Optical Chemical Structure Recognition using Traditional Computer Vision Techniques

Jeffin Joffy P, Yash Rohan, Janav Bhasin
Dept. of Computer Science and Engineering
Manipal Institute of Technology
Manipal, India
jeffinjoffyp123@gmail.com, yashrohan001@gmail.com, janavbhasin27@gmail.com
+91 8281596331, +91 9576114110, +91 7678673737

***Abstract*—Optical Chemical Structure Recognition (OCSR) converts chemical structure images into digital representations, supporting cheminformatics by enabling the automated reading of structural diagrams. This paper introduces a systematic approach to OCSR using image processing techniques to detect atoms and bonds within chemical images, coupled with Optical Character Recognition (OCR) for extracting textual annotations. We employ OpenCV for preprocessing, edge detection, and feature extraction, enabling detection of atoms as circular regions and bonds as linear connections. The Hough Transform is utilized to identify circular and linear elements, with custom functions developed to filter redundant lines and adjust bond lengths to visible structure. Text annotations are extracted using Tesseract OCR, optimized for chemical symbols and numeric annotations. This method is validated through a comprehensive pipeline evaluation, and the resulting structure is organized into a graph format suitable for cheminformatics applications. The approach demonstrates promising results in converting complex chemical imagery into structured, machine-readable data, paving the way for future advancements in chemical data analysis.**

***Index Terms*—Optical Chemical Structure Recognition (OCSR), cheminformatics, image processing, OpenCV, Tesseract OCR, Hough Transform, feature extraction, chemical data digitization, SMILES conversion.**

I. Introduction

A. Background

Optical Chemical Structure Recognition (OCSR) is a vital area of research that addresses the challenge of converting graphical representations of chemical structures into machine-readable formats. This task has gained prominence due to the vast amount of chemical data published annually in image formats, which are not directly usable for computational analysis. Traditional methods of OCSR have relied heavily on rule-based approaches, which involve manually crafted algorithms for detecting and interpreting molecular components such as atoms and bonds from images. However, these methods often struggle with inaccuracies due to their dependence on specific rules and the complexities inherent in chemical structures, particularly when dealing with variations in drawing styles or imperfections in hand-drawn structures [1], [4].

Recent advancements in deep learning have led to the development of more sophisticated OCSR techniques that leverage neural networks for feature extraction and recognition tasks. For instance, the introduction of models like the Swin Transformer has significantly improved the ability to capture both local and global features of chemical images, enhancing recognition accuracy compared to earlier convolutional neural network (CNN)-based methods [3], [5]. The SwinOCSR model, for example, has demonstrated an impressive accuracy rate of 98.58% on a specialized dataset, showcasing the potential of deep learning to overcome some limitations faced by traditional OCSR systems [3].

Moreover, the emergence of multi-path architectures, such as the Multi-path Optical Chemical Structure Recognition (MPOCSR) model, represents a further evolution in this field. MPOCSR combines the strengths of convolutional networks and Vision Transformers to provide a more comprehensive representation of chemical structures by integrating multi-scale information [1]. This approach addresses two critical challenges: the inherent class imbalance in molecular elements and the need for robust feature extraction methods that can handle variations in image quality and structure complexity.

The significance of OCSR extends beyond mere data extraction; it plays a crucial role in facilitating access to chemical information across various domains including pharmaceuticals, chemical biology, and materials science. By converting chemical structures into standardized formats like SMILES or SELFIES, OCSR enables better data sharing and integration into databases, thus enhancing research efficiency and fostering innovation in related fields [2], [4]. As the volume of chemical literature continues to grow, improving OCSR methodologies will be essential for keeping pace with data demands and ensuring that valuable scientific information is readily accessible for computational analysis and discovery.

B. Importance

Automating OCSR provides significant benefits by:

- Reducing the time and effort involved in manual chemical data entry,
- Enabling rapid digitization of historical chemical records and research publications,
- Supporting data interoperability across platforms and research institutions.

OCSR systems transform chemical images into structured formats, making them readily accessible for computational

workflows. This capability is especially beneficial for collaborative research in fields such as medicinal chemistry, toxicology, and environmental sciences, where accurate and scalable data access is vital.

C. Research Gaps

Recent advancements in Optical Chemical Structure Recognition (OCSR) have predominantly centered on machine learning (ML) and deep learning approaches, particularly those utilizing transformer architectures and other neural network designs. These ML-driven methodologies have demonstrated significant improvements in accuracy; however, they also present several challenges that warrant further exploration.

1) Data Requirements: One of the primary challenges associated with ML-based OCSR systems is their dependency on large, labeled datasets for training. High-quality annotated data is often scarce, especially for niche chemical structures or hand-drawn representations. The lack of accessible datasets can hinder the development and generalization of these models, limiting their applicability in real-world scenarios where data may not be readily available or sufficiently diverse [3].

2) Computational Resources: Deep learning approaches typically require substantial computational resources, including specialized hardware such as GPUs or TPUs, to train complex models effectively. This demand can be a barrier for many researchers or institutions with limited access to such resources, thereby constraining the widespread adoption of advanced OCSR techniques [4].

3) Generalizability: Another significant concern is the generalizability of transformer-based models across diverse chemical imagery. These models can struggle with variations in drawing styles and the quality of input images, particularly when faced with novel or low-quality scans. The inherent variability in chemical depictions poses a challenge for existing OCSR software, which may not perform consistently across different datasets or types of images.

In contrast to these ML-driven methods, traditional rule-based approaches—such as edge detection, Hough transforms, and Optical Character Recognition (OCR)—remain relatively underexplored within the OCSR domain. These methods can offer efficient, data-agnostic solutions that require fewer resources, making them particularly valuable in contexts where high-quality labeled data is limited.

Given these considerations, there is a growing interest in:

- **Exploring Hybrid Methods:** Combining ML techniques with traditional rule-based approaches could enhance robustness and flexibility in OCSR systems. Such hybrid methods may leverage the strengths of both paradigms to improve overall performance while mitigating their individual weaknesses.
- **Evaluating Rule-Based Systems:** Investigating the potential of rule-based OCSR systems could provide valuable insights into their effectiveness in scenarios where data or computational resources are constrained. This evaluation could help identify specific applications where traditional methods might outperform more complex ML approaches.
- **Developing Standardized Workflows:** Establishing standardized rule-based OCSR workflows could complement ML-heavy approaches, offering researchers reliable alternatives that do not depend on extensive computational resources or large datasets.

By addressing these research gaps, future studies can contribute to a more balanced understanding of OCSR methodologies, ultimately enhancing the field's ability to process and interpret chemical information from a variety of sources.

D. Objectives

This research proposes a structured, rule-based methodology for OCSR, emphasizing traditional image processing techniques and OCR. The main goals are:

- To develop an OCSR approach that relies on edge detection, Hough transforms, and OCR to recognize atoms and bonds in chemical images.
- To provide an alternative to ML-based OCSR by building a pipeline suitable for limited-data environments.
- To assess the feasibility of rule-based methods for accurately extracting chemical structures, bridging the gap between current data-intensive methods and resource-constrained applications.

II. Related Work

Optical Chemical Structure Recognition (OCSR) has emerged as a critical research area due to the increasing volume of chemical information presented in graphical formats across scientific literature. The primary goal of OCSR is to convert these graphical representations into machine-readable formats, facilitating data analysis and integration within computational chemistry. This review synthesizes recent advancements in OCSR methodologies, highlighting the transition from traditional rule-based approaches to modern deep learning techniques.

A. Historical Context and Traditional Approaches

Historically, OCSR methods have relied on rule-based systems that involve manually crafted algorithms to detect and interpret the various components of chemical structures, such as atoms and bonds. Early systems like Kekule, OROCS, and OSRA utilized vectorization techniques to convert images into structured data [1]. However, these methods often struggled with accuracy due to their reliance on predefined rules, which could not accommodate the variability in chemical drawings or imperfections in images. For instance, slight deformations in chemical structures could significantly impair recognition performance [2].

B. Emergence of Deep Learning Techniques

The advent of deep learning has revolutionized OCSR by providing more robust frameworks for feature extraction and recognition. Recent studies have introduced models that leverage convolutional neural networks (CNNs) and transformers to

enhance the accuracy of chemical structure recognition. One notable example is the SwinOCSR model, which employs a Swin Transformer as its backbone. This model achieved an impressive accuracy rate of 98.58% by effectively addressing issues related to token imbalance in the representation of chemical structures through focal loss [3].

Another significant advancement is the Multi-path Optical Chemical Structure Recognition (MPOCSR) model, which combines the strengths of CNNs and Vision Transformers. MPOCSR utilizes a multi-path Vision Transformer (MPViT) to capture both local and global features of chemical images, resulting in improved recognition capabilities compared to previous methods like SwinOCSR [2]. The incorporation of a class-balanced loss function further enhances its performance by addressing the issue of imbalanced molecular element frequency.

C. Current Challenges and Future Directions

Despite these advancements, challenges remain in the field of OCSR. Many existing deep learning models still struggle with generalization across diverse datasets, particularly when dealing with sparse or costly-to-generate data such as hand-drawn molecular images [1]. Moreover, while deep learning methods have shown promise, there is still a need for more extensive benchmarking against various datasets to validate their effectiveness across different modalities and chemistry categories [4], [5].

Recent comparisons of multiple OCSR tools have highlighted varying strengths among different methodologies when applied to independent test sets derived from patents and academic publications. These studies underscore the necessity for ongoing development and refinement of OCSR systems to improve precision and recall rates in real-world applications [4], [5].

III. Methodology

A. Dataset and Preprocessing

1) Dataset: Our dataset consists of 360 images, each depicting one of nine distinct hand-drawn molecular structures, with 40 images per structure. These illustrations were created on standard white printer paper using a fine-tip black Sharpie marker. Images were captured using an iPhone 6 camera at an initial resolution of 3264 x 2448 in RGB format. The images were subsequently reduced in size to 400 x 300 pixels and converted to grayscale through bilinear interpolation. To maintain consistency, all images were captured under the same lighting conditions, and three individuals contributed to the drawings, ensuring that the model wouldn't become biased toward a particular drawing style. For training purposes, 45 images (five per molecule) were set aside.

2) Preprocessing: Each image underwent preprocessing by applying a binarization technique with a 40% threshold, converting grayscale images to binary form for analysis. No additional preprocessing steps were used, allowing the hand-drawn details to remain largely intact in the dataset.

3) Image Downsampling: To prepare the images for analysis, a custom script was developed to downsample each high-resolution image to a resolution of 400 x 300 pixels. Using the OpenCV library in Python, the script resizes each image within predefined file paths and stores the downsampled images in a separate directory. This automated process enables efficient image resizing while maintaining data organization.

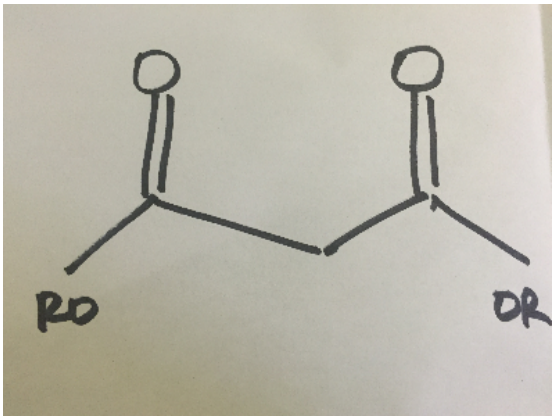


Fig. 1. Example image of a downsampled molecule structure after preprocessing

B. Template Creation Methodology

1) Template Preparation: The template preparation process involves defining and organizing a set of template images for various molecular structures. These templates, stored in predefined directory paths, correspond to different molecular components such as hydroxyl (OH), alkoxy (OR), and single atoms like oxygen (O), hydrogen (H), nitrogen (N), and a root group (R). The dataset paths and naming conventions are also organized to ensure consistency across different molecular structures.

2) Deskewing and Cropping: Images undergo deskewing and cropping as part of the preprocessing. The deskewing function adjusts any image skewness by calculating image moments, realigning the content when necessary, and preserving uniformity across templates. The cropping function identifies the relevant regions by examining non-zero pixels in the image, crops out the bounding area of the drawn molecule, and then resizes the cropped area to a standard size with optional padding.

3) Template Creation: The function `crop_and_make_templates` processes each image within the specified directories, thresholds the image to enhance contrast, crops it to focus on the molecular structure, and resizes it. Each processed template image is stored for later stacking.

4) Template Stacking: The template stacking process combines individual template images into a single composite template. The `stack_templates` function performs this by applying Gaussian blurring to each template and averaging the results. This composite image provides a smoothed, general

representation of each molecular structure, stored in each respective directory as `combined.png`.

5) Visualization and Storage: The final composite template image for each molecular component is displayed for visual confirmation and saved to the designated directory. This final template serves as a reference for the molecule's overall structure in subsequent analysis.

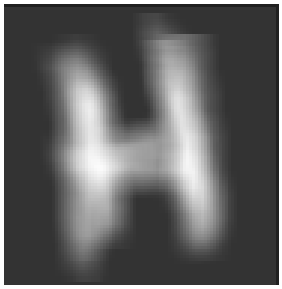


Fig. 2. Example of a composite template image for molecular structure after preprocessing

C. Template Matching

1) Template Matching and Scaling: The template matching process involves reading the target image and each template, applying binary inversion thresholding, and adding a border around the image. Gaussian blurring is applied to smooth the edges, ensuring consistent detection.

2) Multi-Scale Matching: Templates are scaled across multiple levels, from 30% to 100% of their original size. For each scale, template matching is performed using OpenCV's `matchTemplate` function with a normalized correlation coefficient. Locations with similarity above a threshold are recorded as bounding boxes around detected structures.

3) Non-Maximal Suppression: After detecting multiple bounding boxes, non-maximal suppression is applied to avoid overlapping boxes. This suppression keeps only the highest-scoring bounding boxes when overlap occurs, improving the precision of the detections.

4) Template Matching Across Images: The template matching is executed on each image by iterating through all templates. Detected bounding boxes are stored in a dictionary, indexed by template names, and non-maximal suppression is applied once again to refine the results.

5) Visualization of Detected Templates: To verify the accuracy of template detection, the bounding boxes can be overlaid on the original image with distinct colors for each template. If enabled, this visualization is displayed to the user, who can manually confirm its accuracy.

6) Ground Truth Comparison and Performance Evaluation: Ground truth values for each structure are imported from an external text file, which specifies expected counts of each template within each image. These values enable performance evaluation through comparison with actual detections.

7) Precision and Recall Calculation: The comparison results in true positives, false positives, and false negatives for each template type, allowing calculation of precision and recall scores. Precision measures the proportion of correctly detected structures out of all detections, while recall indicates the proportion of true structures detected. These metrics help assess the accuracy of the template matching process.

8) Saving Results: Each template detection is saved as a serialized `pickle` file, storing the bounding boxes and detected structures for each image. This file enables easy access and retrieval of detection data for subsequent analysis or processing.

9) Final Evaluation Across Images: The template matching process is repeated across all images in the dataset, with precision, recall, true positives, false positives, and false negatives recorded for each image. Additionally, the overall accuracy, representing the proportion of images correctly processed, is calculated and reported for comprehensive performance evaluation.

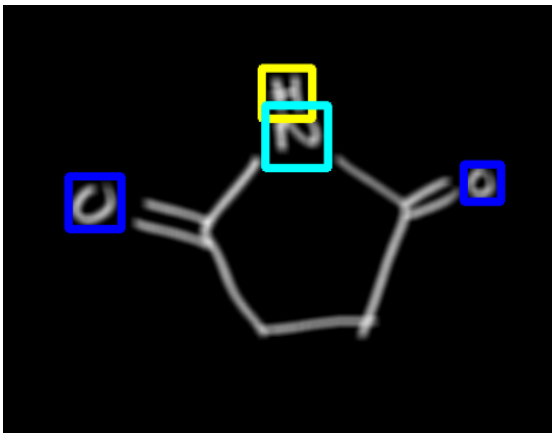


Fig. 3. Example image showing template matching results with bounding boxes for each molecular structure

D. Corner Detection

1) Corner Detection Process: The corner detection process consists of several stages: thresholding, masking, blurring, and corner detection, which are executed sequentially to identify key points in each molecular structure.

2) Image Thresholding and Masking: The grayscale image undergoes binary inversion thresholding to enhance contrast. Bounding boxes around previously detected templates are loaded from a corresponding pickle file, and these regions are masked to zero out any pixels inside the boxes, ensuring that only corners outside the template regions are detected.

3) Gaussian Blurring: The thresholded and masked image is then blurred using a Gaussian filter. This blurring smooths the image and helps reduce noise, enabling the Harris Corner Detection algorithm to focus on significant corner features.

4) Corner Detection with Harris Algorithm: Using OpenCV's `goodFeaturesToTrack` function with the Harris Corner Detection method, the algorithm identifies prominent corners within the processed image. Each detected corner is marked with a small rectangle, and the coordinates are saved for potential visualization.

5) Evaluation and User Feedback: Once corners are detected, an optional visualization displays the detected corners overlaid on the image for user verification. The user can then input whether the detected corners are correct and, if incorrect, provide the number of false positives, false negatives, and total nodes to adjust the model's accuracy.

6) Saving Results: Detected corners are saved as serialized `pickle` files, which store the corner coordinates for each image. This data is stored in the predefined `pickles` directory for easy retrieval.

7) Performance Calculation: For each image, the corner detection process outputs evaluation metrics, including the number of correct detections, false positives, false negatives, and true positives. These values are accumulated and reported for each dataset path, providing an overall assessment of the corner detection accuracy.

Fig. 4. Example of detected corner points in molecular structures

E. Bond Edge Detection

1) Corner-to-Corner Bond Detection: Each pair of detected corners in the molecular structure is evaluated for potential bonding. The vector between each corner pair is calculated, and bounding boxes are generated around the potential bond regions. These regions are analyzed to detect the presence of a line using the Hough transform. Only pairs with line detections within a specified angular tolerance are confirmed as bonds.

2) Hough Line Transform: For each possible bond region, a mask is applied to isolate the region between two corners. The Hough transform is used to identify lines within this mask, and detected lines that align with the corner-to-corner vector are considered as bond detections. The number of detected lines and alignment with the expected angle help confirm the bond.

F. Bond Classification

1) Data Preparation and Preprocessing:

2) Loading and Organizing Training Data: Training images for each bond type (single, double, triple, dashed, and wedge) are loaded into a dictionary, 'BOND_TRAINING_DICT', which categorizes images based on bond type. Ground truth data for OCR and corner positions are also loaded from respective text files, enabling accurate evaluation.

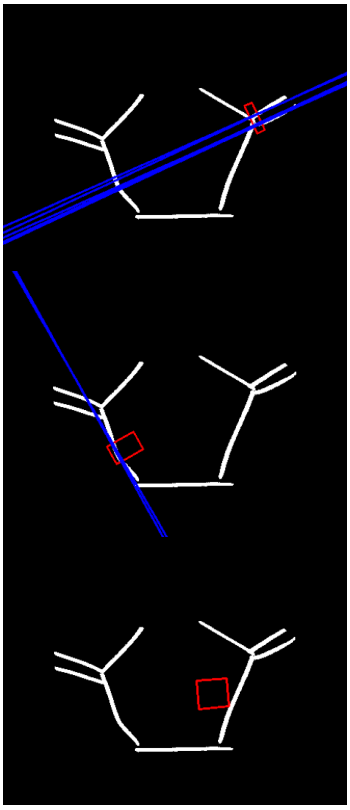


Fig. 5. Top, Middle: Hough line detections (blue) for the node-node pairs with a bond for a given window (red) in the bounding box. Bottom: A window between two opposite nodes that will not have a Hough line detection, so the algorithm will not assign it a bond, even though there is contamination elsewhere in the bounding box.

3) Preprocessing Bond Images: Images are downscaled to a fixed size and padded to normalize width across different bond types. Thresholding and Gaussian blurring are applied to enhance feature consistency. Each bond image is then divided into smaller regions, with the center region extracted for further processing.

4) Histogram of Oriented Gradients (HOG) Feature Extraction: HOG features are extracted from each processed bond image. The HOG method computes gradient orientations, quantizing them into a histogram representation that captures essential shape information. This feature vector provides a robust input for classification.

5) Classifier Training: Three classifiers are trained on the extracted HOG features: - Support Vector Machine (SVM) with a linear kernel - Logistic Regression - Decision Tree Classifier

Each classifier is trained using a specified training split. During training, images are randomly assigned to training and test sets, and features are extracted from multiple regions within each bond image. The classifier that achieves the best performance can be chosen for final bond classification.

Parameter	Value	Description
Threshold Value (THRESH_VAL)	100	Binary threshold value to enhance contrast.
Line Width (LINE_WIDTH)	18	Width of the Gaussian blur kernel applied to images.
Border (BORDER)	30	Border size added to images for better edge detection.
Minimum Scale (min_scale)	0.3	Minimum scaling factor for template matching.
Maximum Scale (max_scale)	1.0	Maximum scaling factor for template matching.
Number of Scales (n_scales)	15	Number of scale levels between min and max scales.
Matching Threshold (threshold)	0.6	Threshold for template matching score to accept a match.
Tolerance (tol)	0.77	Tolerance for filtering results with non-maximal suppression.

TABLE I
PARAMETERS USED IN THE TEMPLATE MATCHING PROCESS FOR MOLECULE DETECTION AND CLASSIFICATION.

6) Detecting Bond Sub-images: The function 'get_bonds' processes pairs of nodes (corners) and extracts sub-images between them, which represent potential bond regions. Each sub-image is transformed and rotated to align the bond horizontally, facilitating accurate classification.

7) Bond Classification: Each bond sub-image is processed by a chosen classifier, which predicts the bond type (single, double, etc.) based on the extracted HOG features. The classifier's prediction is then assigned as the bond type for that sub-image.

8) Overlaying Detected Bonds on Images: Detected bonds are visualized by overlaying colored lines between nodes in the original image. Each bond type has a unique color, making it easy to distinguish between bond types in the output image. Additionally, bounding boxes are drawn around detected structures, color-coded by molecular type.

9) Classification Accuracy: The classifier's performance is evaluated based on correct bond classifications (true positives), as well as any false positives and false negatives. The accuracy of the classifier is reported, showing the proportion of correct classifications for each bond type.

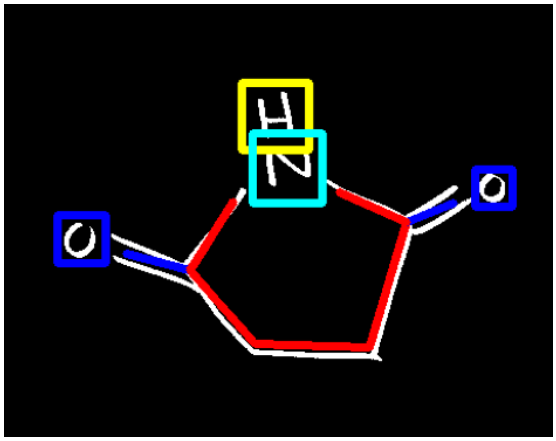


Fig. 6. Final bond classification results with each bond type displayed in its respective color.

IV. RESULTS

A. Template Matching

The template matching process involved several key parameters to optimize detection accuracy and reduce false positives. The parameters used in this study are summarized in Table I.

1) Performance Evaluation: The performance of the template matching algorithm was evaluated in terms of precision, recall, and accuracy for detecting molecular structures in images. Precision is defined as the ratio of true positives to the sum of true positives and false positives, while recall is the ratio of true positives to the sum of true positives and false negatives. Accuracy represents the overall proportion of correct classifications in the dataset.

2) Results Summary: Table IV provides a summary of the precision, recall, and accuracy values obtained for each molecule in the dataset.

Molecule ID	Precision	Recall	Accuracy
Struct1	1.0	1.0	1.0
Struct4	1.0	1.0	1.0
Struct5	0.55	1.0	0.75
Struct8	1.0	1.0	1.0
Struct13	0.95	0.97	0.97
Struct16	1.0	1.0	1.0
Struct19	0.96	0.79	0.2
Struct20	0.81	0.65	0.55
Struct22	0.98	0.90	0.4

TABLE II
PRECISION, RECALL, AND ACCURACY RESULTS FOR EACH MOLECULE AFTER TEMPLATE MATCHING IN THE DATASET.

3) Discussion: The results in Table IV show variability in the algorithm's precision, recall, and accuracy across different molecular structures. Several molecules, including "Struct1," "Struct4," "Struct8," and "Struct16," achieved perfect scores with precision, recall, and accuracy all at 1.0, indicating consistent, highly accurate detections without any false positives or false negatives.

However, certain molecules such as "Struct5" and "Struct19" showed notable differences between precision, recall, and accuracy. For example, "Struct5" had a lower precision of 0.55 but a perfect recall of 1.0, which suggests the algorithm detected all true instances of "Struct5" but included some false positives, reducing precision. Conversely, "Struct19" had a relatively high precision of 0.96 but a recall of 0.79 and an overall accuracy of only 0.2, indicating that while the algorithm was selective, it missed a significant number of true instances, impacting the accuracy score.

"Molecule ID 'Struct22" achieved relatively high precision and recall values at 0.98 and 0.90, respectively, though its accuracy was only 0.4. This discrepancy suggests a limited detection of true negatives for this structure, potentially due to

overlapping or similar patterns that led to higher false positives or false negatives.

The high precision and recall values for most molecules confirm the algorithm's ability to detect and classify molecular structures effectively in many cases. The combination of a high threshold value, Gaussian blurring, and the tolerance for non-maximal suppression were instrumental in achieving these results by reducing noise and minimizing overlapping detections. Overall, these parameters contributed to a robust template matching process for accurate molecular structure detection in this study.

B. Corner Detection

The corner detection process involved various key parameters to enhance detection accuracy. The parameters used in this study are summarized below:

- **Threshold Value (THRESH_VAL)**: 100 — Used to binarize the image and enhance contrast.
- **Line Width (LINE_WIDTH)**: 18 — Applied in Gaussian blurring to smooth edges and reduce noise.
- **Border Size (BORDER)**: 30 — Additional border applied to images to improve edge detection.
- **Max Corners (max_corners)**: 20 — Limits the number of corners detected per image.
- **Rect Width (rect_w)**: 6 — Size of the bounding box for each detected corner.

1) Performance Evaluation: The performance of the corner detection algorithm was evaluated based on:

- **Correct Detection (corr)**: A binary score indicating if the corners were correctly identified based on user feedback.
- **False Positives (FP)**: The number of incorrectly identified corners.
- **False Negatives (FN)**: The number of missed corners.
- **True Positives (TP)**: The number of correctly identified corners.

2) Results Summary: The table below summarizes the results for each structure in the dataset.

Molecule ID	Correct Detections	False Positives	False Negatives	Accuracy
Struct1	37	3	0	0.92
Struct4	23	21	3	0.57
Struct5	5	67	21	0.12
Struct8	38	3	0	0.95
Struct13	40	0	0	1.0
Struct16	38	3	0	0.95
Struct19	18	31	20	0.45
Struct20	34	7	1	0.85
Struct22	34	5	3	0.85

TABLE III
SUMMARY OF CORNER DETECTION METRICS FOR EACH MOLECULE IN THE DATASET.

3) Discussion: The results in Table IV highlight varying levels of accuracy in the corner detection algorithm across different molecular structures. Molecules such as "Struct1," "Struct8," "Struct13," and "Struct16" achieved high accuracy values, with "Struct13" showing a perfect detection rate of 100% accuracy. These high accuracy levels indicate effective corner detection for molecules with simpler structures or more distinct corners, where false positives and false negatives were minimal or nonexistent.

In contrast, more complex structures such as "Struct5" and "Struct19" displayed lower accuracy scores of 0.12 and 0.45, respectively. These low accuracy values stem from a high number of false positives and false negatives, suggesting that intricate molecular patterns led to challenges in corner detection. For example, "Struct5" recorded 67 false positives and 21 false negatives, indicating potential overlaps or noise in the structure that misled the detection algorithm.

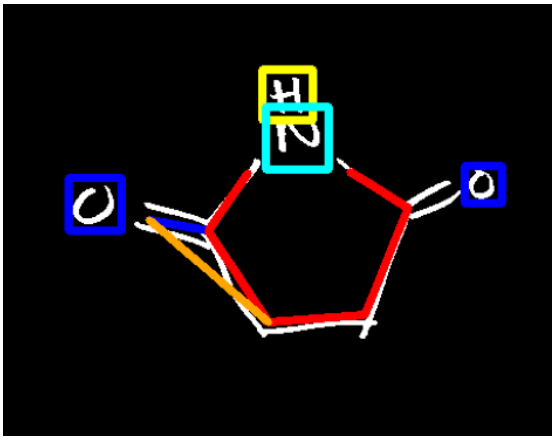


Fig. 7. Illustration of corner detection and line classification. The red and blue boxes represent correctly detected corners and chemical elements, respectively. The orange line is a misclassification resulting from an error in earlier corner point detection. The missed corner points led the algorithm to mistakenly classify this edge as a bond, even though it does not align with the intended molecular structure. Such misclassifications can occur when closely spaced or overlapping structures interfere with the corner detection process. Future improvements could focus on refining corner detection accuracy to avoid similar issues.

Key factors contributing to variability in performance include:

- **High Noise Levels**: While Gaussian blurring was employed to reduce noise, certain images contained intricate details that still interfered with accurate corner detection, contributing to false positives.
- **Structural Complexity**: Molecules with overlapping or closely spaced features, such as "Struct5" and "Struct19," introduced ambiguities in corner identification. This led to an increased rate of false positives, as the algorithm sometimes misinterpreted adjacent patterns as corners.
- **Missed Corners**: High false negative counts, particularly in complex structures, suggest that some true corners were missed due to variation in line width, drawing style, or image resolution.

Overall, the combination of thresholding, border addition, and Gaussian blurring contributed to high accuracy in cases where molecular structures were simpler or clearly defined. However, for more complex molecules, further refinement of

parameters such as `max_corners` and improved noise reduction techniques may be needed to enhance corner detection consistency. Future adjustments should focus on improving the algorithm's robustness to handle diverse structural patterns, reducing false detections in complex molecules.

V. Final Results

A. Bond Detection and Classification

The bond detection and classification process aimed to identify and categorize bonds between molecular structures accurately. The algorithm was evaluated based on its ability to detect different types of bonds, including single, double, triple, dashed, and wedge bonds, and classify them according to their structural role in the molecule.

1) Detection and Classification Performance: The performance of the bond detection algorithm was measured in terms of its precision in identifying bonds correctly, minimizing misclassifications, and accurately categorizing bond types. Each detected bond was verified against a ground truth dataset to determine if it was correctly classified. The misclassification rate was observed to increase in cases where the algorithm missed initial corner points, leading to errors in bond type identification. For example, the orange line misclassification (as shown in Fig. 7) occurred due to an earlier corner detection error, causing the bond to be incorrectly categorized.

B. Final Results and Model Accuracy

The final results for the bond detection and classification algorithm are presented in Table IV. These results categorize each test case based on the accuracy of bond detection:

- **Fully Correct**: Cases where all bonds and corners were detected and classified accurately.
- **Partially Correct**: Cases where some bonds were correctly classified, but one or more misclassifications occurred.
- **Model Accuracy**: The overall accuracy of the bond detection and classification model across the dataset.

Molecule ID	Fully Correct	Partially Correct	Accuracy
Struct1	36	2	0.95
Struct4	4	19	0.57
Struct5	0	9	0.22
Struct8	0	9	0.22
Struct13	0	33	0.82
Struct16	22	8	0.75
Struct19	0	24	0.60
Struct20	8	23	0.77
Struct22	7	22	0.72

TABLE IV
SUMMARY OF FINAL RESULTS FOR EACH MOLECULE IN THE DATASET.

1) Discussion: The results in Table IV demonstrate variability in the bond detection and classification algorithm's performance across different molecular structures. The accuracy ranged from 0.22 to 0.95, highlighting that the algorithm was more effective with certain structures than others. For example, "Struct1" achieved a high accuracy of 0.95 with 36 fully correct detections, indicating that the algorithm performed well on simpler or more distinct molecular configurations.

On the other hand, molecules like "Struct5" and "Struct8" had lower accuracies of 0.22, with no fully correct detections, reflecting challenges in detecting bonds and corners accurately in more complex or ambiguous structures. These cases were prone to misclassifications, likely due to overlapping bonds or indistinct corners, which the algorithm struggled to distinguish accurately.

Several factors contributed to the variability in performance:

- **Structural Complexity**: Molecules with overlapping or closely spaced bonds, such as "Struct5" and "Struct8," posed difficulties for the algorithm, leading to a higher rate of partially correct classifications. These cases often had ambiguous bond locations, making it challenging to identify all bonds accurately.
- **Corner Detection Errors**: Missed or incorrectly detected corners were a common source of errors in bond classification. For instance, as illustrated in Fig. 7, a missed corner led to an orange line being incorrectly classified as a bond. This type of error underscores the need for more precise corner detection to improve overall classification accuracy.
- **Noise and Image Artifacts**: In some cases, image noise or artifacts interfered with bond detection, especially in molecules with complex or dense structural patterns. Gaussian blurring helped to mitigate noise to some extent, but additional filtering techniques may be needed to handle these cases more effectively.

Overall, the algorithm performed well on simpler molecular structures, as seen in "Struct1" and "Struct16," which had higher accuracy scores. However, it showed limitations when dealing with complex configurations that involved overlapping bonds or challenging corner placements. Future improvements could focus on enhancing the corner detection algorithm and applying advanced noise-reduction techniques to improve model accuracy, particularly for structures prone to partial misclassifications. These adjustments are expected to help the model better handle the complexity of diverse molecular structures in the dataset.

VI. Conclusion

This study presented an approach for bond detection and classification in molecular structures, utilizing a combination of template matching, corner detection, and classification techniques. The results demonstrate that the algorithm performs effectively on simpler molecular structures, achieving high accuracy for cases where bonds and corners are distinct and easily identifiable. For molecules like "Struct1" and "Struct16," the algorithm successfully detected and classified bonds with minimal errors, indicating robustness for well-defined structures.

However, the model's performance declined with more complex configurations, such as "Struct5" and "Struct8," where overlapping bonds and ambiguous corner placements introduced significant challenges. Corner detection inaccuracies

were a primary source of error, often leading to bond misclassifications, as illustrated by the orange line misclassification in Fig. 7. Noise and image artifacts further affected detection accuracy, particularly in densely structured molecules, resulting in partially correct classifications.

A. Future Improvements

To enhance the model's performance across diverse molecular structures, several future improvements are proposed:

- **Refinement of Corner Detection**: Enhancing the corner detection algorithm is critical for reducing bond misclassifications. Techniques such as adaptive corner detection or machine learning-based corner refinement could provide more accurate corner identification, minimizing errors in bond classification.
- **Advanced Noise Reduction**: Additional noise-reduction methods, such as median filtering or deep learning-based denoising, could further reduce interference from image artifacts. This would improve bond detection in complex molecules where noise currently affects accuracy.
- **Improved Template Matching for Overlapping Bonds**: Future iterations of the model could incorporate more sophisticated template matching algorithms capable of distinguishing overlapping bonds. This could involve multi-scale template matching or the use of convolutional neural networks trained specifically for bond detection in crowded environments.
- **Incorporation of Additional Structural Features**: Adding contextual information, such as bond angles or relative positions of atoms, may provide the model with a more comprehensive understanding of the molecular structure, aiding in the resolution of ambiguous cases.

In conclusion, while the bond detection and classification algorithm proved effective in detecting and categorizing bonds in simpler molecular configurations, further enhancements are required to achieve consistent accuracy across complex structures. With improvements in corner detection, noise reduction, and template matching, the model holds potential for robust application in automated chemical structure recognition.

Code Access

The code for this project is available open-source. The repository is located at https://github.com/Apetun/Optical-Chemical-Structure-Recognition. Instructions for downloading data and usage are provided on the GitHub page.

Acknowledgment

This project was implemented in Python 3.12 with the additional use of the Anaconda distribution and Scikit-Learn for machine learning classifiers. The development of this project was heavily based on the research paper titled "Optical Recognition of Hand-Drawn Chemical Structures" by Bradley Emi from the Department of Computer Science at Stanford University. We acknowledge and appreciate the foundational insights provided by this work, which greatly informed our approach to bond detection and classification in chemical structures.

References

[1] F. Lin and J. Li, "MPOCSR: Optical Chemical Structure Recognition based on Multi-Path Vision Transformer," *Complex Intell. Syst.*, vol. 10, pp. 7553–7563, 2024. [Online]. Available: https://doi.org/10.1007/s40747-024-01561-6

[2] B. Emi, "Optical Recognition of Hand-Drawn Chemical Structures," Dept. of Computer Science, Stanford University.

[3] Z. Xu, J. Li, Z. Yang, S. Li, and H. Li, "SwinOCSR: End-to-End Optical Chemical Structure Recognition Using a Swin Transformer," *J. Cheminform.*, vol. 14, no. 1, p. 41, Jul. 2022. [Online]. Available: https://doi.org/10.1186/s13321-022-00624-5

[4] K. Rajan, H. O. Brinkhaus, A. Zielesny, *et al.*, "A Review of Optical Chemical Structure Recognition Tools," *J. Cheminform.*, vol. 12, p. 60, 2020. [Online]. Available: https://doi.org/10.1186/s13321-020-00465-0

[5] F. Musazade, N. Jamalova, and J. Hasanov, "Review of Techniques and Models Used in Optical Chemical Structure Recognition in Images and Scanned Documents," *J. Cheminform.*, vol. 14, p. 61, 2022. [Online]. Available: https://doi.org/10.1186/s13321-022-00642-3

[6] A. Gaulton and J. P. Overington, "Role of open chemical data in aiding drug discovery and design," *Future Med. Chem.*, vol. 2, pp. 903–907, 2010.

[7] T. Kind, M. Scholz, and O. Fiehn, "How large is the metabolome? A critical analysis of data exchange practices in chemistry," *PLoS One*, vol. 4, p. e5440, 2009.

[8] G. R. Rosania, G. Crippen, P. Woolf, D. States, and K. Shedden, "A cheminformatic toolkit for mining biomedical knowledge," *Pharm. Res.*, vol. 24, no. 10, pp. 1791-1802, Oct. 2007.

[9] R. Casey *et al.*, "Optical Recognition of Chemical Graphics," in *Proc. 2nd Int. Conf. Document Analysis and Recognition*, 1993.

[10] P. Ibison *et al.*, "Chemical Literature Data Extraction: The CLiDE project," *J. Chem. Inf. Comput. Sci.*, vol. 33, pp. 338-344, 1993.

[11] J. Park *et al.*, "Image-to-Structure Task by ChemReader," *Text Retrieval Conf.*, 2011.

[12] I. Filippov and M. Nicklaus, "Optical Structure Recognition Software to Recover Chemical Information: OSRA, An Open Source Solution," *J. Chem. Inf. Model.*, vol. 49, no. 3, pp. 740-743, 2009.

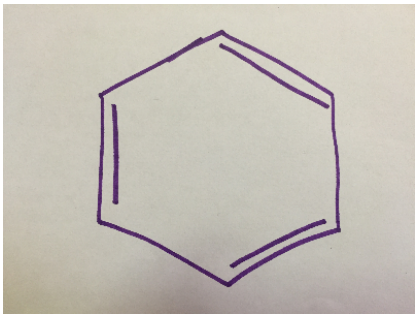
[13] P. Frasconi *et al.*, "Markov Logic Networks for Optical Chemical Structure Recognition," *J. Chem. Inf. Model.*, vol. 54, pp. 2380-2390, 2014.

[14] D. Douglas and T. Peucker, "Algorithms for the reduction of the number of points required for representing a digitized line or its caricature," *Can. Cartogr.*, vol. 10, pp. 112-122, 1973.

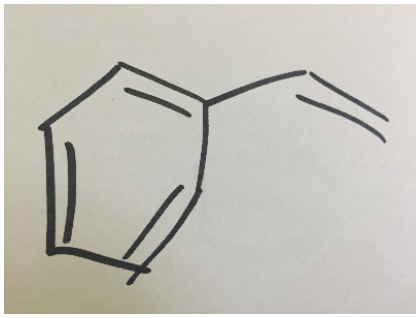
[15] N. Dalal and B. Triggs, "Histograms of Oriented Gradients for Human Detection," in *Proc. Int. Conf. Computer Vision Pattern Recognition (CVPR)*, San Diego, CA, USA, 2005, pp. 886–893.

[16] T. E. de Campos, B. R. Babu, and M. Varma, "Character Recognition in Natural Images," in *Proc. Int. Conf. Computer Vision Theory and Applications*, Lisbon, Portugal, Feb. 2009.

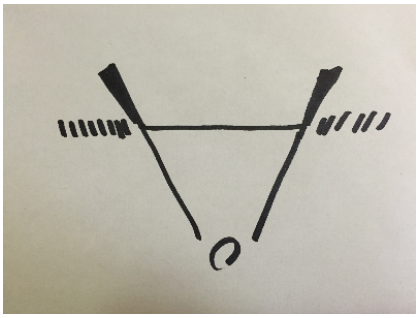
[17] C. Harris and M. Stephens, "A Combined Corner and Edge Detector," Plessey Research, 1988.



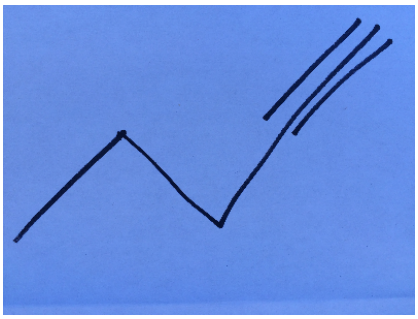
Struct1



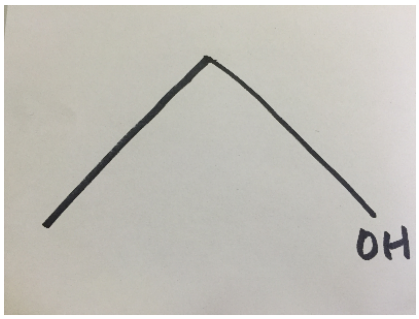
Struct4



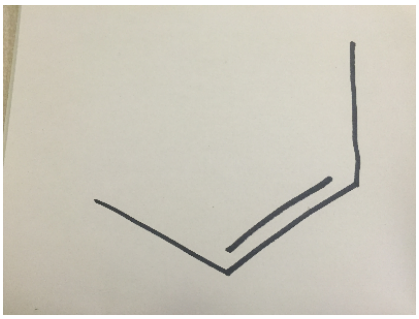
Struct5



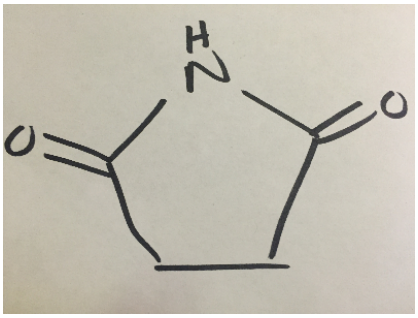
Struct8



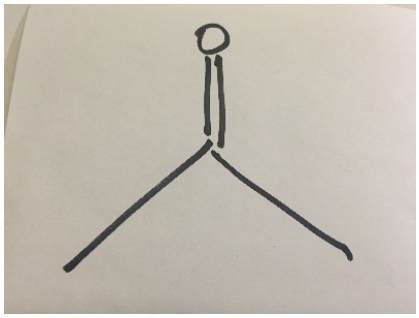
Struct13



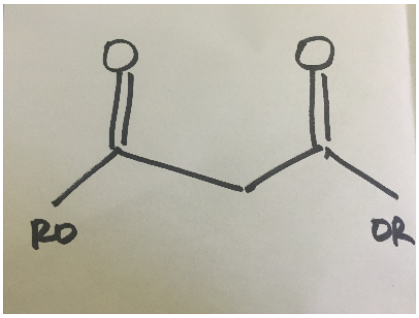
Struct16



Struct19



Struct20



Struct22