Online Notice Board System

T. Yogadharshinii*, R. Jayasri, S. Thasneem Banu, K. Sundara Krishnan, N. Hema Rajini

Department of Computer Science and Engineering, Alagappa Chettiar Government College of Engineering and Technology, Karaikudi 630003, Tamil Nadu, India

Abstract: *This paper presents the development and implementation of an Online Notice Board System (ONBS) that automates departmental notifications using Optical Character Recognition (OCR) and Natural Language Processing (NLP). The system digitizes image- based notices and schedules events through a Django-based event calendar. Image preprocessing techniques like grayscaling and thresholding are applied to improve text extraction accuracy. Extracted text is processed using Python's Tesseract OCR library (PyTesseract), structured for event categorization, and summarized via Natural Language Toolkit (NLTK) for concise display. The ONBS is designed to improve communication within educational institutions, providing a sustainable, accessible, and user-friendly platform.*

Keywords:

Optical Character Recognition (OCR) Natural Language Processing (NLP) Natural Language Toolkit (NLTK)

I. INTRODUCTION

Notice boards are integral to communication in educational institutions, serving as the primary medium for conveying important announcements, event details, and academic information to students and staff. However, traditional physical boards present several challenges, including limited accessibility, the risk of outdated or missing information, and the manual effort required for regular maintenance. Furthermore, in large institutions, the information displayed on these boards may not be easily accessible to everyone, especially for those in remote locations or with mobility challenges.

To address these issues, this study introduces an ONBS that digitizes the traditional notice board using a combination of OCR [1], NLP [2], and a responsive web-based interface. ONBS transforms static, image-based notices into structured, searchable, and categorized content. This not only eliminates the need for manual updates but also ensures that all users, regardless of their physical location, receive timely and accurate information.

The system architecture integrates several key technologies to enhance performance and usability. The frontend is built using React.js [3], enabling dynamic content updates and user- friendly interaction. On the backend, Django [4] manages data handling, authentication, and API interactions with a lightweight SQLite database. For effective OCR, OpenCV [5] is utilized for preprocessing tasks such as grayscaling, thresholding, and morphological operations, which significantly improve the clarity of text before extraction.

Text extraction is performed using Tesseract OCR through the PyTesseract Python wrapper [1], known for its accuracy in recognizing text even from low-quality images. Post-extraction, NLTK (Natural Language Toolkit) [2] is employed for NLP tasks such as text summarization and event parsing, allowing the system to automatically extract relevant details like event titles, dates, and descriptions.

In addition to basic text handling, ONBS supports features such as Automated summarization of lengthy notices to display concise information using NLP techniques [2]. Event scheduling and visualization via a calendar-based interface built in React. Performance optimization, includes reduced execution time through image preprocessing and modular processing workflows [5].

Scalability and future-proofing, allows the system to adapt to multilingual support and deeper machine learning integration [6-8]. With these features, ONBS aims to be a transformative solution for academic institutions by improving communication efficiency, accessibility, and environmental sustainability through reduced paper usage. The platform offers a comprehensive alternative to traditional notice boards, facilitating real-time digital interaction while maintaining an intuitive and inclusive user experience.

II. METHODOLOGY

The ONBS integrates advanced technologies to automate notice processing, event scheduling, and communication. The system architecture is modular, ensuring scalability and adaptability. The key components of the methodology are detailed below:

A. Technology Stack

Frontend: The user interface is developed using HTML, CSS, and React [3], providing a responsive design with real-time updates and seamless navigation.

Backend: A Django-based backend [4, 5, 9] and machine learning using python[10-12] manages data processing, business logic, and secure storage, leveraging SQLite for lightweight and efficient database operations.

Image Processing: OpenCV [5] is employed for preprocessing tasks such as grayscaling, thresholding, and dilation. These operations enhance text clarity, remove noise, and prepare images for OCR.

OCR Module: PyTesseract, a Python wrapper for Tesseract OCR [1], extracts text from uploaded images and PDFs, enabling accurate recognition and conversion of notices into structured data. NLP Module: NLP techniques, implemented using NLTK [2], are utilized for summarization and event parsing. This ensures the extraction of event-specific information such as titles, dates, and descriptions.

B. Workflow

The workflow of the Online Notice Board System (ONBS) is as follows:

- Image Upload: Users upload scanned notices/ PDFs through the user interface.
- Preprocessing: Uploaded files are processed using OpenCV to enhance text clarity.
- Text Extraction: PyTesseract extracts textual content from the preprocessed images.
- Text Analysis: NLTK analyzes extracted text to identify event details like dates and titles.
- Data Storage and Display: Event details are store in SQLite and displayed on a React-based calendar.

C. *Image Preprocessing Techniques for Enhanced OCR Accuracy*

To effectively extract text from notices using OCR, a series of image preprocessing techniques is applied. These transformations enhance text visibility and structure, enabling the OCR engine(e.g., Tesseract) to perform more accurately. The first step in the preprocessing pipeline is converting the input color image into grayscale. This is achieved using the luminance based formula given in the equ. (1).

$$\text{Gray} = 0.299 \times R + 0.587 \times G + 0.114 \times B, \quad (1)$$

where R, G, and B are the red, green, and blue color intensities of each pixel, respectively. This weighted sum preserves brightness information while eliminating color details, which simplifies further image processing.

Following this, thresholding is applied to transform the grayscale image into a binary image (comprising only black and white pixels). The formula used here is given in equ., (2).

$$\text{Binary}(x, y) = 255 \text{ if } \text{Gray}(x, y) > T \text{ else } 0, \quad (2)$$

where T is a fixed threshold value. This technique helps in segmenting foreground text from the background, making the text areas stand out clearly. In scenarios where lighting varies across the image (common in scanned documents or photographs), adaptive thresholding is more effective. This computes a dynamic threshold for each pixel using the formula given in equ. (3)

$T(x,y)$ = mean of neighborhood - C, followed by binarization:

$$\text{Binary}(x, y) = 255 \text{ if } \text{Gray}(x\ y) > T(x, y) \text{ else } 0. , \quad (3)$$

Here, C is a constant subtracted from the neighborhood mean to fine- tune contrast. To improve text visibility further, especially when the text is faint or lightly printed, inversion is performed. The pixel values are flipped using equ.(4)

$$\text{Inverted}(x, y) = 255 - \text{Binary}(x, y), \quad (4)$$

This technique is particularly useful when the OCR engine works better with dark text on a light background. After binarization and inversion, morphological operations are used to enhance the structure of text.

Dilation is employed to connect disjointed elements such as broken characters or fragmented text lines. The pixel value at a location is set to the maximum value within its neighborhood using a predefined structuring element (kernel), mathematically represented as:

Dilated = max(neighborhood).

Conversely, erosion, if applied, helps remove small noise or specks by reducing the size of white regions, defined by:

Eroded = min(neighborhood).

Finally, to isolate individual components of the document such as text blocks, images, or tables, contour detection is applied. This step uses the OpenCV function:

contours, hierarchy = cv2.findContours(binary, mode, method).

It identifies and outlines the boundaries of segmented objects in the binary image, enabling structured document analysis and further processing like table recognition or text region classification. These preprocessing steps collectively ensure that the input images are well- structured and optimized for text extraction, thereby significantly improving the performance and accuracy of OCR in the proposed ONBS.

III. PERFORMANCE MEASURE AND COMPARATIVE ANALYSIS

A comprehensive performance evaluation was conducted to compare the proposed model with a baseline model for the text extraction and summarization modules. The evaluation focused on the metrics of accuracy, summarization quality, and execution time, providing insight into the effectiveness of the proposed system.

A. *Models*

Baseline Model

1. Text Extraction: PyPDF2.[1]
2. Summarization: Hugging Face Transformers (BART model) [7]

Proposed Model

1. Text Extraction: OpenCV for preprocessing [5] and Tesseract OCR [1].
2. Summarization: NLTK for text summarization [2].

B. *Evaluation Metrics*

The models were assessed using the following metrics:

- Precision, Recall, and F1 Score: Evaluated the accuracy of text extraction.
- ROUGE Scores: Measured summarization quality (ROUGE-1, ROUGE-2, ROUGE-L).
- Execution Time: Assessed processing efficiency.
- Improvement Percentage:Highlighted relative performance gains.

C. *Performance Comparison*

No more than three levels of headings should be used. All headings must be in 10pt font. Every word in a heading must be capitalized except for short minor words.

1. *Text Extraction Accuracy*: The integration of OpenCV for preprocessing, combined with Tesseract OCR, enhanced the precision, recall, and F1 scores of the proposed model. These improvements demonstrate the system's robustness in handling noisy and complex image inputs, outperforming traditional text extraction approaches. Text extraction accuracy is given in Table I.

Table I Text Extraction Accuracy

Metric	**Baseline**	**Proposed**	**Improvement**
Precision (%)	82.4	90.6	+8.2%
Recall (%)	80.1	88.6	+8.5%
F1 Score (%)	81.2	89.6	+8.4%

2. Summarization Quality: The proposed model showed a notable increase in ROUGE-1, ROUGE-2, and ROUGE-L scores, indicating a significant improvement in the quality of the generated summaries. This enhancement allows for better retention of important event details while minimizing irrelevant information, making the summaries more concise and user- friendly. Summarization Quality values for baseline, proposed is given in Table II.

Table II Summarization Quality

Metric	**Baseline**	**Proposed**	**Improvement**
ROUGE-1 (%)	75.4	82.0	+6.6%
ROUGE-2 (%)	72.3	81.6	+7.2%
ROUGE-L (%)	78.0	85.2	+7.2%

3. *Execution Time:* The proposed model's processing the pipeline achieved higher efficiency with reduced execution times. The execution time for baseline, proposed is given in Table III.

TABLE III Execution Time

Metric	**Baseline**	**Proposed**	**Improvement**
Text Extraction	1.25	0.95	+24.0%
Summarization	1.60	1.30	+18.75%
Total Execution	2.85	2.25	+22.0%

These comparisons clearly demonstrate the proposed model's superior performance across all evaluated metrics, with notable gains in both accuracy and processing speed. The bar charts also emphasize the model's efficiency, showcasing its ability to handle large datasets faster than the baseline model without compromising quality. The performance comparison for baseline and proposed method is shown in Figure 1.

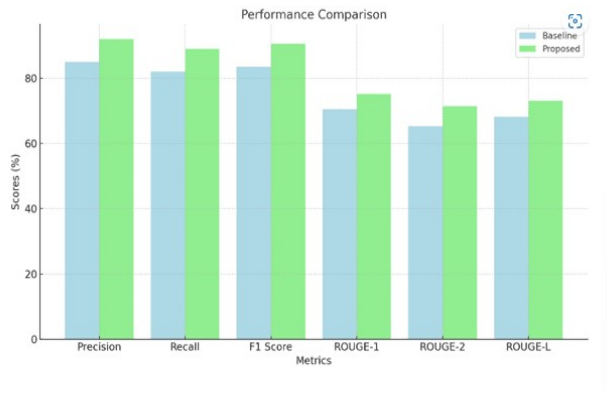

Fig. 1 Performance Comparison

The execution time comparison for baseline and proposed method is given in Figure 2.

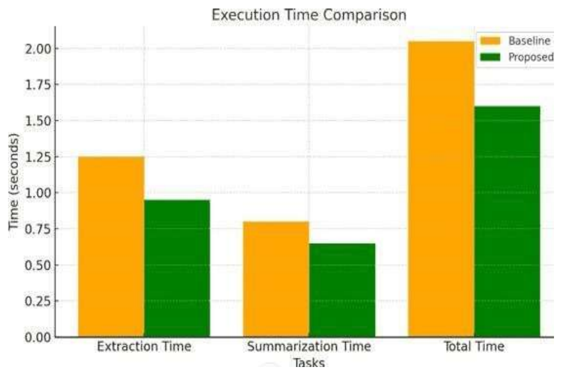

Fig. 2 Execution Time Comparison

The proposed model, integrating Tesseract OCR and NLTK, demonstrates significant improvements over the baseline model that uses PyPDF2 for text extraction and Hugging Face Transformers for summarization. Key advancements include:

- **Accuracy**: The proposed model enhanced text extraction accuracy, with improvements in precision, recall, and F1 score by an average of 8.37%.
- **Quality**: ROUGE scores saw a substantial increase, particularly in ROUGE-2, where the improvement was +9.3%.
- **Efficiency**: Execution time was reduced by 22%, making the model more suitable for real- time applications.

These results underscore the effectiveness of the proposed approach, which successfully balances accuracy, summarization quality, and computational efficiency, delivering a more reliable and faster system compared to the baseline. The performance metrics and execution time comparison for baseline and proposed method is given in Figure 3.

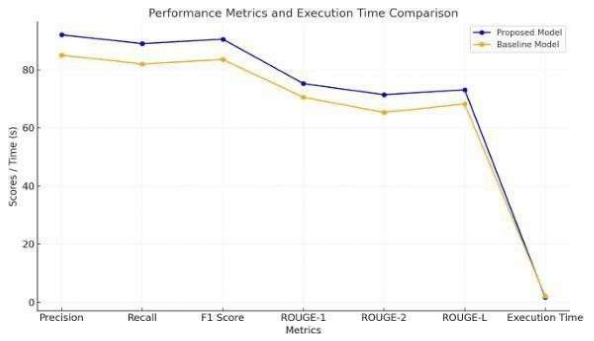

Fig. 3 Performance Metrics and Execution Time Comparison

V. NOVELTY

The ONBS presents a novel approach institutional communication by integrating OCR and NLP into a unified platform. Unlike traditional noticeboards or existing digital solutions, ONBS automates the entire process—from text extraction and summarization to event scheduling—eliminating the need for manual intervention. The system employs OpenCV for advanced image preprocessing and Tesseract OCR for high-accuracy text recognition, even from complex or noisy images, which sets it apart from conventional methods like PyPDF2.Furthermore, the use of NLTK for NLP enables automatic categorization and summarization of event details, ensuring that only the most relevant information is presented to users. The combination of these technologies results in an efficient, scalable, and user-friendly platform that transforms how educational institutions manage and disseminate information.

V. FUTURE ENHANCEMENTS

Advanced text mining approaches may enhance summarization and topic modeling. Machine learning methods could be used to classify and prioritize notices based on importance. Integrating deep learning techniques can further improve OCR performance on complex documents.

1. Improved Handwriting Recognition: Enhancing OCR performance on handwritten documents could expand the application's utility in recognizing a broader range of inputs, especially for more diverse documentation.
2. Language Support and Multilingual Summarization: Extending OCR and summarization to support multiple languages would increase inclusivity and make the platform more useful in multilingual educational or organizational settings.
3. Mobile Accessibility with Offline Capabilities: Integrating offline OCR and summarization capabilities on mobile devices could make the app more accessible for users with limited internet access, while still maintaining functionality.

CONCLUSION

The ONBS offers a scalable and efficient solution for automating institutional communication through OCR and NLP. By streamlining text extraction and event scheduling, it enhances accuracy and ensures timely updates. Future work will explore multi- language support, advanced analytics, and push notifications to further improve user engagement. The system's integration of OpenCV for preprocessing and Tesseract OCR significantly improves the accuracy of text extraction, even from noisy or low- quality images. Additionally, the use of NLTK for text summarization enables more concise and relevant event descriptions, enhancing user experience. As the system evolves, incorporating machine learning techniques for better event categorization and personalized notifications will further enhance its effectiveness and user satisfaction.

REFERENCES

[1] S. G., Tesseract OCR for Beginners, Independently Published, 2019. [Online]. Available: https://www.amazon.com/dp/1098538589

[2] S. Bird, E. Klein, and E. Loper, Natural Language Processing with Python, O'Reilly Media, 2009. [Online]. Available: https://www.oreilly.com/library/view/natural-language- processing/9780596803346/

[3] A. Banks and E. Porcello, Learning React: Functional Web Development with React and Redux, O'Reilly Media, 2019. [Online]. Available: https://www.oreilly.com/library/view/learning-

[4] W. S. Vincent, Django for Beginners: Build Websites with Python and Django, Indie Hackers, 2018. [Online]. Available: https://djangoforbeginners.com/

[5] D. L. Baggio, Mastering OpenCV with Practical Computer Vision Projects, Packt Publishing, 2019. [Online]. Available: https://www.packtpub.com/product/mastering-opencv-with- practical-computer-vision-projects/9781788621753

[6] A. Géron, Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow, 2nd ed., O'Reilly Media, 2019. [Online]. Available: https://www.oreilly.com/library/view/hands-on-machine-learning/9781492032632/

[7] S. Vajjala, B. Majumder, and A. Gupta, Practical Natural Language Processing: A Comprehensive Guide to Building Real- World NLP Systems, O'Reilly Media, 2020. [Online]. Available: https://www.oreilly.com/library/view/practical-natural- language/9781098101577/

[8] F. Chollet, Deep Learning with Python, Manning Publications, 2017. [Online]. Available:https://www.manning.com/books/deep- learning-with-python

[9] A. C. Müller and S. Guido, Introduction to Machine Learning with Python: A Guide for Data Scientists, O'Reilly Media, 2016. [Online]. Available: https://www.oreilly.com/library/view/introduction-to-machine/9781449369880/

[10] M. Alchin, Pro Django, 2nd ed., Apress, 2018. [Online]. Available: https://www.apress.com/gp/book/9781484231784

[11] S. Raschka and V. Mirjalili, Python Machine Learning, 2nd ed., Packt Publishing, 2017. [Online]. Available: https://www.packtpub.com/product/python-machine-learning-second-edition/9781788621753

[12] S. Raschka and V. Mirjalili, Python Machine Learning, 2nd ed., Packt Publishing, 2017. [Online]. Available: https://www.packtpub.com/product/python-machine-learning-second-edition/9781788621753