. Similarly, IMD2020 includes several manipulation types, and only the splicing cases were selected for use in this work.

Each dataset was verified to ensure that every tampered image had a corresponding mask and vice versa. While most datasets passed this integrity check, a small number of samples from CASIA v2.0, CASIA v1.0, and CARVALHO exhibited mismatches in image and mask dimensions. These samples were excluded from further processing. Additionally, two tampered images in CASIA v2.0 lacked masks entirely and were therefore removed. Spatial alignment between images and masks was also examined. In a small number of CASIA v2.0 samples, mask rotation errors were identified. In these cases, corresponding input images were rotated accordingly. Visual in-



(a) (b) (c)

Figure 2.2: (**a**) Image sample from the CUISDE dataset (**b**) Four-region ground-truth mask provided by the dataset creators (**c**) Binarized mask used in this thesis (green regions mapped to 1, others to 0).

spection was then performed to confirm that masks correctly highlight manipulated regions.

Additionally, ground-truth masks were available only for manipulated images. Therefore, binary masks filled with zeros (indicating no manipulation) were generated for authentic images, matching the original dimensions of the corresponding inputs.

In the case of the MISD dataset, each tampered image is accompanied by multiple masks, one per spliced region. These were merged into a single binary ground truth mask per image to ensure consistency with the format used in other datasets.

In the CUISDE dataset, each tampered image is provided with two ground truth masks: one containing four color-coded regions (dark red, bright red, dark green, and bright green), and another in which the bright red and bright green regions are merged into a single blue region. For consistency, the four-region masks were used in this work, where the bright green and dark green areas were assigned a value of one (spliced region), and the remaining pixels were assigned a value of zero (authentic background), as illustrated in Figure 2.2.

As shown in Table 2.1, a total of four datasets (CASIA v2.0, SP COCO, Fantastic Reality and IMD2020) were employed for training, validation, and testing, while six additional datasets (MISD, DIS25k, In-the-Wild, CUISDE, CARVALHO, and CASIA v1.0) were reserved solely for evaluation purposes. The training datasets were first divided into training and testing subsets using an 80:20 split. Subsequently,

Table 2.1: Overview of datasets used in the development and evaluation of the proposed method, including their respective splits for training, validation, and testing. For each split, the numbers are reported in the order tampered / authentic. A dash (–) indicates that the dataset was not used for that purpose.

Dataset	Train	Validation	Test	Usage Notes
CASIA v2.0 [7, 8]	1,262 / 631	146 / 73	366 / 370	Used in pipeline
Fantastic Reality [43]	2,524 / 631	292 / 73	3,885 / 1,488	Used in pipeline
IMD2020 [44]	1,262 / 0	146 / 0	337 / 0	Used in pipeline
SP COCO [40, 49]	3,786 / 0	438 / 0	10k / 1,488	Used in pipeline
CARVALHO [42]	–	–	95 / 100	Evaluation only
CASIA v1.0 [7, 8]	–	–	461 / 800	Evaluation only
CUISDE [41]	–	–	180 / 183	Evaluation only
DIS25k [47]	–	–	10k / 0	Evaluation only
In-the-Wild [48]	–	–	201 / 0	Evaluation only
MISD [51]	–	–	295 / 0	Evaluation only

10% of the training portion was held out for validation. These splits were applied to both authentic and spliced images in CASIA v2.0 and Fantastic Reality, and only to the tampered cases in SP COCO and IMD2020.

For the datasets used solely for evaluation, various amounts of available samples were employed, including both authentic and tampered images from CUISDE, CARVALHO, and CASIA v1.0, and only tampered images from MISD, DIS25k, and In-the-Wild. The total number of samples used from each dataset at each stage of the pipeline is summarized in Table 2.1.

3 Proposed Method

This chapter introduces the proposed deep learning-based approach to image splicing detection, designed to localize manipulated regions in digital RGB images using supervised learning.

The chapter first outlines the overall workflow of the proposed approach, followed by the design motivation and a detailed description of the model architecture. It then provides a comprehensive overview of the training procedure, including the employed loss functions, supervision strategy, and optimization settings. Subsequently, the inference process and postprocessing applied to the model outputs are introduced. The chapter concludes with implementation specifics.

3.1 Overall Workflow

The overall workflow of the proposed method follows a structured pipeline that processes an input image through a sequence of stages, from preprocessing to visualization. Below is an overview of each stage:

1. **Input Preprocessing**
   Each input image is first normalized using the ImageNet channel-wise mean and standard deviation values to ensure compatibility with the pretrained backbone used in the model.
2. **Model Inference**
   The normalized image is processed using a sliding-window inference strategy with a patch size of 512×512 pixels and a stride of 384 pixels — the same configuration as used during validation. The model produces four outputs: three region logits at multiple scales and one edge-prediction logit map. The highest-resolution region output serves as the final prediction map. After applying a sigmoid activation, it represents the probability of splicing for each pixel.
3. **Postprocessing the Output**
   Instead of directly thresholding the sigmoided output, postprocessing techniques are applied to enhance prediction quality and improve the localization of spliced regions.

4. **Visualization**
   The final output includes a visualization of the input image, the model's forgery probability map indicating a probability of splicing at each pixel, and the prediction mask overlaid on the original input image. An example of the visualization result is provided on the CASIA 2.0 sample in Figure 3.1.



Figure 3.1: Example of the visualization output produced by the proposed method on a CASIA v2.0 sample.

3.2 Design Motivation

Semantic segmentation is a pixel-wise classification task in which each pixel in an image is assigned a class label. Given that the goal of this work is to detect and localize spliced regions, the proposed method naturally formulates the problem as a binary segmentation task, where each pixel is classified as either spliced (1) or non-spliced (0).

However, image manipulation localization (IML) differs from conventional semantic segmentation in several important aspects that need to be considered when designing the approach. While many segmentation networks are trained to recognize semantically meaningful objects and regions, IML aims to detect subtle, low-level inconsistencies between authentic and tampered areas. These manipulation cues are typically weak and unevenly distributed across the image and not necessarily aligned with object contours. As a result, models optimized for standard semantic segmentation tend to overlook such forensic evidence.

This limitation is closely related to a well-documented issue of content bias [5, 37] in deep learning-based image forensic approaches.

CNN-based architectures tend to prioritize semantic content over manipulation cues. While this is suitable for tasks such as classification or object detection, it becomes problematic in splicing detection, where the goal is to capture tampering artifacts. In this context, image content acts as a form of noise, overshadowing the subtle splicing traces [37]. Our experiments with conventional CNN-like segmentation architectures confirmed this tendency. Without additional constraints or guidance, the models tended to rely on object structure instead of splicing traces. Even when replacing raw RGB input with stacked forensic feature maps designed to suppress content bias and enhance manipulation cues, the models showed limited generalization across datasets.

Additionally, convolutional architectures primarily capture local context. In contrast, detecting splicing often requires comparing spatially distant regions. This inability to capture long-range relationships further limits CNNs in IML tasks.

Taken together, these insights motivated the development of the approach to image splicing detection presented in this thesis. To address the shortcomings of convolutional architectures, a transformer-based encoder is adopted to better capture relationships between distant regions. To further tailor the model to the specific characteristics of IML, three key design aspects, also discussed in earlier work [23], are incorporated, namely resolution preservation, boundary sensitivity, and multi-scale representation. The following paragraphs describe these aspects and explain why they are particularly important for IML.

Boundary Sensitivity

In IML, the most distinctive tampering cues often appear along the boundary where the pasted region meets the original image. These transition areas may reveal inconsistencies in texture or lighting, even when the interior of the spliced region does not show obvious signs of manipulation. Standard segmentation models mainly rely on information within a region to assign a class label, so boundary precision is not explicitly emphasized. In contrast, IML benefits from guiding the model to focus on these transition areas, as this improves its ability to detect and localize splicing.

Multi-Scale Representation

The size of manipulated areas varies considerably, from small object-level edits to large replacements that affect most of the image. Networks that operate at a single spatial scale may fail to detect small manipulations after downsampling or lose global context for large ones. Introducing multi-scale supervision encourages features to be learned at different spatial resolutions, improving robustness and helping the model generalize better across datasets with varying manipulation characteristics.

Resolution Preservation

CNN-based segmentation models typically reduce spatial resolution early in the encoder to obtain more abstract semantic features. While this is effective for object recognition, it causes subtle forensic cues to disappear in the first few layers. In IML, however, these subtle signals are precisely what the model should preserve. Excessive downsampling or resizing can wash away the artifacts before the model has the chance to learn from them. Preserving resolution in early feature extraction stages is therefore important to retain these cues.

In summary, the identified limitations of CNNs, together with the outlined principles, guided the development of the proposed approach and formed the basis for the architectural choices and training strategy described in the following two sections.

3.3 Model Architecture

The proposed model follows an encoder-decoder design composed of a tiny-sized **Swin Transformer V2** encoder for feature extraction and a lightweight Feature Pyramid decoder for reconstruction. A high-level overview of the architecture is presented in Figure 3.2.

Encoder

The **encoder** is based on a tiny-sized Swin Transformer V2 [52, 53], implemented via the `timm` library. Swin processes the input image

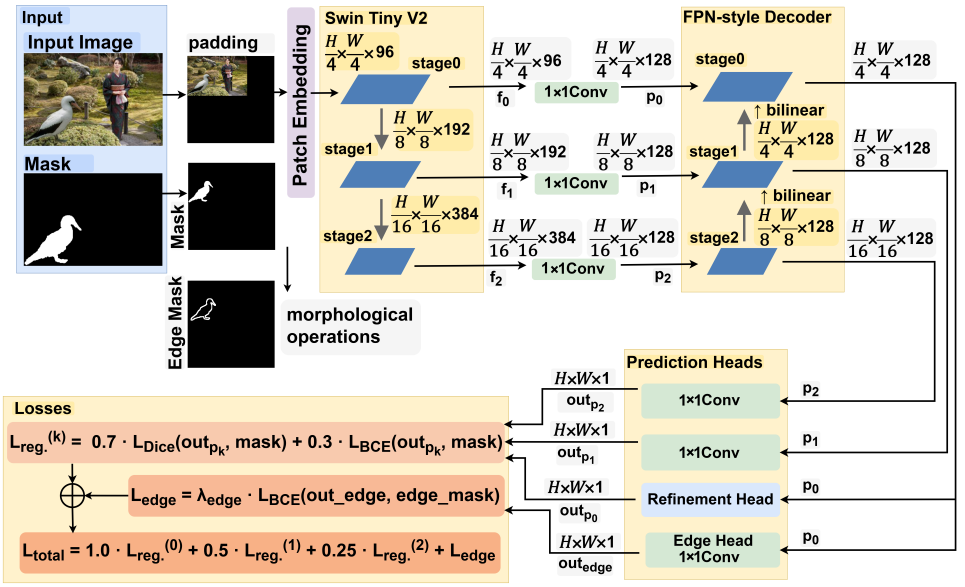


Figure 3.2: Overview of the proposed architecture. Losses are considered for a tampered image.

as a sequence of non-overlapping 4×4 pixel patches, allowing fine-grained forensic cues to enter the network without being excessively averaged or blurred.

Unlike standard Vision Transformers that compute attention globally, Swin applies windowed self-attention within 8×8 windows, which significantly reduces computational cost. The windows shift between layers, allowing features from neighboring windows to interact. This helps the model preserve fine-grained local cues while still building a global understanding of the scene, making Swin well-suited for image splicing localization.

The encoder is initialized with ImageNet-1k [54] pretrained weights, as prior forensic research has shown that transfer learning accelerates convergence and improves generalization.

Instead of using the full Swin, only the first three hierarchical stages are retained. This limits the spatial downsampling to $1/16$ of the input resolution, whereas the full Swin would reduce it to $1/32$. Preserving feature maps at a higher spatial resolution prevents subtle splicing traces from disappearing during encoding, while still allowing

hierarchical feature abstraction. Each stage outputs a feature map at a different resolution, forming a multi-scale representation used later by the decoder.

Decoder and Output Heads

The feature maps extracted by the encoder are fed into a top-down **decoder** inspired by Feature Pyramid Networks (FPN). Each feature map is first projected into a shared 128-channel representation using a 1×1 convolution. Feature fusion then proceeds in a coarse-to-fine manner: the deeper feature map is upsampled and added to the feature map from the previous stage. This process is repeated until the highest-resolution feature map is reconstructed. In this way, the decoder enriches fine-resolution features with contextual information from deeper layers.

The decoder produces segmentation outputs at three different scales. Each scale has its own prediction head and generates a mask from the features available at that stage. This multi-scale prediction setup helps the network handle manipulated regions of different sizes. Coarse heads detect large structural inconsistencies, while finer heads preserve subtle traces and boundary detail.

After the multi-scale predictions are generated, the highest-resolution output is passed through a dedicated refinement block. This module applies sequential convolutions, normalization layers, and a dilated convolution to enlarge the receptive field without reducing the spatial detail. The refinement stage aims to add broader context at the final scale, with the normalization layers helping to keep the feature values consistent across the refinement steps before the model produces the output mask.

To focus on forensic evidence near the splice boundaries, the decoder includes a dedicated boundary prediction head attached to the highest-scale feature map. The head comprises a single 1×1 convolution that outputs a boundary-likelihood map.

In conclusion, these architectural choices are made with the specific goal of adapting the model to the image splicing localization task. The next section presents the training strategy that complements this design.

3.4 Training Details

The model is trained to perform pixel-wise classification using a combination of region supervision and explicit boundary supervision. The training pipeline follows the design principles introduced in Section 3.2, ensuring that the network learns to detect manipulation artifacts rather than semantic content.

The following paragraphs describe the main components of the training setup, including the augmentations, data handling, loss functions, supervision strategy, optimization scheme, and hyperparameters.

Data Handling

To efficiently manage training and validation samples, a custom PyTorch Dataset class is implemented. Each dataset instance stores tuples of file paths pointing to an RGB input image and its corresponding ground-truth mask. Masks are stored as 8-bit images and therefore divided by 255 when loaded to obtain a binary representation $\{0, 1\}$, which is required by the loss functions. After loading and applying any necessary spatial adjustments (padding or cropping), both the image and the mask are converted into tensors. For the training set, augmentations are applied before this conversion.

Because samples vary in spatial resolution, each training image-mask pair is standardized to 512×512 pixels.

- If a sample is smaller than this resolution, it is padded — the image with 0 and the mask with -1, enabling padded pixels to be ignored during loss computation.
- If the sample is larger in any dimension, a random 512×512 crop is applied.

Padding places the original content in the top-left corner, keeping the padded regions spatially consistent and helping the model to learn to ignore them.

To prevent large datasets such as SP COCO from dominating the learning process, a balanced sampling strategy is adopted across the four training sources. Since IMD2020 is the smallest dataset, it serves

as the reference size. For each training epoch, all available IMD2020 samples are included. The subsets from the remaining datasets are selected according to fixed sampling factors: an equally sized subset from CASIA v2.0, a $2\times$ subset from Fantastic Reality, and a $3\times$ subset from SP COCO. Authentic images are sampled in equal proportion from CASIA v2.0 and Fantastic Reality, with the total number matching the size of the IMD2020 subset. This strategy ensures that every batch contains samples from all sources, reducing dataset bias during training.

A similar sampling scheme is applied to the validation data. The IMD2020 validation split again serves as the reference size, and the remaining subsets are selected using the same sampling factors as in training. The number of samples from each dataset used for training and validation can be found in Table 2.1.

For training, five independent instances of the Dataset are created: four corresponding to the tampered parts of the datasets and one for the authentic subset. Each instance has its own DataLoader, sampling according to the factors described above. During training, batches from all loaders are concatenated and shuffled, resulting in a final batch size of 8 per iteration. This setup ensures that each update sees data from all datasets.

For validation, all selected samples are merged into one Dataset and shuffled once before use. A single DataLoader is then used to iterate over this unified set with a batch size of 1.

During training, images are augmented on-the-fly using the augmentation pipeline described below, followed by normalization by ImageNet mean and standard deviation. For validation, only ImageNet normalization is applied.

Data Augmentation

To improve robustness and limit overfitting, augmentations are applied during training on the training set. The augmentation pipeline contains two groups:

1. **Spatial augmentations**, such as horizontal and vertical flipping, to increase geometric variability.

2. **Photometric and compression augmentations**, such as color jitter, gamma adjustment, Gaussian noise and blur, and JPEG compression, which simulate postprocessing artifacts introduced by image editing software, camera pipelines, or repeated saving or uploading.

Although these augmentations may weaken some forensic traces, this is intentional — manipulated images in practice often undergo additional edits or compression. Introducing controlled degradations encourages the model to learn manipulation cues rather than rely on obvious pixel differences.

Supervision Strategy

The model produces four outputs — three region predictions at different spatial resolutions (p_2, p_1, p_0) and one edge prediction. The highest-resolution region output p_0 is used as the final segmentation result.

Following the multi-scale representation principle discussed in Section 3.2, all three region outputs are supervised using the same binary ground-truth manipulation mask $y \in \{0,1\}^{H \times W}$. Applying supervision at multiple scales helps the model learn from features at different levels of detail. Lower-resolution predictions encourage the network to capture global structure, whereas the highest-resolution output focuses on subtle, local traces.

To enforce boundary sensitivity, the edge prediction head receives explicit supervision. A ground-truth edge map is generated on-the-fly from the binary manipulation mask using a morphological gradient (dilation minus erosion) with a kernel size of 3. An example of such an edge map is shown in Figure 3.3. The resulting edge maps highlight the transition between spliced and authentic regions and serve as the supervision target for the edge head.



Figure 3.3: Ground-truth mask from the CASIA v2.0 dataset and the corresponding edge map.

Loss Functions

Given the binary nature of the splicing localization task, the model is optimized using the binary cross-entropy with logits, a numerically stable combination of a sigmoid activation and binary cross-entropy loss. However, manipulated regions typically cover only a small portion of the image, resulting in class imbalance. To compensate for this, BCE is combined with Dice loss, which directly encourages the overlap between prediction and ground truth and has been shown to be effective under pixel-imbalanced conditions. Since some training images include padded areas, these pixels are excluded from all loss computations using a binary mask to ensure they do not affect optimization.

Because the training set contains both tampered and authentic images and only the former contain positive manipulation regions, the loss function must treat these two cases differently. The individual components that apply to tampered images are described below, and the handling of authentic images is discussed subsequently.

1. **Region Loss**

   The model produces three region predictions at different resolutions (p_2, p_1, p_0). Each prediction is optimized using the same loss function, the sum of BCE and Dice loss, but their contributions are weighted differently. The highest-resolution output p_0 receives the largest weight, while coarser outputs receive smaller

weights:

$$L_{\text{region}} = 1.0 \cdot L(p_0) + 0.5 \cdot L(p_1) + 0.25 \cdot L(p_2) \tag{3.1}$$

where

$$L(p_i) = \text{BCE}(p_i) + \text{Dice}(p_i)$$

The weighting scheme reflects that only the finest output is used during inference. The lower-resolution outputs then serve as auxiliary supervision, aiming to stabilize training and encourage feature learning at multiple scales. This aligns with the multi-scale design principle introduced earlier.

2. **Edge Loss**

   In addition to region loss, a dedicated edge loss is applied to the output of the boundary prediction head. The edge loss is defined as a BCE term between the predicted edge logits and the morphologically derived edge target:

   $$L_{\text{edge}} = \lambda \cdot \text{BCE}(p_{\text{edge}}, \text{edge}_{gt}) \tag{3.2}$$

   A weighting factor $\lambda = 6.0$ is used to increase the influence of the boundary signal. This encourages the model to focus on the transition between the manipulated and authentic regions, where the splicing traces are typically more pronounced.

For tampered images, the total loss combines both components:

$$L_{\text{tamp}} = L_{\text{region}} + L_{\text{edge}} \tag{3.3}$$

Authentic images contain no positive pixels and therefore cannot be optimized using Dice or edge losses. Instead, they are assigned only a weakly weighted BCE term on the highest-resolution prediction:

$$L_{\text{auth}} = \alpha \cdot \text{BCE}(p_0, 0)\,. \tag{3.4}$$

where $\alpha = 0.1$. This term discourages false positives.

The final training objective for a given mini-batch is obtained by summing the contributions from tampered and authentic samples:

$$L_{\text{total}} = L_{\text{tamp}} + L_{\text{auth}} \tag{3.5}$$

During validation, only the final region prediction p_0 is considered, while the remaining outputs serve solely as auxiliary training signals.

Optimization and Hyperparameters

The model is optimized using the Adam optimizer with a learning rate of 4×10^{-5}.

To stabilize training, an exponential moving average (EMA) of the model weights is used with a decay factor of 0.999. EMA maintains a smoothed version of the parameters and is only applied during validation and checkpoint selection.

Training runs for 35 epochs, with early stopping applied if the validation loss does not improve for 5 consecutive epochs. This prevents overfitting and avoids unnecessary training iterations. The best model checkpoint is selected based on the lowest validation loss.

For completeness, the key training hyperparameters are summarized in Table 3.1 below.

Table 3.1: Training configuration and key hyperparameters used in the final model.

Hyperparameter	**Value / Setting**
Optimizer	Adam
Learning rate	4×10^{-5}
Batch size (train / val)	8 / 1
Number of epochs	35
EMA decay	0.999
Dice loss weight	0.7
BCE loss weight	0.3
Edge loss λ	6.0

3.5 Model Inference

During inference, the model predicts a manipulation mask for an input image using the highest-resolution region output p_0. However, many images in the evaluation datasets, as well as the real-world inputs, are larger than the 512×512 resolution used during training. Simply resizing these inference-time images, even before they enter the model, can weaken or remove subtle manipulation traces, which conflicts with the resolution preservation principle discussed in Section 3.2. To avoid this, inference is performed using a sliding-window strategy.

In this setup, the input image is divided into partially overlapping patches of size 512×512, matching the training resolution. A stride of 384 pixels is used, which produces a 128-pixel overlap between adjacent patches and ensures that the model receives sufficient context around patch borders.

Each patch is processed independently by the model, and the resulting logits are placed back in their corresponding location in the full-size canvas. In overlapping regions, the logits are averaged to avoid seams between tiles.

After all patches are processed, the stitched logit map is passed through a sigmoid function. During validation, the probability map is directly thresholded. During test-time inference, it is further refined using the postprocessing pipeline described below. This sliding-window inference enables the model to process images of arbitrary size.

3.6 Postprocessing

To refine the raw model predictions and improve the quality of the final segmentation mask, a multi-step postprocessing procedure is applied. This step is based on the assumption that the splicing region typically represents a compact shape.

First, the output undergoes **hysteresis thresholding** [55], which, compared to direct thresholding, more robustly distinguishes manipulated from non-manipulated regions. The low and high thresholds were set to 0.3 and 0.5.

Small gaps and holes within the detected regions are then filled using the **hole filling** [55] algorithm.

Finally, the **morphological closing** [55] is applied to smooth boundaries and connect adjacent detected regions. A square structuring element with a kernel size of 23×23 pixels was used.

This postprocessing sequence aims to produce cleaner and more continuous final predictions while avoiding per-dataset threshold sweeping[1], which is common in the literature but impractical in real-world conditions.

3.7 Implementation and Environment

The proposed method is fully implemented in Python[2]. It significantly benefits from the PyTorch library, which enables efficient tensor operations, GPU-accelerated training, and structured dataset handling. Albumentations is then used for image augmentations and normalization. Additional libraries, including NumPy, OpenCV, scikit-image, and Matplotlib support data preparation, image manipulation, analysis, and visualization throughout the pipeline.

As presented in Section 3.3, the implementation uses the hierarchical Swin Transformer V2 architecture as the model's backbone. Microsoft's official PyTorch implementation [52, 53] is employed for this purpose.

Hardware and Software Setup

The experiments were conducted on a system equipped with a six-core Intel i7 processor, an NVIDIA RTX 2070 GPU, and 16GB of RAM. The implementation runs on Windows 10 with Python 3.11.

The main Python packages used are PyTorch 2.8.0+cu126, Albumentations 1.4.2, NumPy 1.26.1, OpenCV 4.9.0, scikit-image 0.22.0, and Matplotlib 3.8.3.

---

1. The practice of tuning a threshold to achieve the best results on a specific dataset.
2. Relevant libraries: `https://pytorch.org/`, `https://albumentations.ai/`, `https://numpy.org/`, `https://opencv.org/`, `https://scikit-image.org/`, `https://matplotlib.org/`

Final Deliverables

One of the objectives of this thesis is to address the lack of transparent and reproducible implementations of deep learning-based image splicing detection methods. Many existing approaches either do not share their code, restrict access to the utilized training datasets, or suffer from reproducibility issues due to missing details. This thesis provides the following final deliverables:

1. **Python Script**
   The proposed method is implemented in an accompanying Python script (`splicing_method.py`), runnable from the command line with a single input argument (the RGB image path). The script outputs a visualization of the prediction. Its form and components are described in Section 3.1.

2. **Jupyter Notebook**
   The Jupyter Notebook serves as a complementary resource to the Python script, offering a more transparent, detailed, and structured presentation of the method together with markdown explanations. It exposes the full training pipeline, dataset handling logic, model architecture, and evaluation routines, making the overall approach easier to follow, adapt, or build upon.

3. **Dataset Split Files**
   Text files listing the sample names used throughout the experiments are also provided. For in-distribution datasets, separate lists are available for the training, validation, and testing splits. For out-of-scope datasets, the files contain the samples used for evaluation. Where applicable, authentic and tampered samples are listed separately.

4. **Visualization Outputs**
   A large collection of visualized model outputs for each dataset is included to support qualitative evaluation. These examples illustrate the model's behavior across a wide range of samples from all utilized datasets. In addition, visualizations from the experiment with the human annotator introduced in Section 4.3 are also included.

5. **Final Trained Model**
   The final trained model is provided as a `.pth` PyTorch checkpoint file for direct use or further fine-tuning.

All deliverables are provided in compressed archive files accompanying this thesis. The archive containing the implementation, lists of used samples, and the trained model is subject to restricted access and will be made publicly available after the restriction period, while the archive containing visualization outputs is publicly accessible.

4 Evaluation and Results

Evaluating image splicing localization methods remains a complex and inconsistent task. Although numerous approaches have been proposed in the literature, there is still no standardized or commonly agreed evaluation protocol. Different studies often rely on distinct datasets, data splits, and even formulations of the problem itself. The choice of evaluation metrics also varies widely — some works report F1-score, others Area Under Curve (AUC), Intersection over Union (IoU), mean IoU (mIoU), or Matthews Correlation Coefficient (MCC) — making direct comparison difficult. Moreover, there is a very limited number of methods that report performance on fully authentic images, despite their importance for assessing false positive behavior.

A further complication is that many datasets used in prior work mix different types of manipulations. For instance, CASIA v2.0, one of the most widely adopted training sources, contains both splicing and copy-move forgeries, yet methods presented as splicing-focused are often trained on the whole dataset. Combined with the frequent absence of key implementation details, this makes it difficult to evaluate methods under equivalent conditions or reproduce results reliably.

Evaluation is also affected by the fact that some datasets cannot be easily traced (e.g., custom synthetic datasets) and by the limited availability of complete, publicly accessible implementations.

In addition, the research focus has gradually shifted from splicing localization to the broader problem of image manipulation localization. As a consequence, comparisons across studies increasingly mix methods intended for different manipulation types, further blurring the boundaries between the tasks.

Despite attempts to introduce more unified benchmarking procedures, such as protocols utilized by Ma, X. et al. [23] for generic manipulation localization, these standards have not yet been consistently adopted within the splicing literature.

These limitations and inconsistencies have been acknowledged in earlier forensic literature [23], and are likewise acknowledged here. Consequently, the comparisons in this chapter should be viewed as indicative rather than absolute benchmarks. The aim is to provide insight into how the proposed approach performs, rather than to claim

superiority under fully standardized conditions. The evaluation considers both quantitative and qualitative aspects of splicing localization performance. The remainder of this chapter introduces the evaluation metrics and presents results on several datasets, including the test splits and out-of-scope sources.

4.1 Evaluation Metrics

The evaluation is performed at the pixel level by comparing spliced (1) and authentic (0) pixels in the predicted output mask (P) against the ground truth mask (G). To quantitatively assess the performance, it is necessary to distinguish between correctly and incorrectly classified pixels. Each pixel can therefore be categorized into one of the following four groups:

- true positive (TP) – G: 1, P: 1
- false positive (FP) – G: 0, P: 1
- true negative (TN) – G: 0, P: 0
- false negative (FN) – G: 1, P 0

As noted in the opening of this chapter, there is no single universally adopted metric for image splicing localization. Recent work most frequently reports the F1-score and MCC, largely due to their suitability for the highly imbalanced nature of the task. However, different studies adopt different evaluation conventions.

To ensure compatibility with the comparison methods considered in this thesis, the F1-score, mIoU, and permuted mIoU (p-mIoU) are reported. Authentic images are evaluated using Accuracy (Acc) and permuted Acc (p-Acc). For completeness, IoU is also included, although it appears less frequently in recent literature, as it does not account for class imbalance. The remainder of this section provides an overview of the metrics used in the evaluation.

F1-score

The **F1-score** measures the harmonic mean of precision and recall:

$$\text{Precision} = \frac{TP}{TP + FP} \tag{4.1}$$

$$\text{Recall} = \frac{TP}{TP + FN} \tag{4.2}$$

$$F_1 = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}} \tag{4.3}$$

It reflects the model's ability to correctly identify manipulated pixels while penalizing both missed detections and incorrect spliced predictions. The F1-score ranges from 0 (no manipulated pixels correctly identified) to 1 (perfect localization).

IoU-based Metrics

The standard **IoU** quantifies the overlap between the ground truth mask G and the predicted mask P:

$$\text{IoU(G, P)} = \frac{|G \cap P|}{|G \cup P|} = \frac{TP}{TP + FP + FN} \tag{4.4}$$

To better balance the contribution of both classes, the **mIoU** computes IoU for each class separately. The IoU for the non-manipulated class is computed using complemented masks G′ and P′. The two values are then averaged:

$$\begin{aligned} \text{mIoU (G, P)} &= \frac{1}{2}\frac{|G \cap P|}{|G \cup P|} + \frac{1}{2}\frac{|G^{'} \cap P^{'}|}{|G^{'} \cup P^{'}|} \\ &= \frac{1}{2}\frac{TP}{TP + FP + FN} + \frac{1}{2}\frac{TN}{TN + FP + FN}. \end{aligned} \tag{4.5}$$

The **p-mIoU** is defined as:

$$\text{p-mIoU}(G, P) = \max\big(\text{mIoU}(G, P), \text{mIoU}(G, P')\big)\,. \tag{4.6}$$

This provides a measure of how well the model separates the two regions, independent of the particular label assignment.

All IoU-based metrics range from 0 to 1, where higher values indicate better overlap.

Accuracy

For authentic images, the ground truth mask consists solely of non-manipulated pixels and therefore includes no positive pixels. As a result, mIoU, IoU, or F1-score would not meaningfully reflect performance for these samples. In such cases, **accuracy** is used, defined as the ratio of correctly predicted pixels to the total number of pixels. It is computed as:

$$\begin{aligned} \text{Acc (G, P)} &= \frac{|G \cap P| + |G^{'} \cap P^{'}|}{|G \cup G^{'}|} \\ &= \frac{TP + TN}{TP + TN + FP + FN} \end{aligned} \tag{4.7}$$

The corresponding **p-Acc** is then obtained as:

$$\text{p-Acc}(G, P) = \max\big(\text{Acc}(G, P),\ \text{Acc}(G, P')\big) \tag{4.8}$$

Both accuracy versions range from 0 to 1, with higher values indicating a greater portion of correctly classified pixels.

4.2 Quantitative Evaluation

Given the challenges outlined at the beginning of this chapter, the evaluation presented in this thesis is designed to be indicative rather than absolute. The goal is not to establish a universal benchmark, but to position the proposed approach within the broader forensic landscape.

To provide context for the reported results, two representative comparison schemes from recent work are considered. These schemes are not replicated in this thesis. The proposed model is evaluated against the methods reported within them:

1. **CAT-Net four datasets scheme (Protocol A)**
   CAT-Net [40] is presented as a splicing localization method, trained on a combination of CASIA v2.0 (authentic/tampered), Fantastic Reality (authentic/tampered), IMD2020 (tampered), and Spliced COCO (tampered). These datasets overlap with the sources used in this thesis, although CAT-Net is trained on a substantially larger number of samples. Its architecture follows a dual-stream design (RGB and DCT). The RGB stream is initialized with pretrained ImageNet weights, analogous to our encoder initialization.

2. **DEFACTO + fine-tuning scheme (Protocol B):**
   The second reference method [56] is a splicing localization method whose backbone architectures are also initialized with ImageNet-pretrained weights. It is first trained on 23,000 spliced images from the synthetic DEFACTO dataset and consequently fine-tuned on several real-world datasets (CASIA v2.0, NIST16, IMD2020) using an additional 7,135 samples. In contrast, the model proposed in this thesis is trained on a comparable absolute number of real samples (10,096), but without the large-scale DEFACTO pretraining phase. Two datasets used in our training (CASIA v2.0 and IMD2020) overlap with the fine-tuning stage datasets of Protocol B. Additionally, it is important to note that, in this thesis, only the splicing portions of datasets CASIA v2.0, CASIA v1.0, and IMD2020 are used. In contrast, the reference publication pretrains, fine-tunes, and evaluates using subsets that, based on the reported sample counts, contain a mixture of manipulation types.

The same proposed model (trained on the four-dataset mixture) is used for the comparisons corresponding to Protocols A and B. In each case, the evaluation follows the metrics used in the corresponding publications.

The performance values for the two reference methods are taken directly from their original publications. As the corresponding papers also report results for several additional approaches used in their comparisons, this thesis includes a selected subset of those reported numbers in the relevant tables to provide a broader context.

Quantitative Results

Separate instances of the dataset loading pipeline were created for each test subset derived from the training data, as well as for all additional sources used exclusively for evaluation. Tampered images and, where available, authentic images from a given dataset were handled by separate Dataset instances and loaders, allowing their results to be reported independently. The exact number of samples from each dataset used during evaluation is provided in Table 2.1.

Table 4.1 summarizes the performance of the model across all evaluated datasets. Results are shown for both the datasets involved in training (test splits) and the total of six out-of-scope sources. Datasets that provided tampered samples during training are shown in **bold**, while those that contributed also authentic training images are marked with "*". Reported metrics include F1, IoU, p-mIoU, mIoU, and, where an authentic subset is used, Acc and pAcc. A dash indicates that no authentic subset was evaluated.

Table 4.1: Performance of the proposed method across different datasets evaluated with various metrics. For authentic images, Acc and p-Acc are reported.

Dataset	F1-score	IoU	mIoU	p-mIoU	Acc	p-Acc
CASIA v2.0 [7, 8] *	0.8538	0.7449	0.8497	0.8497	0.9854	0.9854
Fantastic Reality [43] *	0.9041	0.8250	0.8914	0.8914	0.9701	0.9701
IMD2020 [44]	0.6086	0.4374	0.6893	0.6893	–	–
SP COCO [40, 49]	0.9230	0.8571	0.9144	0.9144	0.9807	0.9807
CARVALHO [42]	0.5123	0.3444	0.6160	0.6160	0.9937	0.9937
CASIA v1.0 [7, 8]	0.8129	0.6847	0.8195	0.8195	0.9826	0.9826
CUISDE [41]	0.8202	0.6952	0.7891	0.7891	0.9774	0.9774
DIS25k [47]	0.7549	0.6063	0.7806	0.7806	–	–
In-the-Wild [48]	0.5292	0.3598	0.6090	0.6090	–	–
MISD [51]	0.7315	0.5767	0.7409	0.7409	–	–

Overall, the model demonstrates strong performance on the in-distribution datasets, achieving high pixel-level scores on CASIA v2.0, SP COCO, and Fantastic Reality. This suggests that the model captures the characteristics of the training distribution well and can localize the manipulated region reliably when tested under similar conditions.

IMD2020 is the most challenging of the in-distribution sets, which is reflected in its lower F1-score (≈ 0.61). This is consistent with the increased difficulty of its manipulations.

For the out-of-scope datasets, performance varies. CASIA v1.0, CUISDE, MISD, and DIS25k achieve competitive F1-scores in the range of 0.73–0.82, indicating a degree of cross-dataset generalization. In contrast, CARVALHO stands out as the most difficult source, with the lowest overall F1-score and IoU. As discussed in Section 2.1, this dataset contains carefully blended people insertions with very smooth boundaries and highly photorealistic compositions.

We assume this drop is partly linked to the patch-based training regime used in the proposed approach. Because the model operates on 512×512 crops, and only a subset of samples are smaller than this resolution, it has access to limited global context during both training and inference. This design aims to preserve spatial resolution before the sample enters the network and works well on datasets where manipulated regions are relatively compact or where a 512×512 window captures most of the relevant content. However, it becomes less suitable for sources such as CARVALHO and In-the-Wild, where spliced regions are larger and more smoothly blended, leaving few local cues for the model to rely on. Potential ways to mitigate this limitation are discussed in Section 5.1.

Tables 4.2 and 4.3 present the comparison corresponding to Protocol A (CAT-Net scheme). Table 4.2 reports performance on the test splits of the datasets shared with CAT-Net, while Table 4.3 presents results on the out-of-scope CARVALHO and CUISDE datasets. In addition to the CAT-Net scores, both tables include the values reported for EXIF [48] and Mantra-Net [36]. For these two methods, only the out-of-scope performance is shown. Results are reported in terms of mIoU, p-mIoU, Acc, and pAcc.

The proposed model performs competitively under this evaluation scheme, especially considering the significantly smaller training set used in comparison with CAT-Net. On the shared test splits, the results are in line with those of CAT-Net across both accuracy and pixel-level metrics. On the out-of-scope datasets, the model continues to perform within the range of the methods reported in the CAT-Net paper.

Table 4.4 and Table 4.5 summarize the comparison with methods reported under Protocol B (DEFACTO + fine-tuning scheme). Since

Table 4.2: Comparison on datasets used for training by both the proposed approach and CAT-Net, with each method evaluated on its own test splits.

Method	CASIA v2.0				Fantastic Reality				SP COCO		IMD2020	
	authentic		tampered		authentic		tampered		tampered		tampered	
	Acc	p-Acc	mIoU	p-mIoU	Acc	p-Acc	mIoU	p-mIoU	mIoU	p-mIoU	mIoU	p-mIoU
CAT-Net [40]	0.9966	0.9966	0.8763	0.8769	0.9973	0.9975	0.9331	0.9331	0.9387	0.9387	0.7600	0.7653
Ours	0.9854	0.9854	0.8497	0.8497	0.9701	0.9701	0.8914	0.8914	0.9144	0.9144	0.6893	0.6893

Table 4.3: Comparison with CAT-Net and the methods reported in their paper on the out-of-scope CARVALHO and CUISDE datasets.

Method	CUISDE				CARVALHO			
	authentic		tampered		authentic		tampered	
	Acc	**p-Acc**	**mIoU**	**p-mIoU**	**Acc**	**p-Acc**	**mIoU**	**p-mIoU**
EXIF [48]	0.6760	0.6820	0.8081	0.8529	0.6204	0.6356	0.4840	0.5133
Mantra-Net [36]	0.9566	0.9566	0.5234	0.5240	0.9865	0.9865	0.5628	0.5646
CAT-Net [40]	0.9954	0.9954	0.8305	0.9009	0.9979	0.9979	0.7944	0.7944
Ours	0.9774	0.9774	0.7891	0.7891	0.9937	0.9937	0.6160	0.6160

the reference method is fine-tuned on IMD2020 and the proposed model is trained on this dataset, IMD2020 represents the only point of partial in-distribution comparison. The remaining datasets reported in the reference paper — CASIA v1.0, CUISDE, and CARVALHO — are included here as out-of-scope sources.

Both tables include a subset of the additional comparison methods included in the original publication. Their abbreviations are retained for consistency. ELA [57] and CFA [58] represent traditional approaches, while the remaining methods are deep learning-based. All reported values follow the metrics used in the reference work. Table 4.4 provides F1-scores, and Table 4.5 provides IoU values.

Across the datasets considered in this scheme, the proposed model shows outcomes consistent with the trends observed in the earlier evaluation. On CASIA v1.0, both the F1-score and IoU are among the stronger results in the reported comparison. On CUISDE, the F1-score falls near the upper part of the reported range, while the IoU is moderate. The results on IMD2020 are numerically close to the stronger deep learning-based methods, while the scores on CARVALHO remain no-

Table 4.4: F1-score comparison with methods reported under Protocol B in the reference paper on IMD2020 and out-of-scope datasets.

Method	CASIA v1.0	CUISDE	IMD2020	CARVALHO	Average
ELA [57]	0.214	0.470	0.183	0.267	0.284
CFA [58]	0.263	0.574	0.229	0.402	0.367
RRU [22]	0.577	0.883	0.451	0.516	0.607
MVSS [39]	0.671	0.889	0.318	0.582	0.615
PSCC [59]	0.661	0.881	0.229	0.537	0.577
CFL [60]	0.694	0.808	0.483	0.652	0.659
EITL [61]	0.709	0.883	0.610	0.606	0.702
PIM [62]	0.698	0.875	0.461	0.674	0.677
DS-SNL+HFF [56]	0.722	0.920	0.598	0.716	0.739
Ours	0.813	0.820	0.609	0.512	0.689

Table 4.5: IoU comparison with methods reported under Protocol B in the reference paper on IMD2020 and out-of-scope datasets.

Method	CASIA v1.0	CUISDE	IMD2020	CARVALHO	Average
ELA [57]	–	–	–	–	–
CFA [58]	–	–	–	–	–
RRU [22]	0.319	0.850	0.202	0.412	0.446
MVSS [39]	0.338	0.854	0.234	0.512	0.485
PSCC [59]	0.345	0.843	0.145	0.351	0.421
CFL [60]	0.458	0.779	0.235	0.547	0.505
EITL [61]	0.479	0.876	0.240	0.533	0.532
PIM [62]	0.452	0.881	0.268	0.576	0.544
DS-SNL+HFF [56]	0.459	0.909	0.271	0.591	0.558
Ours	0.685	0.695	0.437	0.344	0.540

ticeably lower, reflecting the difficulty discussed above. Overall, these observations indicate that the proposed approach aligns with the spectrum of methods reported in the reference work, while acknowledging the differences in experimental conditions.

4.3 Qualitative Evaluation

Quantitative metrics such as mIoU or F1-score offer a practical way to measure how well the model aligns with the available ground-truth annotations. However, these values capture only part of the picture, especially in image splicing localization, where the goal is not only to maximize numerical scores but also to provide cues that can assist a forensic analyst in identifying and verifying potentially tampered regions.

For this reason, the quantitative results in the previous section are complemented by a qualitative evaluation of the predicted masks. Visual inspection helps assess whether the highlighted areas form coherent and meaningful regions, offering insight that numerical metrics alone cannot provide. The examples shown in this section illustrate the model's behavior on individual datasets, showing both its strengths and the situations in which it struggles. Representative cases are presented in Figures 4.1–4.3.

Across the datasets included during training, the model generally produces clear and well-localized masks. On CASIA v2.0, the predicted spliced regions typically align closely with the ground truth mask, and boundary precision is often high. A similarly stable behavior is observed for SP COCO and Fantastic Reality, where the model reliably highlights the manipulated region without excessive false positives. Performance on IMD2020, which contains much more complicated manipulations, shows greater variability. Many spliced regions are detected correctly, but the model occasionally captures only part of the manipulated area, overlooks some tampered areas, or produces softer boundaries, reflecting the increased difficulty of this dataset.

When evaluated on datasets not seen during training, the model still provides useful cues, though performance naturally declines as the distribution shift increases. On CASIA v1.0, DIS25k, and CUISDE, the masks remain generally coherent and capture manipulated regions in a large number of cases. On more challenging sources, such as CARVALHO and the In-the-Wild datasets, where manipulations are subtle and carefully blended, the model tends to produce fragmented masks, introduce holes and false positives, and often misses the spliced region entirely.

The MISD dataset presents its own challenges. As each image contains multiple spliced regions, the model sometimes localizes only a subset of them. Moreover, as noted in Section 2.1, the ground-truth masks in MISD are coarse and often extend beyond the actual manipulated areas. In several examples, the model's predictions were cleaner and more precise than the annotations themselves, leading to lower reported metrics despite visually satisfactory results. This makes the MISD dataset a clear example of why qualitative analysis is an essential complement to metric-based evaluation.

Finally, several particularly challenging cases were identified across the datasets to illustrate the model's limitations. These include images with extremely subtle boundary transitions, carefully blended splicing, low-contrast regions, or highly textured backgrounds. In such situations, the model may either produce scattered false positives or miss the manipulation entirely. Among all evaluated datasets, CARVALHO proved to be the most challenging for the model.

Lastly, a small human baseline was constructed for comparison. A subset of CASIA v1.0 images (including both authentic and manipulated samples) was shown to a human annotator with no prior exposure to the dataset. The annotator was asked to outline any areas suspected of being spliced, while knowing that an image might contain none, one, or multiple manipulated regions. In addition to many correctly marked samples and some missed manipulated regions, the results show that the participant often produced false positives in visually ambiguous areas, especially in authentic images that seemed semantically unusual or otherwise atypical. The annotator also noted that semantic inconsistencies were often a stronger cue than the visual quality of the edit itself. Figure 4.4 shows several examples from the experiment. This human baseline serves as an additional reference when interpreting the model's qualitative results.

To avoid redundancy and maintain clarity, only a subset of visualized outputs is included in this section. These examples are selected to illustrate both typical predictions and challenging cases. A more comprehensive set of visual results is then provided in the appendices, and additionally, a wider range of predicted visualized outputs for all datasets used is included in the electronic archive of the thesis.

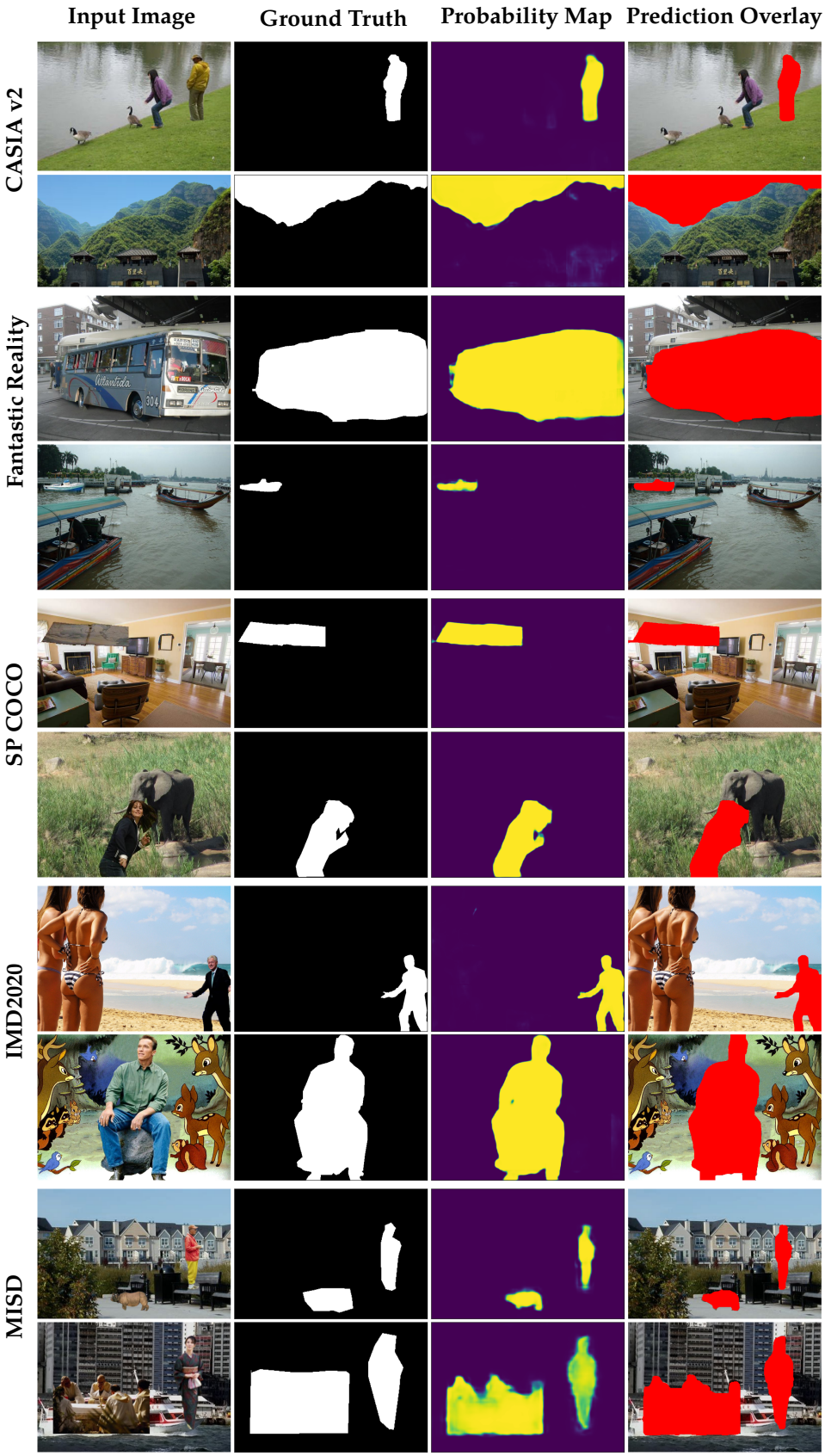


Figure 4.1: Qualitative splicing detection results for two samples per dataset (Part 1). From left to right: input image, ground-truth mask, predicted probability map, and predicted splicing mask overlay.

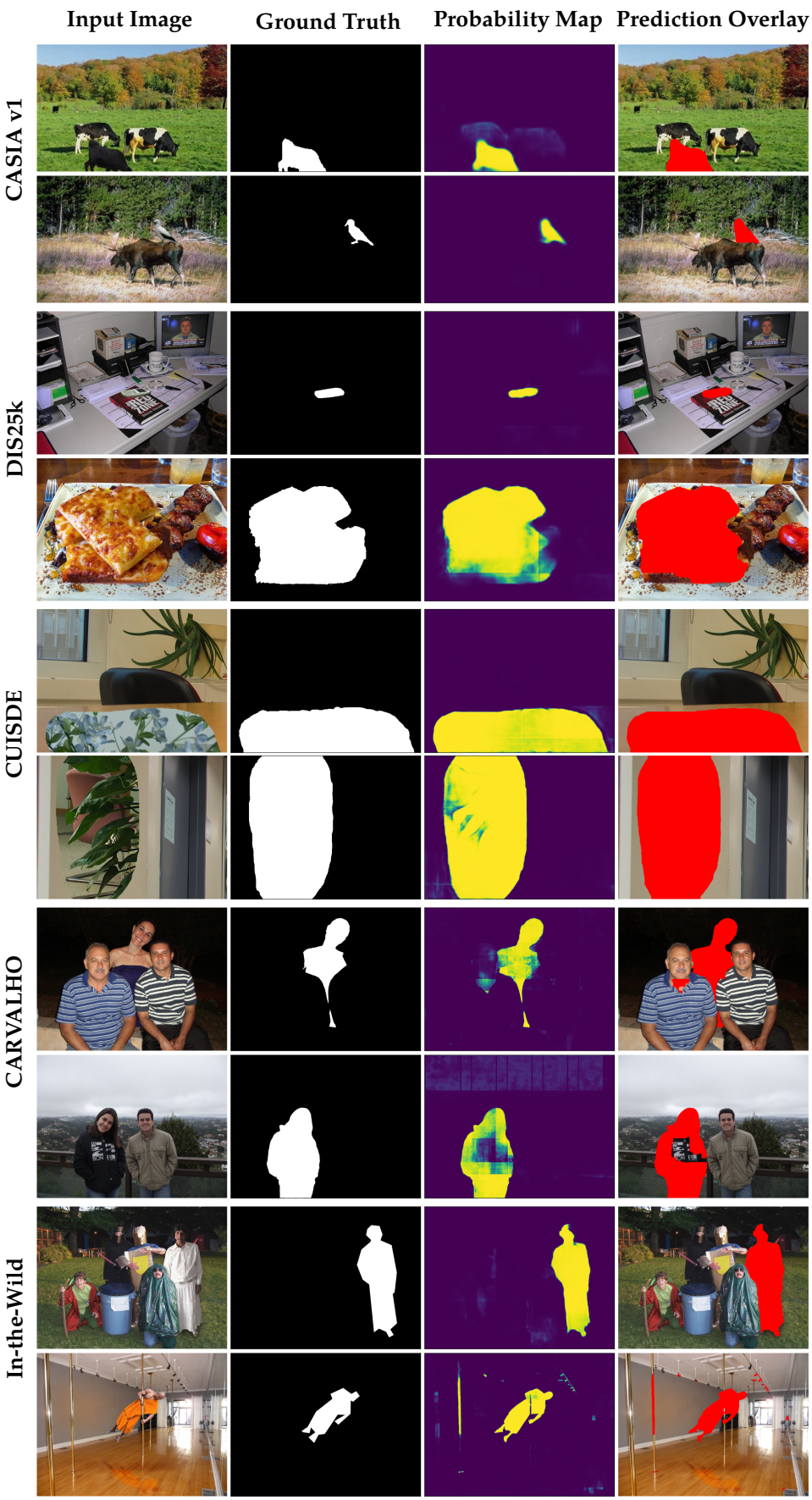


Figure 4.2: Qualitative splicing detection results for two samples per dataset (Part 2). From left to right: input image, ground-truth mask, predicted probability map, and predicted splicing mask overlay.

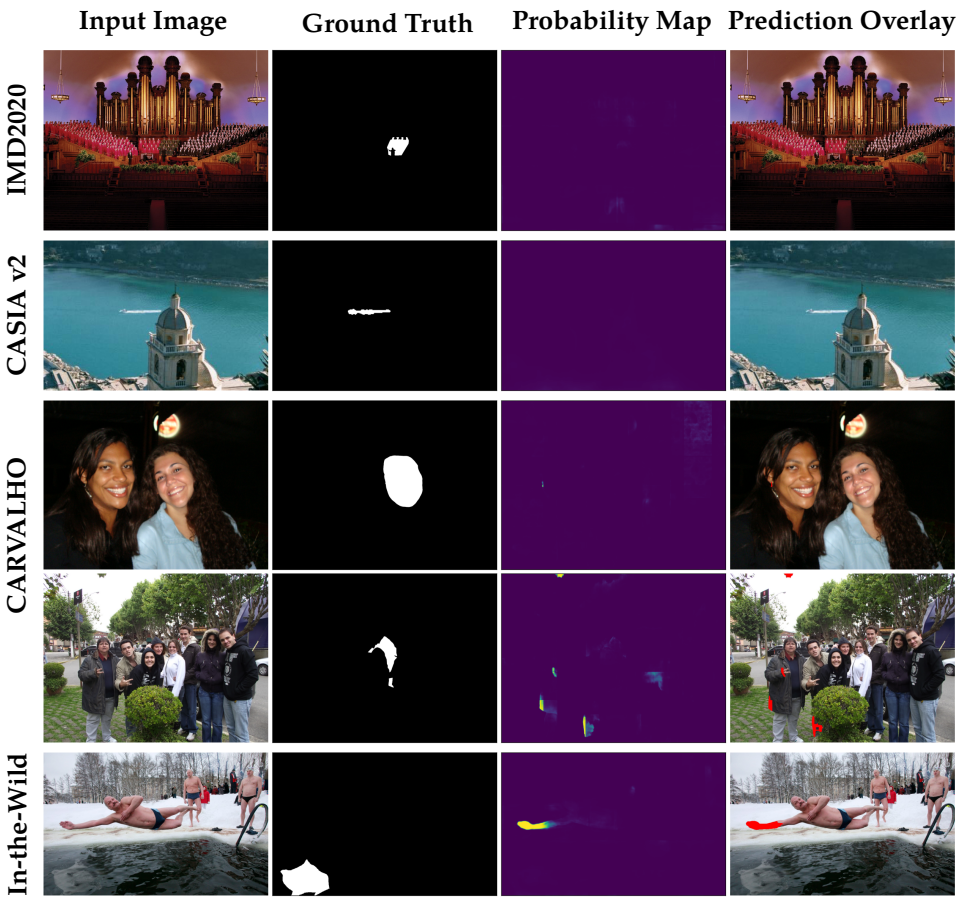


Figure 4.3: Qualitative results on particularly challenging samples from different datasets, illustrating scenarios where the proposed method struggles. From left to right: input image, ground-truth mask, predicted probability map, and predicted splicing mask overlay.

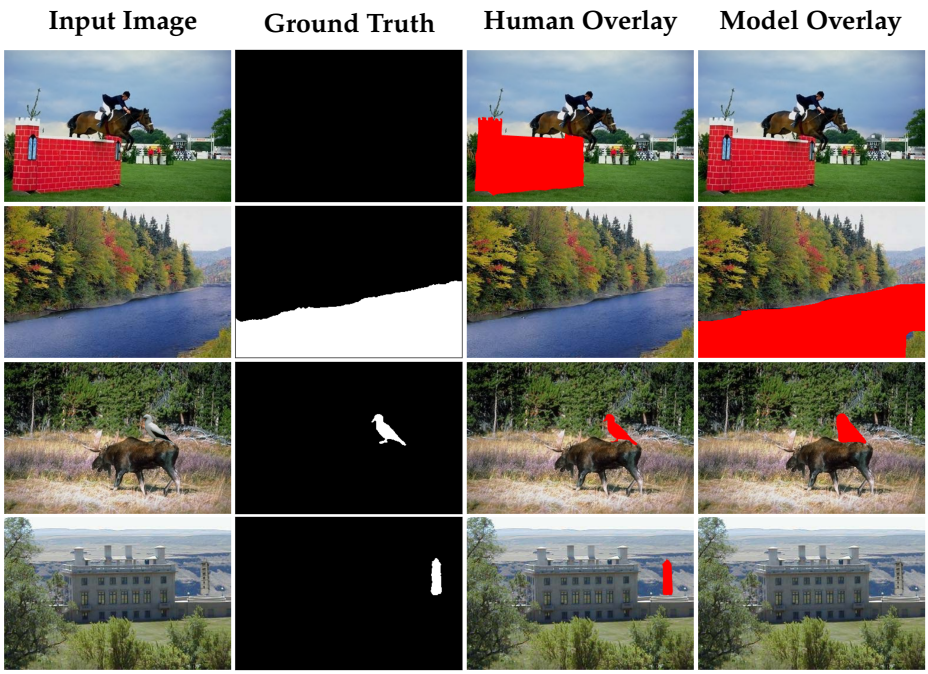


Figure 4.4: Examples from the experiment with a human annotator. From left to right: input image, ground-truth mask, human prediction, and the model's splicing mask prediction overlaid on the input image.

5 Conclusion and Future Work

The main aim of this thesis was to explore the detection of image splicing using deep learning techniques.

To address this objective, the work first introduced the broader context of digital image forgery. In order to better understand the underlying principles of splicing detection, existing hand-crafted and deep learning-based approaches were reviewed, highlighting how the field evolved from feature engineering to end-to-end learning. Furthermore, publicly available datasets commonly used in forensic research were surveyed, and the datasets selected for training and evaluation in this thesis were described in detail.

Based on insights from prior research, a hierarchical transformer-based framework for pixel-level splicing detection was designed and implemented. The architecture and training strategy were adapted to the specifics of manipulation localization, which introduces challenges not present in the standard semantic segmentation. The approach incorporates multi-scale supervision, encouraging prediction consistency across different spatial resolutions, and an auxiliary boundary-detection head, enforcing stronger learning around manipulation boundaries where forensic traces are often most pronounced. Throughout the implementation, transparency and reproducibility were prioritized. Therefore, all implementation details and experimental configurations were clearly documented and provided.

The presented method was evaluated on publicly available datasets using both quantitative metrics and qualitative visualization. The results show that the model can localize manipulated regions and exhibits promising generalization on images outside the training domain, while using a significantly simpler architecture than several recently published multi-stream or computationally heavy solutions.

Although the proposed framework achieves encouraging results, splicing detection in real-world conditions remains highly challenging. Manipulations vary widely in quality and execution, and forensic traces may be very subtle or deliberately concealed. These factors make generalization particularly difficult and highlight that even effective models have practical limits when encountering previously

unseen manipulation styles. This suggests that there is still room for improvement and further exploration.

5.1 Future Work

The findings of this thesis highlight several natural directions for future research:

- **Incorporating additional forensic cues**
  The current model operates only on RGB input. Extending the architecture by incorporating forensic feature maps, e.g., noise residuals, illumination inconsistencies, or compression artifacts, could further improve the robustness of the approach.

- **Extending the approach to other manipulation types**
  The method is tailored to pixel-level splicing detection. However, similar architectural principles and training strategies could be applied to related manipulation types, such as copy-move forgery, inpainting, or content removal. Extending the framework toward a unified manipulation localization model represents a natural progression of the work.

- **Increasing global context**
  The current model operates on 512×512 crops, which preserve spatial resolution but limit the amount of global context available during training and inference. Several strategies could be explored to mitigate this limitation. One option is to train on larger patches, which directly increases the amount of image content available to the model. Alternatively, the resizing strategy proposed by Ma, X. et al. [23] could be adopted: padding all images to a fixed resolution and downscaling only the larger ones, with aspect ratio preserved. Finally, if sufficient context is available, modules that provide more explicit global reasoning could be integrated into the hierarchical Swin backbone, such as a transformer layer operating on flattened feature maps or CBAM-style attention. These modifications could further enhance the model's ability to capture long-range dependencies beyond what shifted-window attention in Swin offers.

- **Leveraging larger and more diverse datasets**
  The performance of splicing detection models is strongly influenced by the quality and diversity of the training data. In this work, four datasets were combined to capture a wider range of splicing artifacts. However, constructing or accessing balanced datasets that cover diverse manipulation scenarios — different artifact types, blending methods, region sizes, and multiplicity of regions — remains a significant challenge. Addressing this could enhance not only the robustness and performance of the approach proposed in this thesis, but also the other future methods in image forensics.

Beyond the directions listed above, several further refinements could be explored, such as experimenting with alternative decoder designs, different attention mechanisms, or refined loss formulations. These could help address edge cases where manipulation traces are subtle or heavily blended.

Bibliography

1. SINGH, Mayank; TYAGI, Vipin; GUPTA, P. K.; FLUSSER, Jan; ÖREN, Tuncer; SONAWANE, V. R. (eds.). *Advances in Computing and Data Sciences: 5th International Conference, ICACDS 2021, Nashik, India, April 23–24, 2021, Revised Selected Papers, Part II*. Vol. 1441. 1st ed. Cham, Switzerland: Springer Cham, 2021. Communications in Computer and Information Science. ISBN 978-3-030-88243-3. ISSN 1865-0929. Available from DOI: 10.1007/978-3-030-88244-0.
2. MEENA, Kunj Bihari; TYAGI, Vipin. Image splicing forgery detection techniques: A review. In: *Advances in Computing and Data Sciences: 5th International Conference, ICACDS 2021, Nashik, India, April 23–24, 2021, Revised Selected Papers, Part II*. Springer, 2021, pp. 364–388. Available from DOI: 10.1007/978-3-030-88244-0_35.
3. QURESHI, Muhammad Ali; DERICHE, Mohamed. A review on copy move image forgery detection techniques. In: *2014 IEEE 11th International Multi-Conference on Systems, Signals Devices (SSD14)*. 2014, pp. 1–5. Available from DOI: 10.1109/SSD.2014.6808907.
4. KUMARI, Ritesh; GARG, Hitendra. Image splicing forgery detection: A review. *Multimedia Tools and Applications*. 2024. ISSN 1573-7721. Available from DOI: 10.1007/s11042-024-18801-z.
5. MEENA, Kunj Bihari; TYAGI, Vipin. A Deep Learning based Method for Image Splicing Detection. *Journal of Physics: Conference Series*. 2021, vol. 1714, no. 1, p. 012038. Available from DOI: 10.1088/1742-6596/1714/1/012038.
6. ITIER, Vincent; STRAUSS, Olivier; MOREL, Laurent; PUECH, William. Color noise correlation-based splicing detection for image forensics. *Multimedia Tools and Applications*. 2021, vol. 80, no. 9, pp. 13215–13233. ISSN 1573-7721. Available from DOI: 10.1007/s11042-020-10326-5.
7. DONG, Jing; WANG, Wei; TAN, Tieniu. CASIA Image Tampering Detection Evaluation Database. In: *2013 IEEE China Summit and*

International Conference on Signal and Information Processing. IEEE, 2013. Available from DOI: 10.1109/chinasip.2013.6625374.

8. PHAM, Nam Thanh; LEE, Jong-Weon; KWON, Goo-Rak; PARK, Chun-Su. Hybrid Image-Retrieval Method for Image-Splicing Validation. *Symmetry*. 2019, vol. 11, no. 1, p. 83. Available from DOI: 10.3390/sym11010083.
9. ALAHMADI, Amani; HUSSAIN, Muhammad; ABOALSAMH, Hatim; MUHAMMAD, Ghulam; BEBIS, George; MATHKOUR, Hassan. Passive detection of image forgery using DCT and local binary pattern. *Signal, Image and Video Processing*. 2017, vol. 11, no. 1, pp. 81–88. ISSN 1863-1711. Available from DOI: 10.1007/s11760-016-0899-0.
10. SHARMA, Surbhi; GHANEKAR, Umesh. Spliced image classification and tampered region localization using local directional pattern. *Int. J. Image Graph. Signal Process.* 2019, vol. 11, no. 3, pp. 35–42. Available from DOI: 10.5815/ijigsp.2019.03.05.
11. KANWAL, Navdeep; GIRDHAR, Akshay; KAUR, Lakhwinder; BHULLAR, Jaskaran S. Digital image splicing detection technique using optimal threshold based local ternary pattern. *Multimedia Tools and Applications*. 2020, vol. 79, no. 19, pp. 12829–12846. ISSN 1573-7721. Available from DOI: 10.1007/s11042-020-08621-2.
12. FANG, Zhen; WANG, Shuozhong; ZHANG, Xinpeng. Image Splicing Detection Using Color Edge Inconsistency. In: *2010 International Conference on Multimedia Information Networking and Security*. 2010, pp. 923–926. Available from DOI: 10.1109/MINES.2010.196.
13. JAISWAL, Ankit Kumar; SRIVASTAVA, Rajeev. A technique for image splicing detection using hybrid feature set. *Multimedia Tools and Applications*. 2020, vol. 79, no. 17, pp. 11837–11860. ISSN 1573-7721. Available from DOI: 10.1007/s11042-019-08480-6.
14. DAS, Debjit; NASKAR, Ruchira. Image splicing detection using low-dimensional feature vector of texture features and Haralick features based on Gray Level Co-occurrence Matrix. *Signal Processing: Image Communication*. 2024, vol. 125, p. 117134. ISSN 0923-5965. Available from DOI: 10.1016/j.image.2024.117134.

15. MOGHADDASI, Zahra; JALAB, Hamid A.; NOOR, Rafidah Md. Image splicing forgery detection based on low-dimensional singular value decomposition of discrete cosine transform coefficients. *Neural Computing and Applications*. 2019, vol. 31, no. 11, pp. 7867–7877. ISSN 1433-3058. Available from DOI: 10.1007/s00521-018-3586-y.

16. EL-ALFY, E-Sayed M.; QURESHI, Muhammad Ali. Combining spatial and DCT based Markov features for enhanced blind detection of image splicing. *Pattern Analysis and Applications*. 2015, vol. 18, no. 3, pp. 713–723. ISSN 1433-755X. Available from DOI: 10.1007/s10044-014-0396-4.

17. HE, Zhongwei; LU, Wei; SUN, Wei; HUANG, Jiwu. Digital image splicing detection based on Markov features in DCT and DWT domain. *Pattern Recognition*. 2012, vol. 45, no. 12, pp. 4292–4299. ISSN 0031-3203. Available from DOI: 10.1016/j.patcog.2012.05.014.

18. HAN, Jong Goo; PARK, Tae Hee; MOON, Yong Ho; EOM, Il Kyu. Quantization-based Markov feature extraction method for image splicing detection. *Machine Vision and Applications*. 2018, vol. 29, no. 3, pp. 543–552. ISSN 1432-1769. Available from DOI: 10.1007/s00138-018-0911-5.

19. ZENG, Hui; ZHAN, Yifeng; KANG, Xiangui; LIN, Xiaodan. Image splicing localization using PCA-based noise level estimation. *Multimedia Tools and Applications*. 2017, vol. 76, no. 4, pp. 4783–4799. ISSN 1573-7721. Available from DOI: 10.1007/s11042-016-3712-8.

20. ZHU, Nan; LI, Zhao. Blind image splicing detection via noise level function. *Signal Processing: Image Communication*. 2018, vol. 68, pp. 181–192. ISSN 0923-5965. Available from DOI: 10.1016/j.image.2018.07.012.

21. WANG, Jinwei; NI, Qiye; LIU, Guangjie; LUO, Xiangyang; JHA, Sunil Kr. Image splicing detection based on convolutional neural network with weight combination strategy. *Journal of Information Security and Applications*. 2020, vol. 54, p. 102523. ISSN 2214-2126. Available from DOI: 10.1016/j.jisa.2020.102523.

22. BI, Xiuli; WEI, Yang; XIAO, Bin; LI, Weisheng. RRU-Net: The Ringed Residual U-Net for Image Splicing Forgery Detection. In: *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*. 2019, pp. 30–39. Available from DOI: 10.1109/CVPRW.2019.00010.

23. MA, Xiaochen; DU, Bo; JIANG, Zhuohang; DU, Xia; HAMMADI, Ahmed Y. Al; ZHOU, Jizhe. *IML-ViT: Benchmarking Image Manipulation Localization by Vision Transformer*. 2024. Available from DOI: 10.48550/arXiv.2307.14863.

24. ZHANG, Jing; ZHAO, Yun; SU, Yuting. A new approach merging markov and DCT features for image splicing detection. In: *2009 IEEE International Conference on Intelligent Computing and Intelligent Systems*. 2009, vol. 4, pp. 390–394. Available from DOI: 10.1109/ICICISYS.2009.5357642.

25. SHI, Yun Q.; CHEN, Chunhua; CHEN, Wen. A Natural Image Model Approach to Splicing Detection. In: 2007, pp. 51–62. Available from DOI: 10.1145/1288869.1288878.

26. PAN, Xunyu; ZHANG, Xing; LYU, Siwei. Exposing image splicing with inconsistent local noise variances. *2012 IEEE International Conference on Computational Photography, ICCP 2012*. 2012. Available from DOI: 10.1109/ICCPhot.2012.6215223.

27. IAKOVIDOU, Chryssanthi; ZAMPOGLOU, Markos; PAPADOPOULOS, Symeon; KOMPATSIARIS, Yiannis. Content-aware detection of JPEG grid inconsistencies for intuitive image forensics. *Journal of Visual Communication and Image Representation*. 2018, vol. 54, pp. 155–170. ISSN 1047-3203. Available from DOI: 10.1016/j.jvcir.2018.05.011.

28. LIU, Bo; PUN, Chi-Man; YUAN, Xiao-Chen. Digital Image Forgery Detection Using JPEG Features and Local Noise Discrepancies. *The Scientific World Journal*. 2014, vol. 2014, no. 1, p. 230425. Available from DOI: 10.1155/2014/230425.

29. YE, Shuiming; SUN, Qibin; CHANG, Ee-Chien. Detecting Digital Image Forgeries by Measuring Inconsistencies of Blocking Artifact. In: *2007 IEEE International Conference on Multimedia and Expo*. 2007, pp. 12–15. Available from DOI: 10.1109/ICME.2007.4284574.

30. LIN, Zhouchen; HE, Junfeng; TANG, Xiaoou; TANG, Chi-Keung. Fast, automatic and fine-grained tampered JPEG image detection via DCT coefficient analysis. *Pattern Recognition*. 2009, vol. 42, no. 11, pp. 2492–2501. ISSN 0031-3203. Available from DOI: 10.1016/j.patcog.2009.03.019.

31. REDMON, Joseph; DIVVALA, Santosh; GIRSHICK, Ross; FARHADI, Ali. You Only Look Once: Unified, Real-Time Object Detection. In: *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016, pp. 779–788. Available from DOI: 10.1109/CVPR.2016.91.

32. LONG, Jonathan; SHELHAMER, Evan; DARRELL, Trevor. Fully convolutional networks for semantic segmentation. In: *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2015, pp. 3431–3440. Available from DOI: 10.1109/CVPR.2015.7298965.

33. KRIZHEVSKY, Alex; SUTSKEVER, Ilya; HINTON, Geoffrey E. ImageNet Classification with Deep Convolutional Neural Networks. In: PEREIRA, F.; BURGES, C.J.; BOTTOU, L.; WEINBERGER, K.Q. (eds.). *Advances in Neural Information Processing Systems*. Curran Associates, Inc., 2012, vol. 25. Available also from: https://proceedings.neurips.cc/paper_files/paper/2012/file/c399862d3b9d6b76c8436e924a68c45b-Paper.pdf.

34. LIU, Bo; PUN, Chi-Man. Locating splicing forgery by fully convolutional networks and conditional random field. *Signal Processing: Image Communication*. 2018, vol. 66, pp. 103–112. ISSN 0923-5965. Available from DOI: 10.1016/j.image.2018.04.011.

35. POMARI, Thales; RUPPERT, Guillherme; REZENDE, Edmar; ROCHA, Anderson; CARVALHO, Tiago. Image Splicing Detection Through Illumination Inconsistencies and Deep Learning. In: *2018 25th IEEE International Conference on Image Processing (ICIP)*. 2018, pp. 3788–3792. Available from DOI: 10.1109/ICIP.2018.8451227.

36. WU, Yue; ABDALMAGEED, Wael; NATARAJAN, Premkumar. ManTra-Net: Manipulation Tracing Network for Detection and Localization of Image Forgeries With Anomalous Features. In:

2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). 2019, pp. 9535–9544. Available from DOI: 10.1109/CVPR.2019.00977.

37. WU, Jian; CHANG, Xu; YANG, Tongfeng; FENG, Kai. Blind Forensic Method Based on Convolutional Neural Networks for Image Splicing Detection. In: *2019 IEEE 5th International Conference on Computer and Communications (ICCC)*. 2019, pp. 2014–2018. Available from DOI: 10.1109/ICCC47050.2019.9064258.

38. LI, Jie; WANG, Zuling. Visual Transformer for Image Splicing Localization. In: *2023 4th International Conference on Big Data Artificial Intelligence Software Engineering (ICBASE)*. 2023, pp. 60–65. Available from DOI: 10.1109/ICBASE59196.2023.10303114.

39. DONG, Chengbo; CHEN, Xinru; HU, Ruohan; CAO, Juan; LI, Xirong. MVSS-Net: Multi-View Multi-Scale Supervised Networks for Image Manipulation Detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*. 2023, vol. 45, no. 3, pp. 3539–3553. Available from DOI: 10.1109/TPAMI.2022.3180556.

40. KWON, Myung-Joon; YU, In-Jae; NAM, Seung-Hun; LEE, Heung-Kyu. CAT-Net: Compression Artifact Tracing Network for Detection and Localization of Image Splicing. In: *2021 IEEE Winter Conference on Applications of Computer Vision (WACV)*. 2021, pp. 375–384. Available from DOI: 10.1109/WACV48630.2021.00042.

41. HSU, Yu-feng; CHANG, Shih-fu. Detecting Image Splicing using Geometry Invariants and Camera Characteristics Consistency. In: *2006 IEEE International Conference on Multimedia and Expo*. 2006, pp. 549–552. Available from DOI: 10.1109/ICME.2006.262447.

42. CARVALHO, Tiago José de; RIESS, Christian; ANGELOPOULOU, Elli; PEDRINI, Hélio; REZENDE ROCHA, Anderson de. Exposing Digital Image Forgeries by Illumination Color Classification. *IEEE Transactions on Information Forensics and Security*. 2013, vol. 8, no. 7, pp. 1182–1194. Available from DOI: 10.1109/TIFS.2013.2265677.

43. KNIAZ, Vladimir V.; KNYAZ, Vladimir; REMONDINO, Fabio. The Point Where Reality Meets Fantasy: Mixed Adversarial Generators for Image Splice Detection. In: *Advances in Neural Information Processing Systems*. Curran Associates, Inc., 2019, vol. 32. Available also from: https://proceedings.neurips.cc/paper_files/paper/2019/file/98dce83da57b0395e163467c9dae521b-Paper.pdf.

44. CHRISTLEIN, Vincent; RIESS, Christian; JORDAN, Johannes; RIESS, Corinna; ANGELOPOULOU, Elli. An Evaluation of Popular Copy-Move Forgery Detection Approaches. *IEEE Transactions on Information Forensics and Security*. 2012, vol. 7, no. 6, pp. 1841–1854. ISSN 1556-6021. Available from DOI: 10.1109/tifs.2012.2218597.

45. GUAN, Haiying et al. MFC Datasets: Large-Scale Benchmark Datasets for Media Forensic Challenge Evaluation. In: *2019 IEEE Winter Applications of Computer Vision Workshops (WACVW)*. 2019, pp. 63–72. Available from DOI: 10.1109/WACVW.2019.00018.

46. MAHFOUDI, Gaël; TAJINI, Badr; RETRAINT, Florent; MORAIN-NICOLIER, Frédéric; DUGELAY, Jean Luc; PIC, Marc. DEFACTO: Image and Face Manipulation Dataset. In: *2019 27th European Signal Processing Conference (EUSIPCO)*. 2019, pp. 1–5. Available from DOI: 10.23919/EUSIPCO.2019.8903181.

47. TAHIR, Eren; BAL, Mert. *Deep Image Composition Meets Image Forgery*. 2024. Available from DOI: 10.48550/arXiv.2404.02897.

48. HUH, Minyoung; LIU, Andrew; OWENS, Andrew; EFROS, Alexei A. *Fighting Fake News: Image Splice Detection via Learned Self-Consistency*. 2018. Available from DOI: 10.48550/arXiv.1805.04096.

49. KWON, Myung-Joon; NAM, Seung-Hun; YU, In-Jae; LEE, Heung-Kyu; KIM, Changick. Learning JPEG Compression Artifacts for Image Manipulation Detection and Localization. *International Journal of Computer Vision*. 2022, vol. 130, no. 8, pp. 1875–1895. Available from DOI: 10.1007/s11263-022-01617-5.

50. LIN, Tsung-Yi; MAIRE, Michael; BELONGIE, Serge; BOURDEV, Lubomir; GIRSHICK, Ross; HAYS, James; PERONA, Pietro; RAMANAN, Deva; ZITNICK, C. Lawrence; DOLLÁR, Piotr. *Microsoft COCO: Common Objects in Context*. 2015. Available from DOI: 10.48550/arXiv.1405.0312.

51. KADAM, Kalyani Dhananjay; AHIRRAO, Swati; KOTECHA, Ketan. Multiple Image Splicing Dataset (MISD): A Dataset for Multiple Splicing. *Data*. 2021, vol. 6, no. 10. ISSN 2306-5729. Available from DOI: 10.3390/data6100102.

52. LIU, Ze; HU, Han; LIN, Yutong; YAO, Zhuliang; XIE, Zhenda; WEI, Yixuan; NING, Jia; CAO, Yue; ZHANG, Zheng; DONG, Li; WEI, Furu; GUO, Baining. *Swin Transformer V2: Scaling Up Capacity and Resolution*. 2022. Available from DOI: 10.48550/arXiv.2111.09883.

53. WIGHTMAN, Ross. *PyTorch Image Models* [https://github.com/huggingface/pytorch-image-models]. GitHub, 2019. Available from DOI: 10.5281/zenodo.4414861.

54. DENG, Jia; DONG, Wei; SOCHER, Richard; LI, Li-Jia; LI, Kai; FEI-FEI, Li. ImageNet: A large-scale hierarchical image database. In: *2009 IEEE Conference on Computer Vision and Pattern Recognition*. 2009, pp. 248–255. Available from DOI: 10.1109/CVPR.2009.5206848.

55. BURGER, Wilhelm; BURGE, Mark J. *Digital Image Processing: An Algorithmic Introduction*. Springer Cham, 2022. ISBN 978-3-031-05744-1. Available from DOI: 10.1007/978-3-031-05744-1.

56. HU, Zhentao; LIU, Fuyi; ZHANG, Shengjia; SU, Yujie. Image splicing localization based on dual-stream spatial-noise learning and hierarchical frequency fusion. *Applied Soft Computing*. 2026, vol. 186, p. 114121. ISSN 1568-4946. Available from DOI: 10.1016/j.asoc.2025.114121.

57. KRAWETZ, Neal; SOLUTIONS, Hacker Factor. A picture's worth. *Hacker Factor Solutions*. 2007, vol. 6, no. 2, p. 2.

58. FERRARA, Pasquale; BIANCHI, Tiziano; DE ROSA, Alessia; PIVA, Alessandro. Image Forgery Localization via Fine-Grained Analysis of CFA Artifacts. *IEEE Transactions on Information Forensics and Security*. 2012, vol. 7, no. 5, pp. 1566–1577. Available from DOI: 10.1109/TIFS.2012.2202227.

59. LIU, Xiaohong; LIU, Yaojie; CHEN, Jun; LIU, Xiaoming. PSCC-Net: Progressive Spatio-Channel Correlation Network for Image Manipulation Detection and Localization. *IEEE Transactions on Circuits and Systems for Video Technology*. 2022, vol. 32, no. 11, pp. 7505–7517. Available from DOI: 10.1109/TCSVT.2022.3189545.

60. NILOY, Fahim Faisal; KUMAR BHAUMIK, Kishor; WOO, Simon S. CFL-Net: Image Forgery Localization Using Contrastive Learning. In: *2023 IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*. 2023, pp. 4631–4640. Available from DOI: 10.1109/WACV56688.2023.00462.

61. GUO, Kun; ZHU, Haochen; CAO, Gang. Effective Image Tampering Localization Via Enhanced Transformer and Co-Attention Fusion. In: *ICASSP 2024 - 2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 2024, pp. 4895–4899. Available from DOI: 10.1109/ICASSP48485.2024.10446332.

62. BAI, Ningning; WANG, Xiaofeng; HAN, Ruidong; HOU, Jianpeng; WANG, Yihang; PANG, Shanmin. PIM-Net: Progressive Inconsistency Mining Network for image manipulation localization. *Pattern Recognition*. 2025, vol. 159, p. 111136. ISSN 0031-3203. Available from DOI: 10.1016/j.patcog.2024.111136.

A An appendix

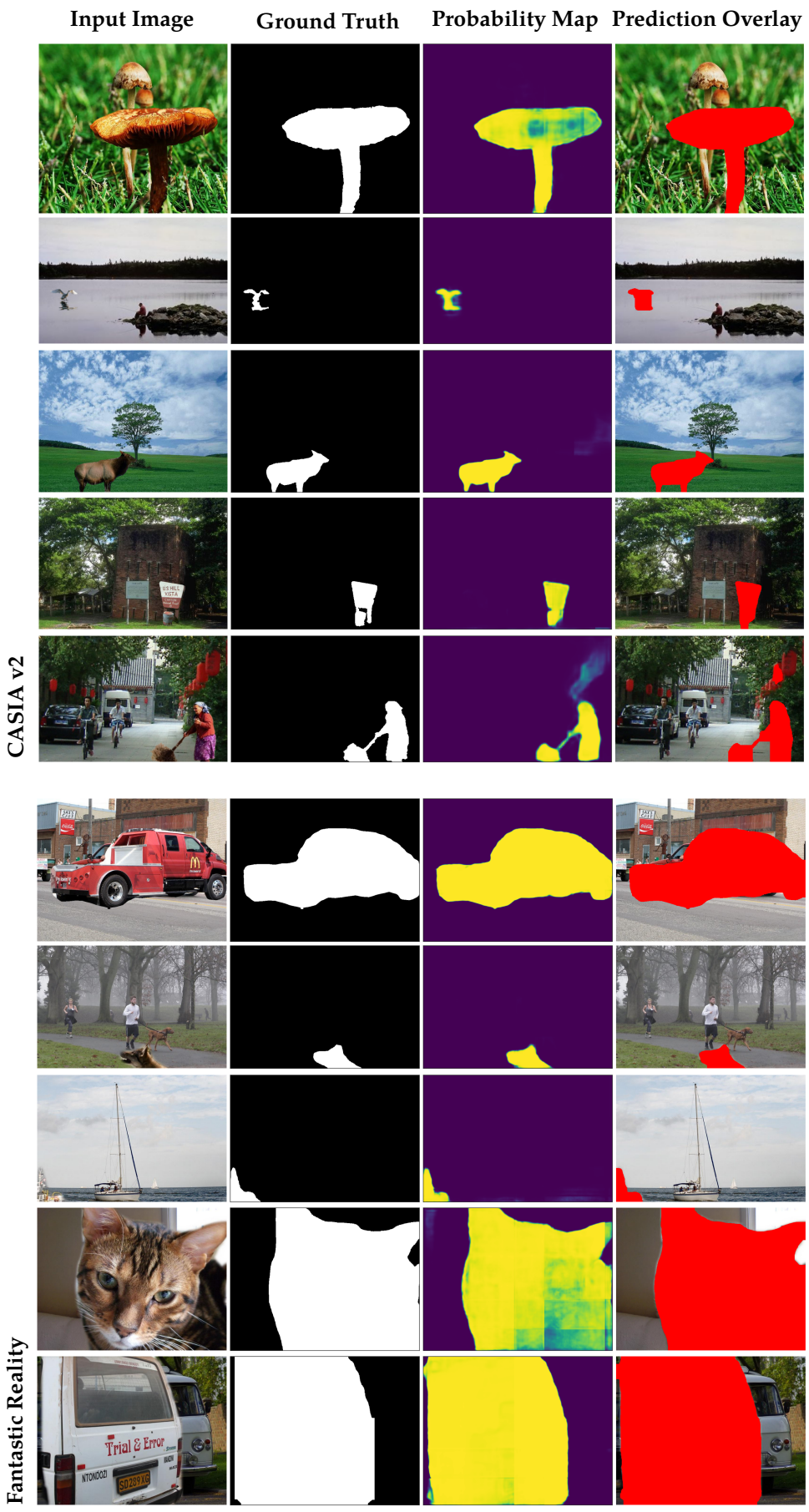


Figure A.1: Qualitative splicing detection results for five samples per dataset (Part 1). From left to right: input image, ground-truth mask, predicted probability map, and predicted splicing mask overlay.

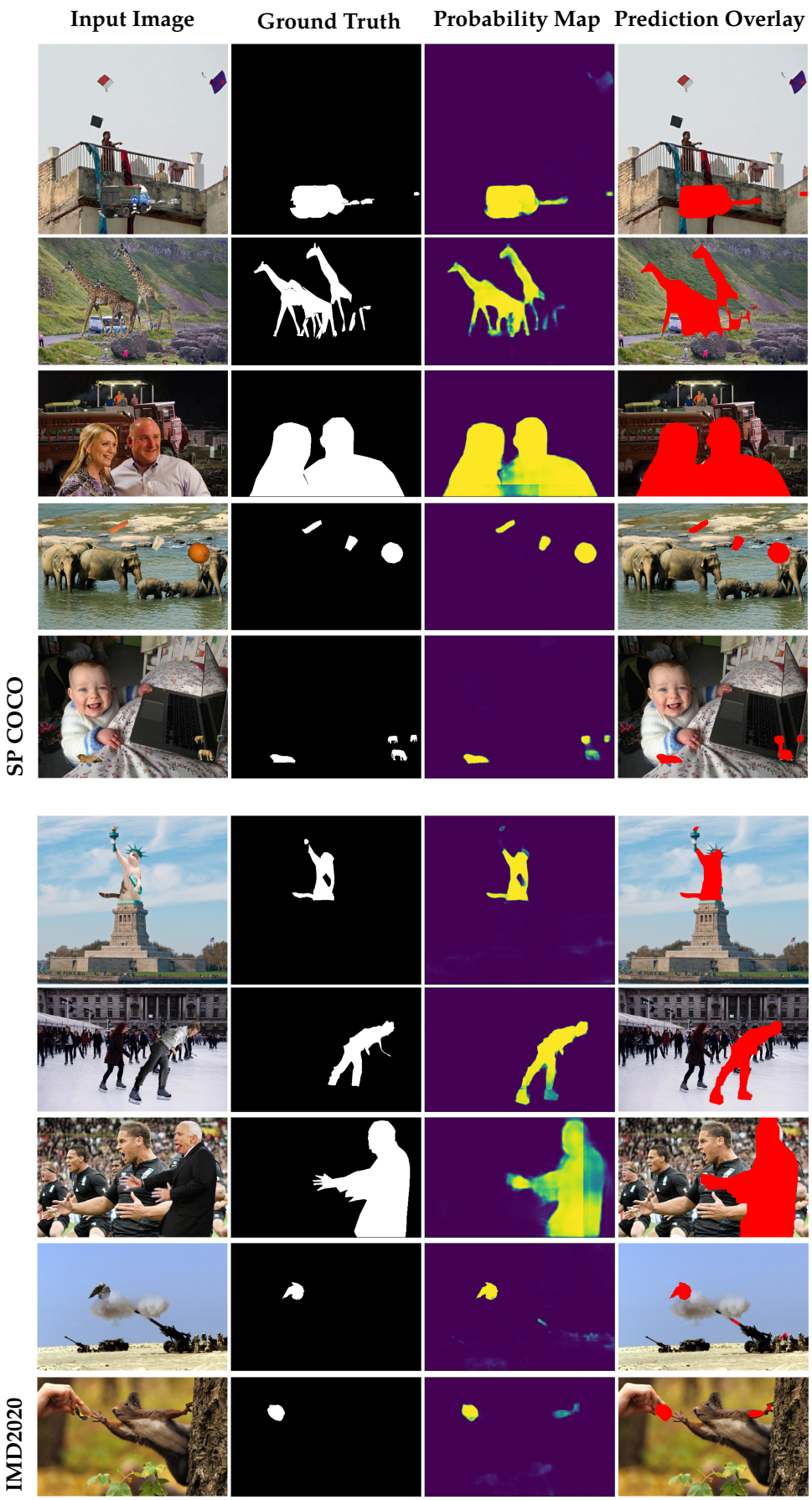


Figure A.2: Qualitative splicing detection results for five samples per dataset (Part 2). From left to right: input image, ground-truth mask, predicted probability map, and predicted splicing mask overlay.

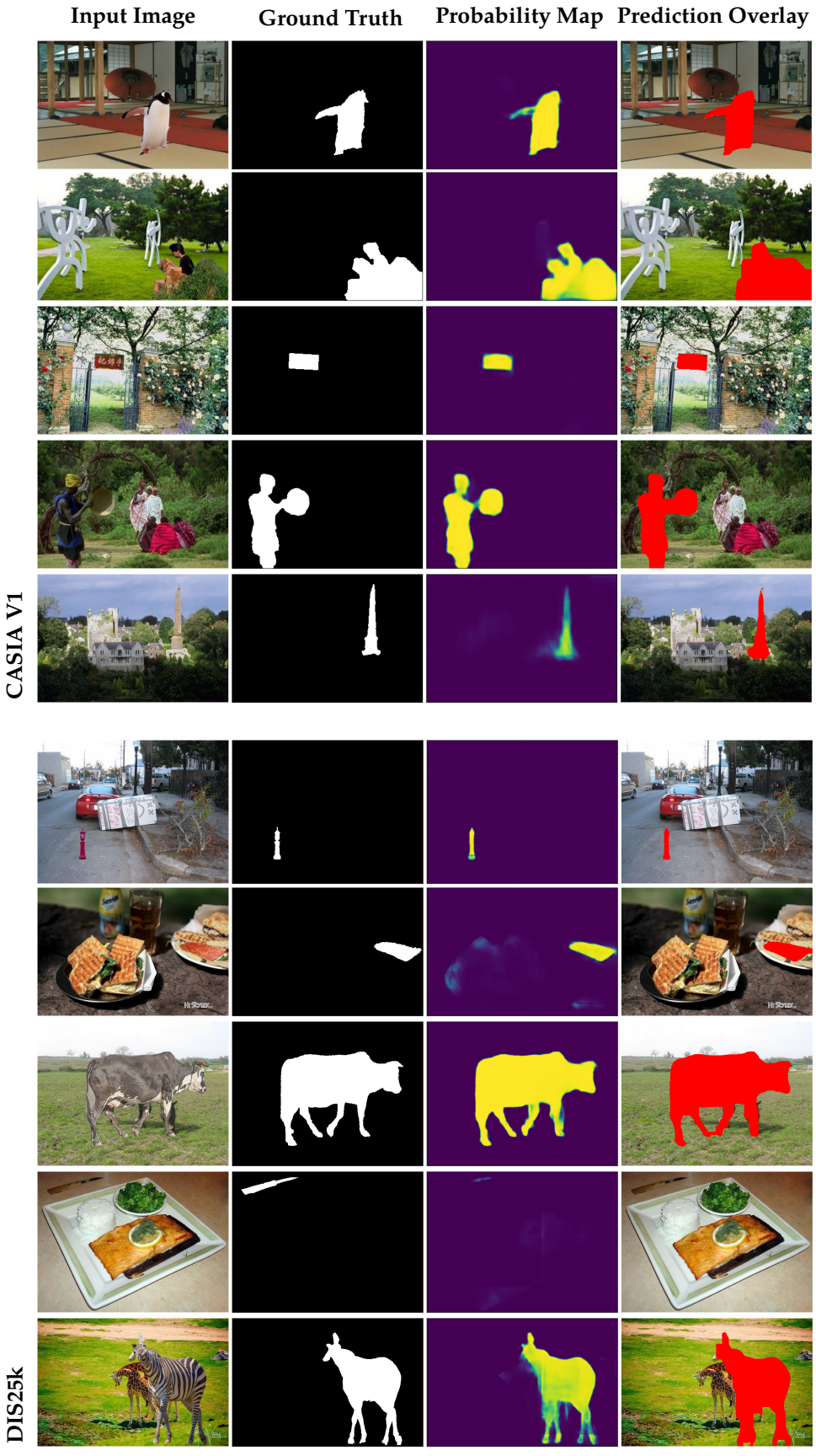


Figure A.3: Qualitative splicing detection results for five samples per dataset (Part 3). From left to right: input image, ground-truth mask, predicted probability map, and predicted splicing mask overlay.

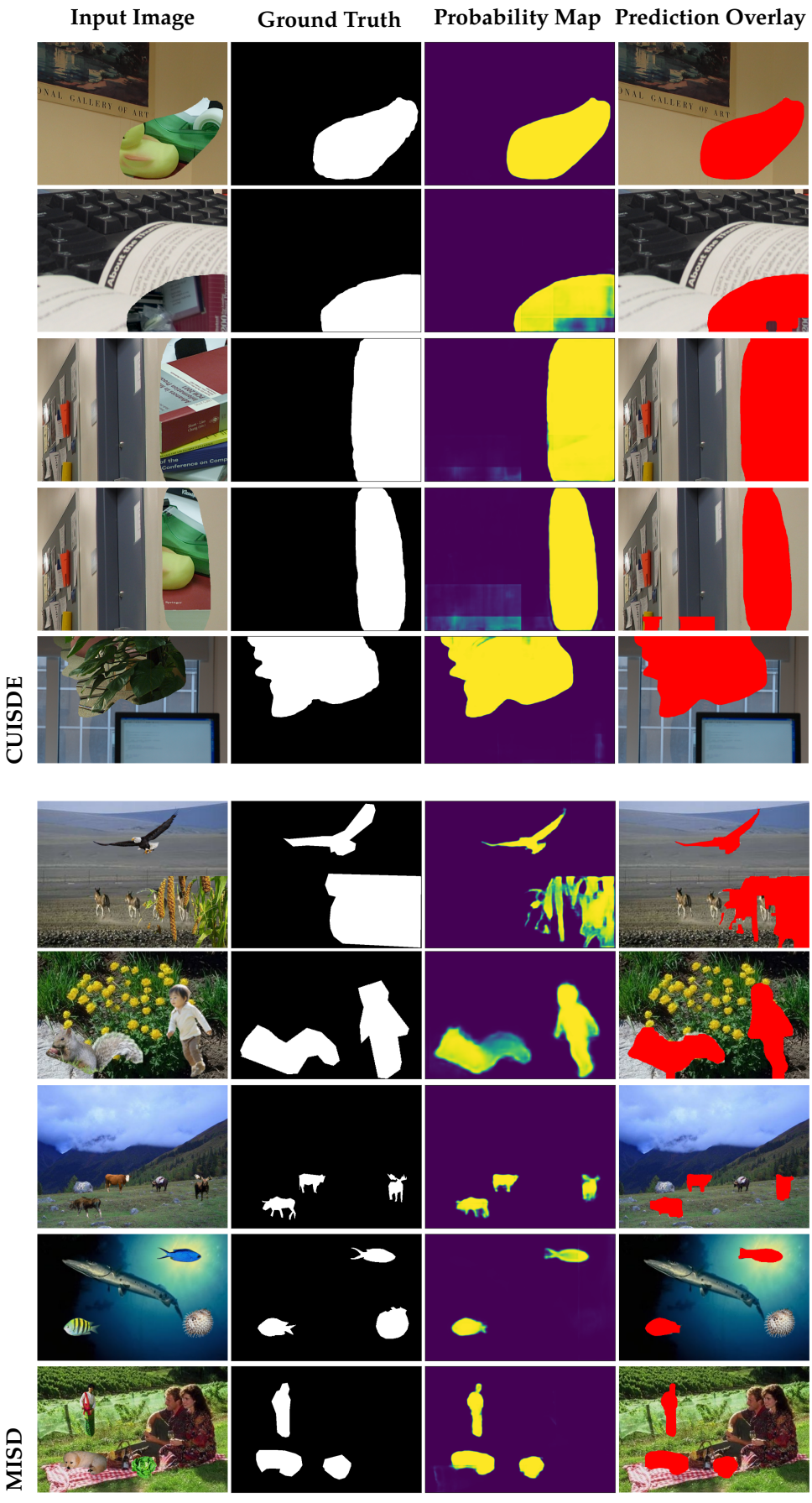


Figure A.4: Qualitative splicing detection results for five samples per dataset (Part 4). From left to right: input image, ground-truth mask, predicted probability map, and predicted splicing mask overlay.



Figure A.5: Qualitative splicing detection results for five samples per dataset (Part 5). From left to right: input image, ground-truth mask, predicted probability map, and predicted splicing mask overlay.