INT3404E 20 - Image Processing - Group 3
Sino-nom Character Localization Full Report

Hoang Duc Anh Tran Thi Van Anh
Nguyen Thi Thuy Huong Tran Hanh Uyen

1 Introduction

This report presents the project undertaken for the Mid-term Evaluation, focusing on the task of Sino-nom character localization. Sino-nom characters hold significant cultural and historical importance in Vietnamese heritage, serving as a bridge to the country's rich literary past. The primary objective of this project is to develop a robust system capable of accurately localizing Sino-nom characters in various text images, leveraging a provided dataset.

Background

Chữ Nôm (Sino-Nom), an ancient ideographic vernacular script, was adopted as the national script of Vietnam after its independence from China in 939 CE. Over the next millennium, from the 10th century to the 20th, Sino-Nom became the primary script for Vietnamese literature, philosophy, history, law, medicine, religion, and government policy.

One notable form of Sino-Nom documentation is the Sino-Nom woodblock. These woodblocks were created by carving the document content in reverse onto a wooden board, which was then used to print books. This method of printing was prevalent in the 19th and early 20th centuries. The contents of these woodblocks varied widely, encompassing official literature, classic texts, and historical books.

Sino-Nom woodblocks represent a significant cultural and historical heritage. They provide valuable insights into Vietnam's literary and administrative history and are crucial for the study and preservation of Vietnamese culture. However, the intricate and variable nature of Sino-Nom characters poses challenges for their digitization and automated localization, necessitating sophisticated image processing techniques to accurately interpret these ancient scripts.

Problem Statement

The task of this project is to design and implement an efficient method for localizing Sino-nom characters within text images. The provided dataset contains a diverse collection of images featuring these characters, which serves as the basis for training and evaluating our system. The project aims to address the following key objectives:

- Develop a preprocessing pipeline to handle the unique characteristics of the dataset.
- Implement and optimize machine learning algorithms to accurately localize Sino-nom characters.
- Evaluate the performance of the proposed system using appropriate metrics and compare it with existing approaches.

Motivation

This project offers a valuable opportunity to enhance our skills in image processing and machine learning. By tackling the problem of Sino-nom character localization, we gain practical experience in applying theoretical knowledge to real-world challenges. Additionally, this project allows us to contribute to the preservation of Vietnamese cultural heritage by facilitating the digitization and analysis of historical texts.

We would like to extend our gratitude to our lecturer for providing us with this meaningful project. The guidance and resources offered have been instrumental in our learning journey, enabling us to deepen our understanding of image processing techniques and their applications. Through this project, we have not only advanced our technical skills but also gained a deeper appreciation for Vietnamese culture and history.

2 Approaches Taken

2.1 Model selection

For object detection purpose, we selected the top-performing methods and built a pipeline for each one with the provided original dataset. These approaches include:

- Detectron2 has its own model zoo for computer vision models written in PyTorch.
- DETR is an end-to-end object detection model that combines a Transformer architecture with a convolutional neural network (CNN) backbone.
- YOLOv5 is a real-time object detection system that uses deep learning to identify and track objects in videos or images.
- YOLOv8 is the newest model in the YOLO algorithm series – the most well-known family of object detection and classification models in the Computer Vision (CV) field.
- YOLOv9 represents a significant advancement in real-time object detection, introducing breakthrough techniques such as Programmable Gain Information (PGI) and Generalized Efficient Layer Aggregation Network (GELAN).

2.2 Result and conclusion

In our initial evaluation, we employed the Average Precision (AP) metric, a common metric in object detection tasks to measure precision and recall, to assess the performance of various object detection models. The models under evaluation included Detectron2, DETR, YOLOv5, YOLOv8, and YOLOv9. Their mAP scores were approximately 0.4, 0.3, 0.7, 0.8, and 0.8 respectively. Despite being the newest model, YOLOv9's optimization process is less mature compared to YOLOv8. This resulted in a significantly longer training time for YOLOv9, even though it achieved a comparable mAP of 0.8. Based on this finding, we decided to focus further development efforts on YOLOv8 due to its superior balance between performance and training efficiency. Additionally, the input and output of the model have the same format as required by the task, as well as easy customization capabilities and a simple pipeline.

3 Analysis

3.1 Data Analysis

After analyzing the provided dataset, we reached the following conclusions:

- **Data size:** The dataset, comprising 70 images with a total of 13,669 annotated objects, demonstrates significant complexity and richness. The high object density per image suggests the presence of challenging scenarios that necessitate careful consideration during model training and evaluation. To enhance model performance and robustness, we need to consider applying data augmentation techniques such as geometric transformations, photometric adjustments, noise addition, and synthetic data generation (e.g., MixUp, CutMix).
- **Visualize a few sample images with annotated bounding boxes:** This process involves verifying the accuracy of the bounding box annotations to ensure they correctly enclose the intended objects without any misalignments or omissions. Additionally, this visualization provides insights into the diversity of Sino-nom character sizes and highlights special cases in the dataset, such as text with white color on a black background (contrasting with the more common black text on a white background) and instances of text with inkblots.

- **Object Distribution:** Most aspect ratios are clustered around 1, meaning that the objects are generally close to being square. There are fewer objects with higher aspect ratios, and the frequency diminishes rapidly beyond a ratio of 2, indicating that elongated objects are rare in this dataset. To address this imbalance, generating synthetic examples of objects with higher aspect ratios can help the model learn to detect them more reliably. It's essential to consider these special cases and plan for them by augmenting the data to ensure the model performs well.

3.2 Error Analysis

Our evaluation of YOLOv8 for text detection revealed several error categories: (1) missing detections for small text, large text, and blurry text; (2) bounding boxes inadequately encompassing large text; and (3) misidentification of specific non-standard text characters, such as the word '-' commonly used to represent the word "First." While YOLOv8 demonstrates proficiency in general object detection, these text-specific errors likely stem from limitations in the training data. The model might not have been exposed to a sufficient variety of text characteristics, including:

- **Variations in text size:** extremely small or large text.
- **Text quality:** blurry or degraded text.
- **Special characters:** white character on squared black background.
- **Stained background:** background contains dust-staining

4 Proposed method: YOLOv8

4.1 Implementation tools and environments

This project utilized the following tools and technologies:

- **Deep Learning Framework:** While a specific deep learning framework like TensorFlow or PyTorch wasn't directly used, the project leveraged the functionalities of the **ultralytics** library. This library provides pre-trained YOLO models (including YOLOv8) and functionalities for object detection tasks.
- **Google Colab:** This cloud-based platform provided a convenient environment for running the **ultralytics** library and training the YOLOv8 model for text detection. Colab offers pre-configured virtual machines with GPUs, facilitating efficient training for deep learning models.

Code structure and repository organization

4.2 Data Augmentation

After validating the model's performance, we decided to apply techniques for solving these problems: characters too small, white characters on black backgrounds, yī characters, and stained images.
We used Albumentation - a powerful open-source library for data augmentation. Albumentation offers numerous pre-defined operations for dataset transformation, along with the option to customize your own techniques.

4.3 Model pipeline

1. Environment Setup
    - Ensure all necessary libraries and frameworks are installed like Python, PyTorch, ultralytics and the YOLOv8 package.

- Configure GPU settings if available to speed up the training process.
- Create and activate a virtual environment to manage dependencies.

2. Dataset Preparation
   - Apply data augmentation techniques to increase the diversity of the training data. This might include rotating, flipping, or scaling images.
3. Model Configuration
   - Set up the YOLOv8 configuration file. This includes specifying the number of classes (in this case, the different Sino-Nom characters), input image size, and other hyperparameters.
   - Train the YOLOv8 model on the prepared dataset. This involves setting training parameters such as learning rate, batch size (batch = 3), and the number of epochs (epochs = 40), image size (imgsize = 928).
   - Keep track of the training process using metrics like loss, precision, recall, and mAP (mean Average Precision). Adjust hyperparameters as needed.
4. Evaluate and Save the Model Evaluate the model on the test set to check its performance. Use metrics like precision, recall, F1 score, and mAP to assess how well the model is localizing Sino-Nom characters.

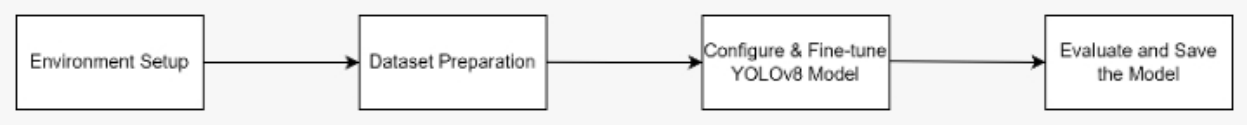

Figure 1: Model pipeline

5 Results

Our results, presented in Table 1, demonstrate that the YOLOv8 model performs consistently well across various training configurations on the Sino-nom dataset.

- Across all runs, the model demonstrates high precision and recall, indicating reliable detection capabilities.
- Combining all subsets and including additional data leads to better mAP values, highlighting the importance of diverse and comprehensive training data for enhancing localization performance.
- The model maintains efficient run times, making it suitable for real-time applications.
- While individual subsets maintain high precision and recall, they slightly lag in mAP compared to the original and combined datasets, suggesting that the full dataset's diversity is crucial for optimal performance.

6 Conclusion

The YOLOv8 model, when fine-tuned with a comprehensive and diverse dataset, exhibits excellent localization performance with high precision, recall, and mAP values, while maintaining computational efficiency. This is why we chose the YOLOv8 model for object localization tasks in our midterm evaluation.

Table 1: Results

RUN	Training dataset	Stage	Result			run in script
			P	R	mAP	
RUN 0	**wb_localization_dataset_original**	train	0.992	0.993	0.865	**0.85**
		val	0.993	0.994	0.879	
RUN 1.1	**wb_localization_dataset_set1**	train	0.992	0.994	0.87	0.859
		val	0.995	0.992	0.881	
RUN 1.2	**wb_localization_dataset_set2**	train	0.995	0.993	0.875	0.856
		val	0.995	0.994	0.887	
RUN 1.3	**wb_localization_dataset_set3**	train	0.992	0.993	0.866	0.841
		val	0.994	0.992	0.88	
RUN 1.4	**wb_localization_dataset_set4**	train	0.993	0.995	0.868	0.849
		val	0.994	0.993	0.883	
RUN 2	**wb_localization_dataset_all**	train	0.994	0.994	0.877	0.858
		val	0.993	0.994	0.888	
RUN 3	**wb_localization_dataset_more**	train	0.993	0.996	0.879	**0.86**
		val	0.994	0.995	0.89	

7 Contribution

7.1 Task Descriptions

- Nguyen Thi Thuy Huong
  - Implementing a pipeline in Detectron2
  - Conducting error analysis
  - Data Augmentation using this dataset
  - Reviewing and refining documentation for clarity and completeness.
- Hoang Duc Anh
  - Implementing, testing and tuning YOLOv9 model
  - Implementing, testing and tuning YOLOv8(V8L, V8M) model
  - Training and tuning the model using augmented data
  - Evaluating data quality and selecting data for augmentation to enhance model performance
- Tran Thi Van Anh
  - Implementing a pipeline in DETR
  - Sino-nom data analysis
  - Documenting the entire project workflow and methodologies
  - Evaluating data quality and selecting data for augmentation to enhance model performance
- Tran Hanh Uyen
  - Implementing a pipeline in YOLOv5

  - Gathering and curating datasets for training
  - Filtering augmented data for model training process

7.2 Self-Evaluation of Contributions

- Hoang Duc Anh: 27 %
- Tran Thi Van Anh: 26.5 %
- Nguyen Thi Thuy Huong: 26.5 %
- Tran Hanh Uyen: 20 %