Project Document: Invoice Automatic Scanning and Text Predictions

1. Business Overview

Background

In the era of digital transformation, businesses have transitioned from paper invoices to digital invoices. However, reconciling these digital invoices remains a manual task, consuming significant time and resources. Automating this process can substantially enhance efficiency, accuracy, and productivity.

Objective

This project aims to automate the extraction of key information from digital invoices, focusing on three critical classes:

1. Invoice Number
2. Billing Date
3. Total Amount

Applications

The solution is applicable across various industries that manually process invoices and record details in ledgers, including finance, retail, healthcare, and logistics.

2. Solution Methodology

Approach

The approach involves leveraging Object Detection and Optical Character Recognition (OCR) techniques to automate the extraction of key details from invoices.

Steps:

1. **Setting up YOLOv4 Environment:**
    - Download AlexeyAB's YOLOv4 repository.
    - Adjust the Makefile to enable OpenCV and GPU for Darknet, then build Darknet.
2. **Downloading Pre-trained YOLOv4 Weights:**
    - Use pre-trained weights from the COCO dataset (trained on 80 classes) for initial predictions.

3. **Creating Display Functions:**
    - Develop functions to display the predicted classes on images for verification.
4. **Data Collection and Labeling:**
    - Collect a dataset of invoice images.
    - Label the images using LabelImg for precise annotation.
5. **Configuring Files for Training:**
    - Modify the custom .cfg file, obj.data, obj.names, train.txt, and test.txt files.
        - obj.names: Lists the classes to be detected.
        - obj.data: Contains paths and configurations for training.
        - train.txt and test.txt: Contain file paths for training and testing datasets.
6. **Download Pre-trained Weights:**
    - Acquire pre-trained weights for convolutional layers to accelerate training.
7. **Training Custom Object Detector:**
    - Train the YOLOv4 model on the labeled dataset.
8. **Evaluating the Model:**
    - Assess the model's performance using Mean Average Precision (mAP).
9. **Predicting Image Classes and Storing Coordinates:**
    - Use the trained model to predict classes and save coordinates of detected elements.
10. **Detecting Text from Predicted Classes:**
    - Import and configure Tesseract OCR.
    - Extract text using the Tesseract pre-trained LSTM model and fine-tune it for better accuracy.

3. Technology Stack

Programming Language

- Python

Object Detection

- YOLO V4

Text Recognition

- Tesseract OCR

Environment

- PyCharm

4. Implementation Details

Setting Up and Installation

- Clone the YOLOv4 repository from GitHub.
- Configure the Makefile for OpenCV and GPU support.
- Build Darknet.
- Set up the PyCharm environment by configuring the Python interpreter and installing the necessary packages.

Data Labeling

- Collect invoice images and label them using LabelImg for accurate annotations.

Training Configuration

- Adjust custom .cfg, obj.data, obj.names, train.txt, and test.txt files for the specific project needs.
- Download and integrate pre-trained convolutional layer weights.

Training Process

- Train the YOLOv4 model with the labeled dataset.
- Evaluate the model using Mean Average Precision (mAP).

Text Extraction

- Configure Tesseract OCR for text extraction.
- Extract and fine-tune text predictions using the Tesseract LSTM model.

5. Project Workflow

Phase 1: Setup and Installation

- **Task:** Clone the YOLOv4 repository, adjust the Makefile, and build Darknet.
- **Deliverables:** Configured YOLOv4 environment ready for object detection tasks.

Phase 2: Data Preparation

- **Task:** Collect and label invoice images using LabelImg.
- **Deliverables:** Labeled dataset ready for training.

Phase 3: Training Configuration

- **Task:** Modify configuration files and integrate pre-trained weights.
- **Deliverables:** Configured YOLOv4 model ready for training.

Phase 4: Model Training

- **Task:** Train the custom object detector using the prepared dataset.
- **Deliverables:** Trained YOLOv4 model.

Phase 5: Model Evaluation

- **Task:** Evaluate the model using Mean Average Precision (mAP).
- **Deliverables:** Evaluation metrics and performance report.

Phase 6: Text Extraction

- **Task:** Configure Tesseract OCR and fine-tune the LSTM model for text extraction.
- **Deliverables:** Extracted text from predicted classes with high accuracy.

Phase 7: Integration and Testing

- **Task:** Integrate the trained model and OCR system into the application, perform end-to-end testing.
- **Deliverables:** Fully functional invoice processing system.

6. Conclusion

Summary

The Invoice Automatic Scanning and Text Predictions project automates the extraction of key information from digital invoices, enhancing efficiency and reducing manual effort. Leveraging YOLOv4 for object detection and Tesseract OCR for text extraction, the solution is robust and scalable across various industries.

Future Enhancements

- Enhance the model to handle diverse invoice formats and layouts.
- Integrate advanced machine learning models to improve accuracy and extraction capabilities.
- Automate the entire pipeline from invoice upload to ledger entry in a seamless manner.