Automatic Stamp Detection via Semantic Segmentation and Post-Processing

Abstract

In this study, I address the problem of automatic stamp detection in scanned document images by combining semantic segmentation with post-processing techniques. I implemented a U-Net-based semantic segmentation model trained on annotated document images to identify stamp regions. To improve output quality, I incorporated post-processing techniques such as density-based clustering (DBSCAN) and color-based masking, producing extractions ready for reuse. The final model produced high-quality transparent stamp extractions with sufficiently high precision.

1. Introduction

In the age of smart-literally-everything, it is quite surprising how paperwork still remains a major part of everyday life for us, and processing it is often slow and manual. Thus, there is a growing need for smart tools that can automatically extract useful information from documents. Examples of such are official stamps, which often indicate approval and authenticity. However, stamps can be difficult to detect due to their varying size, color, placement, and quality.

Initially, my idea for the project was to automate chessboard state recognition by converting their images into FEN strings, i.e. computer-readable format. However, throughout the past few months I found myself frequently working with scanned documents as part of family and personal bureaucracy. While managing this paperwork, I observed that many of these documents included stamps, sometimes faint or tilted, whose presence or content played an important role in validation. So, a new question arose: could deep learning be used to detect and extract stamps from such documents given their variability? I realized this topic switch seems more relevant, applicable, and, frankly, manageable given time constraints; therefore I proceeded.

Recent methods in stamp detections involve deep learning, particularly semantic segmentation models like U-Net, which showed strong performance in extracting non-textual elements from documents, including stamps, signatures, and logos (Lopes et al., 2021). Object detectors like YOLO have also been used effectively for locating stamps (Bento et al., 2025), but they tend to produce boxes, not precise stamp masks. Post-processing methods such as DBSCAN clustering (Kezzoula & Gaceb, 2024) help refine segmentation outputs and reduce false positives. Building on these, the project aims to combine segmentation with spatial and color-based post-processing to develop a robust method for automatic stamp extraction in real-world scanned documents.

Research Question: To what extent can we accurately detect and extract stamps from scanned document images, considering variability in size, shape, and color?

2. Data and Exploratory Analysis

The dataset used for this project was obtained from Roboflow, consisting of almost 5800 high-resolution document scans with pixel-level binary masks identifying stamp regions. Each mask used a two-level encoding scheme, where background pixels were labeled with 0 and stamp pixels - 255.

To better understand the nature of the stamps, I performed a 3D color space analysis by plotting RGB values of stamp pixels across multiple samples. This revealed a high degree of overlap between stamp colors and other front elements, such as colored text and logos. I concluded that color-based filtering alone would not be sufficient for reliable detection. I also experimented with Hough Circle Transform, assuming that stamps are typically circular; however, this method frequently misidentified page textures, seals, and margins as stamps, so it was scrapped.

3. Methods

3.1 Semantic vs. Instance Segmentation

I treated stamp detection as a semantic segmentation task: the goal was to classify each pixel as "stamp" or "non-stamp." Instance segmentation, which differentiates between multiple instances of

the same object class, was not necessary, since most documents contain only a single stamp. Semantic segmentation simplifies both training and inference in this context.

3.2 Thresholding

Initially, I attempted to use simple color thresholds (e.g., high blue, low red) to isolate stamp regions. However, this approach failed to generalize due to significant background noise, such as colored text, image artifacts, and lighting variations. These distractions frequently created false positives across the image.

To fix this, I decided to segment the stamp area first using a neural network, and only then apply thresholding inside the detected zone. This two-stage method reduced the impact of irrelevant areas and improved the accuracy of color-based separation.

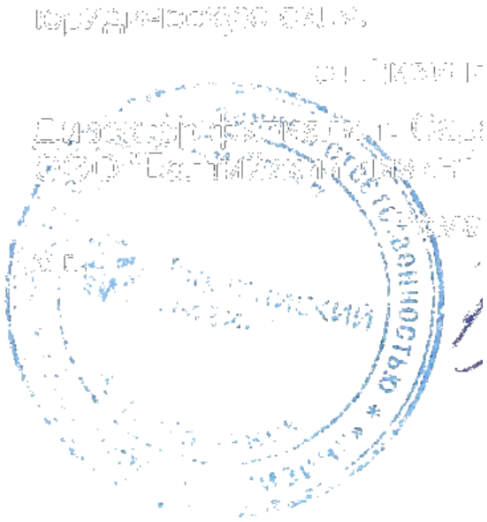

Figure 1: Thresholding vs. background noise.

3.3 3D Color Distribution Visualization

To better understand the range of stamp colors, I created 3D color scatter plots. Each pixel's (R, G, B) value was plotted in 3D space, and the image was downsampled for performance. This helped me observe how color clusters were distributed and provided empirical evidence for choosing thresholds. I confirmed that many irrelevant background elements overlapped in color with stamp pixels, highlighting the need to apply color filtering only after semantic segmentation.

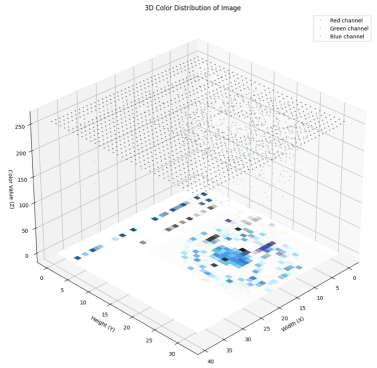

Figure 2: 3D RGB color distribution of a stamp-containing image.

3.4 Hough Circle Transform

I also tested the classical Hough Circle Transform, assuming that stamps are generally circular. However, it performed poorly. The method identified many false-positive circles in textured backgrounds or near document margins and missed light or irregularly shaped stamps. Due to this, I abandoned the Hough method in favor of a learning-based one.

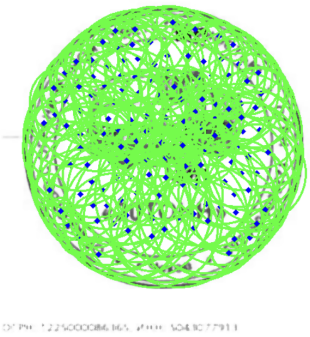

Figure 3: Hough Transform with circular over-detection.

3.5 Deep Learning: U-Net Semantic Segmentation

To segment stamps accurately, I implemented a U-Net model using the `segmentation_models_pytorch` library.

Architecture Overview: U-Net uses an encoder-decoder structure: The encoder (ResNet34) captures contextual features at multiple scales. The decoder upsamples these features and incorporates spatial information using skip connections. The final output is a one-channel binary mask representing the stamp region.

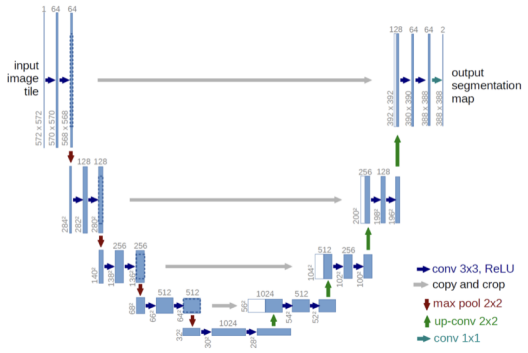

Figure 4: U-Net architecture used for semantic segmentation. Adapted from Ronneberger et al. (2021)

Training	**setup:**
Framework	PyTorch
Input Size	256x256 RGB images
Loss Function	Dice Loss (effective for imbalanced data)
Optimizer	Adam (lr = 0.001)
Device	CUDA-enabled GPU

Data Augmentation: To increase model robustness, I applied augmentations such as random horizontal flips, rotations up to 30 degrees, brightness and contrast adjustments, Gaussian blur

Training Outcome: After training, the model reliably learned to detect both the circular outline and internal structure of stamps. It performed well even with partially faded stamps, enabling the extraction of binary masks.

3.6 Post-Processing: DBSCAN and Conditional Thresholding

Once a binary mask is predicted by U-Net, I apply the following post-processing steps:

1. DBSCAN Clustering: I extracted coordinates of all mask-positive pixels, applied DBSCAN (eps=1.5, min_samples=1) to find connected regions, and selected the largest cluster as the main stamp area. This allowed us to identify and retain the primary stamp area while filtering out small, isolated false positives. DBSCAN is advantageous because it identifies arbitrarily shaped connected areas without requiring a predefined number of clusters. It also filters out isolated noise pixels, which are common in segmentation outputs.

2. Color-Based Filtering (Localized): With the main stamp region isolated, I applied threshold-based filtering to its pixels: blue channel > defined threshold, red channel < defined threshold, green optionally excluded or included. I filtered pixels based on empirical color thresholds, retaining those with high blue channel intensity and low red channel intensity. This filtering improved the visual quality of the output by excluding non-stamp elements that may have been included in the segmentation mask.

3. Transparent RGBA Output: Finally, I cropped the detected region and converted it into a transparent RGBA image, where only pixels matching the color condition were retained with non-zero alpha.

4 Results

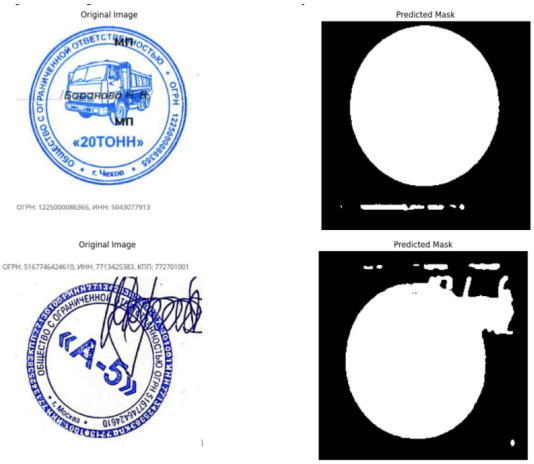

Figure 5: Predicted binary stamp masks on test examples.

The method demonstrated high accuracy in extracting stamps over a diverse set of test images. Compared to classical threshold approaches, the U-Net model consistently produced cleaner and more complete results. Post-processing with DBSCAN then reduced false detections, and the localized filtering strategy outputted results that were practically useful.

Example results include documents with problematic lighting conditions, overlaps of stamps with printed text, and irregular outlines. In each case, the model worked well on isolating the relevant region and producing a transparent extracted version with minimal background involvement. The final pipeline offers an interpretable solution to stamp detection in various set ups.

5. Discussion

This project demonstrates that deep learning in collaboration with targeted post-processing, can provide a strong alternative to classical image processing for the task of stamp detection. The

U-Net model effectively learned to identify features associated with stamps, while the addition of DBSCAN and color filtering added precision to the outputs.

Yet, several limitations need to be addressed. The dataset used was relatively small, which may affect the generalizability of the model to more stamp styles. In particular, stamps that are multicolored or with extremely low contrast were harder to detect. Additionally, model performance may worsen on low-resolution or compressed scans, where boundaries are not so distinct.

6. Conclusions and Future Work

In conclusion, the proposed approach successfully combines semantic segmentation with clustering and filtering techniques to perform automatic stamp detection in document images. Future directions include expanding the training dataset with more diverse stamps, and exploring transformer-based segmentation models for finer boundary detection. Also, it could be promising to involve synthetic stamp generation to augment training data and simulate edge cases that are not present in the current dataset. Finally, it would be relevant to develop a user-friendly tool for batch-processing large document archives.

7. Reference list

- Bento, L., da Silva, M. R., & Sousa, A. J. (2025). Performance Evaluation of YOLOv8, YOLOv9, YOLOv10, and YOLOv11 for Stamp Detection in Scanned Documents. *Applied Sciences, 15*(6), 3154. https://doi.org/10.3390/app15063154

- Kezzoula, Z., & Gaceb, D. (2025). A new fast DBSCAN using dual-space analysis and colour integral volume for document image segmentation. *International Journal of Computational Vision and Robotics, 15*(3), 395–416. https://doi.org/10.1504/IJCVR.2025.146298

- Lopes, R. G., da Silva, F. N., & Almeida, J. P. (2021). A Handwritten Signature Segmentation Approach for Multi-resolution and Complex Documents Acquired by Multiple Sources. Springer. https://doi.org/10.1007/978-3-030-86334-0_21

- Roboflow. (n.d.). Semantic Stamp Segmentation Dataset [Dataset]. Roboflow Universe. https://universe.roboflow.com/semseg-lx7i4/semantic-uk7u3

- Ronneberger, O., Fischer, P., and Brox, T. (2021). Brain Tumor Classification via UNET Architecture of CNN Technique. Springer. https://doi.org/10.1007/978-3-031-15784-4_2