REBS Review of Economic & Business Studies

Volume 19, Issue 1/June 2026, pp. 189 of 198
ISSN-1843-763X
DOI: 10.47743/rebs-2026-1-0009

FROM CODE TO CLICKS: THE ECONOMIC AND EDUCATIONAL TRADE-OFFS IN AI-DRIVEN LICENSE PLATE RECOGNITION

ANDREI-OCTAVIAN MALAN[i], DAN-ANDREI SITAR-TĂUT[ii]

Abstract: *This paper explores the economic and educational trade-offs between code-based and low-code approaches in developing AI-driven license plate recognition (LPR) systems by using a comparative case study; one built in Python with custom models, the other via Roboflow's low-code platform, analyzing cost, scalability, and accessibility.*

Beyond development speed and technical performance, the study highlights how each approach affects access to AI tools, skill development, and implementation feasibility in resource-constrained contexts; as this analysis will show, low-code platforms reduce barriers for non-specialists and accelerate prototyping, while code-based systems offer greater flexibility, control, and long-term adaptability.

Finally, the paper argues that the choice between code and low code is not only technical but strategic; shaping how AI is integrated into education, entrepreneurship, and public infrastructure. Recognizing these trade-offs is important for institutions and innovators aiming to deploy AI in economically sustainable and performant ways.

Keywords: *educational trade-offs; technology; economic innovation; low-code platforms; license plate recognition; society impact.*

JEL Classification: *L86, D81*

1. INTRODUCTION

As artificial intelligence continues to be more present across industries, the balance between technological sophistication and practical accessibility becomes increasingly important. One domain where this contrast is particularly evident is license plate recognition (LPR), a field that combines computer vision, object detection, and optical character recognition (OCR) to automate vehicle identification in both public and private

[i] Babeș-Bolyai University, Faculty of Economics and Business Administration, Cluj-Napoca, Romania, andrei.malan@stud.ubbcluj.ro
[ii] Babeș-Bolyai University, Faculty of Economics and Business Administration, Cluj-Napoca, Romania, dan.sitar@econ.ubbcluj.ro

infrastructure; while traditionally the implementation of such systems has required significant programming and machine learning expertise, the rise of low-code and no-code platforms now offers new horizons for deploying intelligent solutions with minimal technical input. Beyond simplifying development, these tools also carry educational and economic significance by lowering entry barriers, reducing costs, and enabling larger participation in AI-related projects across sectors. Another perspective on the multitude of methods and techniques can be found in the literature (Shashirangana *et al.*, 2021), proving the complexity of the subject and the importance of addressing it from different angles. However, despite this growing interest, there is still a notable lack of comparative studies that directly evaluate low-code platforms against traditional code-based approaches in the context of LPR, leaving open questions about their practical and educational impact.

Consequently, this paper investigates the economic and educational implications of using code-based versus low-code approaches for developing AI-driven LPR systems; the core interest lies not only in comparing performance or accuracy, but in understanding how each development model influences cost structures, deployment feasibility, educational access, and the potential for innovation in emerging economies. For municipalities and startups alike, the choice between traditional programming and low-code workflows can affect development time, team composition, financial investment, and long-term sustainability. At the same time, in the context of higher education and tech entrepreneurship, these tools shape how future professionals learn about and interact with artificial intelligence and its specific applications.

To anchor this analysis, we use a comparative case study approach: on one side, a fully custom Python-based application built with YOLOv11, PaddleOCR, and GPT-based processing; on the other, a Roboflow based low code pipeline assembled using pre-trained object detection and OCR modules. This juxtaposition allows to examine not only the technical execution, but also the hidden trade-offs, such as platform dependency, flexibility, and the educational value of interacting directly with model logic versus abstracting it.

The research questions addressed in this paper are:

(RQ1) What are the financial and time-based trade-offs between developing a license plate recognition (LPR) system using traditional code-based approaches versus low-code/no-code platforms?

(RQ2) How can AI-driven LPR solutions be integrated into educational programs or leveraged in tech entrepreneurship initiatives, particularly to support workforce development?

(RQ3) What are the broader economic and educational implications of adopting low-code vs. code-based AI solutions in civic infrastructure like smart traffic systems?

The structure of the paper is organized to emphasize not only the technical dimensions of AI-driven license plate recognition, but more importantly, the broader educational and economic trade-offs involved in adopting different development paradigms. Chapter 2 offers the necessary background of previous studies and academic endeavors found in specific literature. Chapter 3 details the methodology employed in building and evaluating both a low-code and a code-based version of the system, highlighting the development paths taken rather than the algorithms themselves. Chapter 4 presents the core findings of the study, addressing three research questions that explore cost efficiency, scalability, accessibility in under-resourced contexts, and educational value. Finally, Chapter 5 highlights the conclusions and outlines potential directions for future research.

2. LITERATURE REVIEW

The literature discusses these topics in a variety of contexts. When speaking about YOLO models applied to a license plate recognition system, Istiak and Feng (2024) highlight that experimental results from a YOLOv8-based LPR model tailored to Bengali license plates demonstrate its capacity to address complex detection challenges, including script-specific recognition, corner-angle captures, and low-resolution imagery. The model effectively localized and segmented characters under conditions such as broken or obscured plates and the presence of multiple plates within a frame; these outcomes suggest that domain-specific training can produce models capable of reaching accurate detection and recognition in challenging visual environments. Moreover, migrating to a low-code task, Jenitta *et al.* (2023) proposed a Roboflow-enabled pipeline employing YOLOv5 for number plate detection, that achieved good bounding box generation on images. While the technical outcomes indicated high detection scores, the study also presented broader considerations: automated number plate recognition systems can improve traffic safety, reduce congestion, and enhance security, but they must be implemented with strict measures to protect privacy. Speaking in numbers, Jamtsho *et al.* (2021) observed that in a field evaluation involving 137 non-helmeted motorcyclists, the detection system successfully identified 135 as non-helmeted, corresponding to a recall rate above 98%. For license plate detection within this subset, the system achieved a rate of 98.5% (133 out of 135), such metrics indicating low false-positive rates and high operational reliability. Also, these findings may suggest that similar high-accuracy detection solutions could be economically viable for large-scale deployment. Using Roboflow may result in good performance metrics, as it is suggested in a study by Prasad *et al.*

(2024), where a YOLOv8-based object detection system developed for identifying the red pumpkin beetle in agricultural fields achieved notable performance metrics, including 86% precision, 83% accuracy, a mean average precision of 89%, an F1-score of 86.9%, and a recall rate of 88%. The authors used Roboflow for data preparation, image conversion, augmentation, and annotation, while training was conducted in Google Colab; these results illustrate YOLOv8's adaptability for real-time, high-accuracy detection tasks beyond traditional license plate recognition applications. An interesting experiment took place within Khachatryan and Parise (2025)'s work, where students were given a hands-on project requiring Roboflow for an object detection task, to mark their engagement and will to actively participate in the project's success. The note was that integrating Roboflow into educational contexts has been observed to increase student engagement and comprehension in computer vision coursework. In addition to this, by working with real-world datasets and model training and testing environments, students not only develop technical proficiency but also acquire practical problem-solving skills relevant to industry applications.

On the other hand, Businesswire (2023) argues that forecasts from the International Data Corporation project that the global AI software market will expand from $64 billion in 2022 to nearly $251 billion in 2027, reflecting a compound annual growth rate of 31.4%. This finding indicates an increasing demand for AI-driven solutions, including computer vision applications such as license plate recognition solutions; the economic implications are on 2 sides: organizations adopting such technologies can capitalize on efficiency gains and competitive advantages, while educational systems that train AI-capable professionals can directly contribute to and benefit from this market expansion. In addition to these figures, a McKinsey report based on survey data indicates that 78% of organizations have incorporated AI into at least one business function, up from 72% in early 2024 and 55% the previous year Sukharevsky *et al*. (2025). The largest increase was in IT, where adoption rose from 27% to 36% in just six months.

3. METHODOLOGY

In order to comparatively analyze the trade-offs between code-based and low-code AI development approaches, this study implemented the same license plate recognition (LPR) task using two distinct workflows: one developed in a visual, low-code environment (Roboflow), and the other constructed from scratch in Python, integrating YOLOv11 for object detection and PaddleOCR for text extraction. Each workflow consists of a multi-stage pipeline combining object detection, image

preprocessing, and optical character recognition, though their configuration, level of control, and customization capabilities differ in many aspects.

3.1. Code-Based Implementation Using YOLOv11n and PaddleOCR

The code-based system was implemented in Python and executed through a Streamlit web application interface, enabling interactive uploads and real-time visual feedback; the goal was to construct a fully functional license plate recognition (LPR) pipeline that can detect vehicles, localize plates, and extract alphanumeric characters from them.

The workflow begins with image input that is passed to a YOLOv11 Nano (YOLOv11n) model trained on the COCO dataset, which performs object detection to identify vehicles such as cars, buses, motorcycles, or trucks. Detected vehicles are then cropped from the scene to narrow the focus and improve the subsequent detection.

Each cropped vehicle region is processed by a second YOLOv8 model trained specifically for a license plate detection task, this model isolating the possible position of the plate within the vehicle bounding box. Text recognition is performed using PaddleOCR, a high-performance and multilingual OCR system designed for structured text extraction from images, according to the official documentation provided by PaddleOCR (2025). The output consists of recognized character strings along with confidence scores; the system parses these results, taking into account only those text segments with confidence levels exceeding a defined threshold (0.3 in our case), to form an initial raw prediction of the license plate number.

Post-processing is then applied to standardize and validate the text. A custom rule corrects common OCR errors (e.g., misinterpreting "0" as "O" or "8" as "B", but the examples may be more, as this is a standard issue in OCR tasks) and enforces a Romanian license plate format: one or two letters (county code), followed by two digits, and ending in three letters.

Although development was accelerated through the use of Cursor, an AI-based coding assistant, significant effort was still required to define robust pre and post-processing steps and to implement logic for OCR correction and validation. This underlines the reality that, even with AI-assisted tooling, code-based workflows require domain or business understanding and iterative refinement in different development zones as well as thorough testing.

This code-based pipeline provides a high degree of flexibility and transparency in each processing step, making it well suited for experimentation, model configuration or integration with advanced logic. However, it also requires considerable development time, familiarity with machine learning libraries, and access to computational resources.

3.2. Low-code Based Implementation Using Roboflow

The low-code-based implementation for License Plate Recognition (LPR) using Roboflow begins with an input image containing vehicles; we move to the first major step in the workflow which is the detection of vehicles using the YOLO11n object detection model. Here certain modifiable parameters include the Intersection over Union (IoU) threshold, which determines how overlapping bounding boxes are filtered(which in this case was set to 0.9, meaning that the system will try to recognize as many cars as possible, regardless of the overlapping degree), and the confidence threshold, which discards detections below a specified probability(in this case, the default value is 0.4) could be mentioned as more relevant. Following detection, a dynamic cropping stage isolates the license plate regions from the larger image.

A secondary detection phase is then applied, using a pretrained model specialized for license plate identification; unlike the general YOLO11n model, this detector is trained exclusively on license plate datasets, enabling it to handle variations in plate shapes or regional formatting conventions. Once the license plate region is precisely localized, the text recognition phase begins, by implementing two complementary methods: traditional Optical Character Recognition (OCR) and AI-based visual text extraction, using a GPT model. In the OCR method, a pretrained model optimized specifically for license plate fonts and formats detects individual characters or character groups. The resulting text is then standardized through a property definition process, converting all letters to uppercase in alignment with common license plate formatting standards across the world(Thus, it would apply to Romanian plates as well, to be consistent with the code-based approach and its processing).

In parallel, the cropped plate image is processed by OpenAI's GPT-4 model with vision capabilities(as a built-in block, this model is defaulted). Here, a custom prompt instructs the model to "read the license plate text, return just the text, else null if not found". The AI output can either serve as a fallback in cases where OCR fails or as a validation mechanism to confirm OCR results. The entire flow can be visualized in Figure 1.

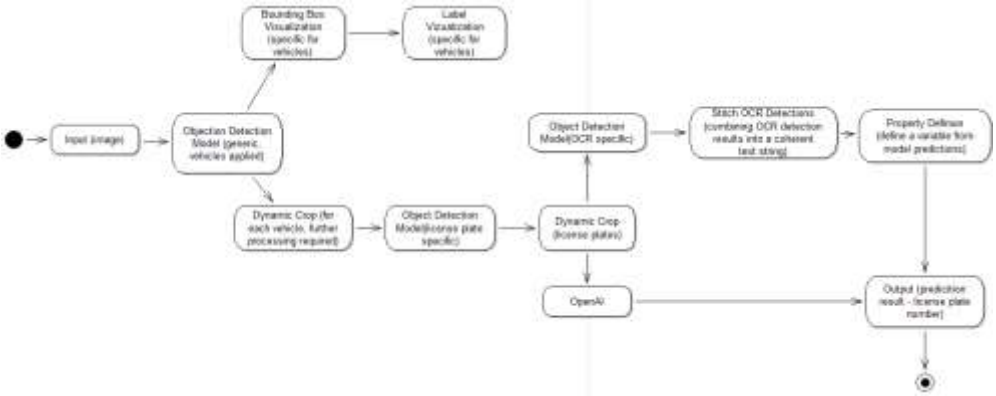


Figure 1. Low-code Roboflow methodology

4. DISCUSSIONS AND RESULTS

If we observe the outcome, the development of the code-based solution demanded a deeper technical understanding and longer setup time, particularly for tasks such as model configuration, bounding box calibration, custom rule creation, and integration of OCR logic. The YOLOv11n and YOLOv8 models, while highly performant, required more local testing and several iterative refinements to produce reliable outputs. Additionally, OCR accuracy was impacted by image quality, necessitating pre- and post-processing stages that had to be carefully tuned. One aspect that needs to be added here is the extensive need for quality images, as clarity plays an important role in all image-based tasks. In contrast, the Roboflow-based solution significantly reduces the complexity for deployment. Leveraging pretrained object detection and OCR blocks, along with a modular drag-and-drop interface, the system could be configured and tested with minimal programming knowledge. However, this ease of use came at the cost of reduced flexibility; fine-tuning parameters or enforcing specific formatting rules proves to be more challenging within the constraints of the platform. Economically, the low-code solution offered faster prototyping and reduced labor costs but could introduce, if scaled up, ongoing costs related to limitations of the subscription models.

Speaking in numbers, scaling an LPR system on Roboflow's Pro plan would cost approximately $49/month for basic package(360 credits/year included, where a credit is the unit by which the computation cost is calculated), $299/month at the base subscription for startups, with a custom pricing for enterprises, according to the Roboflow website (2025). We can see from this that when speaking of a span of over a

year, this amounts to at least $3,600 in operational expenses. In terms of development time, the code-based pipeline required 8 hours of active engineering effort, while the low-code Roboflow configuration was completed in about 2 hours, representing a 75% reduction in initial setup time. While code-based approaches require higher development effort, they offer long-term cost savings and superior customization; low-code platforms, by contrast, offer quick deployment and accessibility but may require subscription-based investment for advanced features and large-scale deployment. By providing all these insights, we can consider the first research question answered.

The code-based implementation aligns well with academic curricula in computer science, data science or engineering, offering hands-on opportunities for students or participants to work with popular AI models such as YOLOv11 and PaddleOCR. Moreover, the development process involved critical thinking, debugging, algorithm design, testing tasks or refinement-oriented mentality, essential skills in today's technical education. On the other hand, the Roboflow platform can serve as an introductory tool for learners or entrepreneurs who do not yet possess advanced coding skills, as it lowers the barrier to entry, allowing rapid experimentation and visualization of AI workflows, which can be useful in non-technical disciplines or business schools seeking to embed digital innovation into their curriculum.

In entrepreneurship, the ability to prototype an AI system quickly using Roboflow could empower startups or the small and medium-sized enterprises in logistics, parking management or urban surveillance; however, scaling such prototypes into full-featured commercial systems may require transitioning to a more code-centric framework for performance optimization, a higher degree of customizability and data security. Overall, both approaches provide distinct kinds of educational value: low-code tools offers accessibly to AI concepts, while code-based systems provide depth or critical thinking, very suitable for higher-level education and innovation, enabling a more advanced analysis of the problem design or architecture. Thus, the second research question has been responded to.

The final research question tried to illustrate the broader societal consequences of adopting either development path; from an economic standpoint, low-code tools democratize AI access, potentially reducing the reliance on specialized technical employees and enabling smaller organizations to adopt smart technologies; this democratization, however, may create a dependency on external platforms, affecting long-term control over data and intellectual property. The code-based approach creates the premises of local capacity building, encouraging deeper use of the underlying technologies and the logic behind them. For educational institutions, this supports skill development aligned with high-demand job markets; for governments or civic

organizations, it enables long-term investment in open, modifiable systems without third-party lock-in and all the risks or the complications that may be associated with them, technically, business or perhaps pecuniary related.

The trade-off, therefore, lies in immediacy versus autonomy, because while low-code platforms accelerate prototyping and broaden participation, they may constrain innovation and impose hidden costs. Code-based systems, though slower to develop, offer greater long-term control, customization capability and alignment with local strategic goals in both education and public infrastructure. By making this juxtaposition, we can see a well defined answer to third research question.

5. CONCLUSIONS

This study compared a code-based YOLOv11n and PaddleOCR pipeline against a low-code, Roboflow-based pipeline for AI-driven license plate recognition, mainly focusing on their economic and educational trade-offs. The findings demonstrated that while the low-code approach substantially reduced initial development time and required minimal coding expertise, it came with the disadvantage of ongoing subscription costs and limited customization; in the same time, the code-based approach, despite demanding a significantly longer development effort, achieved a greater degree of control through its increased customizability, granting developers complete freedom over aspects such as model logic.

From an educational perspective, the code-based workflow is well-suited for advanced curricula, as it actively cultivates essential skills in problem-solving or critical-thinking; in contrast, the low-code solution proves highly effective for introducing AI concepts to non-technical audiences, thereby making it an ideal tool for entrepreneurship training and various interdisciplinary courses. Economically, while municipalities and small-to-medium enterprises in resource-constrained settings might find the low-code solution advantageous for rapid prototyping, they should ultimately consider migrating to a code-based system to ensure long-term scalability and data protection.

Future research should expand upon this analysis by incorporating multilingual plate recognition and critically evaluating hybrid approaches that aim to combine the rapid development of low-code platforms with the given flexibility of custom code.

REFERENCES

1. Businesswire (2023) *IDC Forecasts Revenue for Artificial Intelligence Software Will Reach $307 Billion Worldwide in 2027, https://www.businesswire.com/news/home/20231201614318/en/IDC-Forecasts-Revenue-for-Artificial-Intelligence-Software-Will-Reach-%24307-Billion-Worldwide-in-2027*.
2. Istiak, A. and Feng, W. (2024) "Enhancing Bangladeshi License Plate Recognition: A YOLOv8 Approach with Roboflow Integration for Accuracy and Speed Optimization," *International Journal for Research in Applied Science and Engineering Technology*, 12(5), pp. 1686–1699. Available at: https://doi.org/10.22214/ijraset.2024.61520.
3. Jamtsho, Y., Riyamongkol, P. and Waranusast, R. (2021) "Real-time license plate detection for non-helmeted motorcyclist using YOLO," *ICT Express*, 7(1), pp. 104–109. Available at: https://doi.org/10.1016/j.icte.2020.07.008.
4. Jenitta, J. *et al.* (2023) "License Plate Recognition Using Raspberry Pi," in *2023 World Conference on Communication & Computing (WCONF)*. IEEE, pp. 1–5. Available at: https://doi.org/10.1109/WCONF58270.2023.10235084.
5. Khachatryan, D. and Parise, S. (2025) "Teaching Real-time Object Detection with an Emphasis on Engagement and Inclusiveness," *Communications of the Association for Information Systems*, 56, pp. 613–632. Available at: https://doi.org/10.17705/1CAIS.05625.
6. PaddleOCR (2025) *PaddleOCR Documentation, https://www.paddleocr.ai/main/en/index.html*.
7. Prasad, N. *et al.* (2024) "Smart Pest Detection for an Agricultural Field Crop Based on Deep Learning," *Kelappaji College of Agricultural Engineering and Food Technology* [Preprint].
8. Roboflow (2025) *Pricing, https://roboflow.com/pricing*.
9. Shashirangana, J. *et al.* (2021) "Automated License Plate Recognition: A Survey on Methods and Techniques," *IEEE Access*, 9, pp. 11203–11225. Available at: https://doi.org/10.1109/ACCESS.2020.3047929.
10. Sukharevsky, A. *et al.* (2025) *The state of AI: How organizations are rewiring to capture value, https://www.mckinsey.com/capabilities/quantumblack/our-insights/the-state-of-ai#/*.