# Chương 3: Thực nghiệm và đánh giá mô hình

## 3.1 Chuẩn bị dữ liệu huấn luyện

Sử dụng tool sinh ảnh [TextRecognitionDataGenerator](https://github.com/Belval/TextRecognitionDataGenerator), của Belval. TextRecognitionDataGenerator (TRDG) là một công cụ mã nguồn mở trên GitHub, được thiết kế để tạo ra dữ liệu hình ảnh văn bản tổng hợp nhằm huấn luyện các hệ thống nhận dạng ký tự quang học (OCR) một cách tùy chỉnh.

TRDG giúp tạo ra các hình ảnh văn bản tổng hợp với nhãn chính xác, hỗ trợ huấn luyện và tinh chỉnh các mô hình OCR. Công cụ này đặc biệt hữu ích khi dữ liệu thực tế khan hiếm hoặc không đủ đa dạng.

Mục tiêu sử dụng cung cụ này để thực hiện sinh ảnh văn bản tiếng việt với mục tiêu là 1000 ảnh.

**Ưu điểm**

* **Tạo dữ liệu nhanh chóng và đa dạng**: TRDG cho phép tạo ra hàng nghìn hình ảnh văn bản với các tuỳ chọn về phông chữ, màu sắc, kích thước, khoảng cách giữa các ký tự và hiệu ứng nhiễu như làm mờ hoặc biến dạng.
* **Hỗ trợ nhiều ngôn ngữ**: Công cụ hỗ trợ cả văn bản không phải chữ Latin, bao gồm tiếng Việt, giúp mở rộng phạm vi ứng dụng.
* **Dễ dàng tích hợp và sử dụng**: TRDG được viết bằng Python và có thể cài đặt dễ dàng thông qua pip, phù hợp với nhiều hệ điều hành.
* **Tự động gán nhãn**: Vì dữ liệu được tạo ra từ văn bản đầu vào, nhãn cho mỗi hình ảnh được tự động gán, tiết kiệm thời gian và công sức so với việc gán nhãn thủ công.

**Nhược điểm**

* **Thiếu tính chân thực**: Dữ liệu tổng hợp có thể không phản ánh đầy đủ các biến thể và phức tạp của dữ liệu thực tế, như ánh sáng không đồng đều, nhiễu nền hoặc các yếu tố môi trường khác.
* **Hạn chế về phông chữ và ngôn ngữ**: Mặc dù hỗ trợ nhiều ngôn ngữ, nhưng việc tạo dữ liệu cho các ngôn ngữ đặc biệt hoặc phông chữ hiếm có thể gặp khó khăn nếu không có sẵn trong hệ thống.

**Ý nghĩa**

TRDG là một công cụ quan trọng trong việc phát triển và cải thiện các hệ thống OCR, đặc biệt là khi dữ liệu thực tế không đủ hoặc không đa dạng. Nó cho phép các nhà nghiên cứu và phát triển tạo ra các bộ dữ liệu tùy chỉnh để huấn luyện mô hình, từ đó nâng cao độ chính xác và khả năng tổng quát hóa của các hệ thống nhận dạng văn bản.

### Thực hiện sinh dữ liệu

Theo thuật toán sinh ảnh ta sẽ nhận được một file ảnh và một file label txt tương ứng. Kịch bản là dữ liệu đầu vào là ảnh của đoạn văn bản word đầu ra là một file txt là kêt quả tương ứng với van bản tỏng ảnh nhận được.

Ảnh văn bản.tif:

A black text on a white background

AI-generated content may be incorrect.

Label có dạng name.gt.txt:

A black and white text

AI-generated content may be incorrect.

## 3.2 Training

Thực hiện huấn luyện Tesseract với custom dataset đã được chuẩn bị từ bước trước. Ta được mẫu log sau:

A screenshot of a computer code

AI-generated content may be incorrect.

A screenshot of a video game

AI-generated content may be incorrect.

* Total weights = 392755: Số lượng trọng số trong mạng LSTM.
* Learning rate = 0.002: Tốc độ học (learning rate) khá nhỏ – giúp mô hình học ổn định.
* Momentum = 0.5: Tăng cường cập nhật trọng số dựa vào lịch sử gradient.

**Xét mỗi vòng lặp huấn luyện:**

At iteration 3700/3700/3700, mean rms=5.891%, delta=54.028%, BCER train=95.935%, BWER train=98.467%, skip ratio=0.000%, New worst BC

|  |  |
| --- | --- |
| Thành phần | Ý nghĩa |
| Iteration 3700/3700/3700 | Vòng lặp hiện tại / số vòng sau checkpoint / tổng số vòng lặp tối đa. |
| mean rms=5.891% | Root Mean Square Error trung bình – sai số trung bình trong dự đoán ký tự. Càng thấp càng tốt. |
| delta=54.028% | Mức thay đổi rms so với checkpoint trước. Giúp theo dõi sự tiến triển |
| BCER train=95.935% | Character Error Rate (CER) – phần trăm ký tự nhận đúng. |
| BWER train=98.467% | Word Error Rate (WER) – phần trăm từ nhận đúng. |
| skip ratio=0.000% | Tỷ lệ mẫu bị bỏ qua trong quá trình training. Thường do lỗi input. |
| New worst BC | Đánh dấu đây là mô hình tệ nhất mới (nếu BCER tăng lên). Nếu không có, thì mô hình đang cải thiện. |
| wrote checkpoint | Khi gặp dòng này, mô hình hiện tại được lưu lại thành checkpoint. Tesseract ghi lại trạng thái model hiện tại vì nó đạt thành tích tốt hơn về BCER. |

Sau khi chạy xong traning   
Finished! Selected model with minimal training error rate (BCER) = 95.575

BCER = 95.575 → Đây là Best Character Error Rate, chính xác hơn là:

* 95.575% là tỷ lệ ký tự được nhận diện đúng trong tập huấn luyện.
* Tương đương với Character Accuracy = 95.575%, hay ngược lại CER ≈ 4.425%.

Kết luận từ kêt quả training trên ta có:

* Character Accuracy (độ chính xác ký tự): 95.575% là khá tốt, chứng tỏ mô hình đã học được đáng kể từ dữ liệu huấn luyện.
* CER (Character Error Rate): ~4.425% là chấp nhận được trong nhiều ứng dụng OCR, đặc biệt nếu dữ liệu đầu vào có nhiễu, font lạ, hoặc hình ảnh chất lượng không cao.

Qúa tình tranning ta được kết quả trả về cho từng ảnh tỏng tập train như sau:

A screenshot of a computer

AI-generated content may be incorrect.

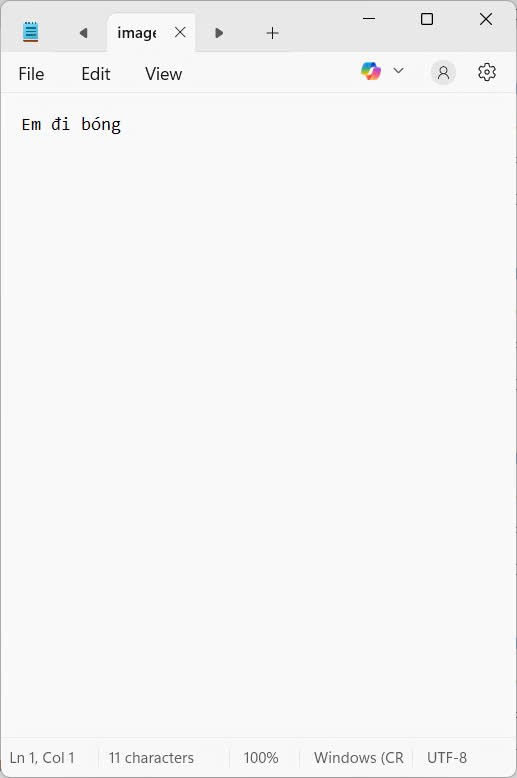
|  |  |
| --- | --- |
| **Tên file** | **Ý nghĩa** |
| imageXXXX.tif | Đây rất có thể là ảnh gốc được sử dụng làm dữ liệu đầu vào cho quá trình huấn luyện hoặc đánh giá mô hình OCR. Định dạng TIF thường được sử dụng cho hình ảnh chất lượng cao và có thể chứa nhiều trang hoặc thông tin bổ sung. |
| imageXXXX.gt.txt | Đây là tệp "Ground Truth" (GT). Nó chứa nội dung văn bản chính xác tương ứng với ảnh imageXXXX.tif. Tệp này được sử dụng làm nhãn (label) trong quá trình huấn luyện để mô hình học cách nhận diện ký tự và từ. |
| imageXXXX.box | Đây là tệp chứa thông tin về bounding box (khung bao quanh) của từng ký tự hoặc từ trong ảnh imageXXXX.tif. Mỗi dòng trong tệp này thường chỉ định tọa độ (x\_min, y\_min, x\_max, y\_max) và ký tự tương ứng. Tệp này cũng rất quan trọng trong quá trình huấn luyện để mô hình học cách định vị các ký tự. |
| imageXXXX.tr | Đây là tệp chứa thông tin về bounding box (khung bao quanh) của từng ký tự hoặc từ trong ảnh imageXXXX.tif. Mỗi dòng trong tệp này thường chỉ định tọa độ (x\_min, y\_min, x\_max, y\_max) và ký tự tương ứng. Tệp này cũng rất quan trọng trong quá trình huấn luyện để mô hình học cách định vị các ký tự. |
| imageXXXX.lstmf | Đây cũng là một loại tệp đặc trưng được tạo ra từ ảnh, nhưng có khả năng được tối ưu hóa hoặc định dạng đặc biệt cho các mô hình dựa trên mạng nơ-ron hồi quy dài-ngắn hạn (LSTM), một kiến trúc phổ biến trong OCR hiện đại. Nó có thể là phiên bản mới hơn hoặc chứa thông tin khác so với tệp .tr. |

Kết quả bounding box:

A screenshot of a computer

AI-generated content may be incorrect.

Kết quả sau text khi train:



## 3.3 Kiểm thử

Kiểm thử model

Sau khi huấn luyện (training) xong mô hình Tesseract với dữ liệu tùy chỉnh (custom dataset), bạn cần **đánh giá trên tập validation/test** để kiểm tra **độ chính xác**, **overfitting**, hoặc các lỗi khác. Dưới đây là **quy trình đầy đủ** để thực hiện điều đó:

**✅ BƯỚC 1: Chuẩn bị tập test (validation)**

Bạn cần có:

* Một thư mục chứa ảnh test (test-images/)
* Mỗi ảnh phải có một file ground truth .gt.txt tương ứng. Ví dụ:

css

CopyEdit

test-images/

├── sample1.tif

├── sample1.gt.txt ← chứa đúng nội dung của ảnh

├── sample2.tif

├── sample2.gt.txt

**✅ BƯỚC 2: Chạy mô hình đã huấn luyện trên test set**

Bạn sử dụng lệnh sau để chạy Tesseract và tạo ra output để so sánh:

bash

CopyEdit

tesseract test-images/sample1.tif output\_sample1 --oem 1 --psm 6 -l my-lang

* --oem 1: sử dụng LSTM engine
* --psm 6: page segmentation mode (thường dùng cho dòng văn bản)
* -l my-lang: sử dụng mô hình my-lang.traineddata (trong tessdata)

Sau khi chạy, bạn sẽ có file output\_sample1.txt là kết quả Tesseract OCR.

**✅ BƯỚC 3: So sánh output với ground truth**

Cách đơn giản là dùng công cụ compute-wer.py (hoặc viết script Python so sánh ký tự):

**➤ Cài công cụ tính CER/WER:**

Bạn có thể dùng jiwer (Word Error Rate) hoặc python-Levenshtein:

bash

CopyEdit

pip install jiwer

**➤ Ví dụ script:**

python

CopyEdit

from jiwer import wer, cer

with open("test-images/sample1.gt.txt", "r", encoding="utf-8") as f:

ground\_truth = f.read().strip()

with open("output\_sample1.txt", "r", encoding="utf-8") as f:

prediction = f.read().strip()

print("WER:", wer(ground\_truth, prediction))

print("CER:", cer(ground\_truth, prediction))

**✅ BƯỚC 4: Lặp lại cho nhiều ảnh test**

* Bạn nên viết script chạy batch để xử lý tất cả .tif trong thư mục test-images/
* Tính **trung bình CER/WER** cho cả tập test

**✅ BƯỚC 5: So sánh kết quả**

| **Chỉ số** | **Ý nghĩa** |
| --- | --- |
| CER (Character Error Rate) | Càng thấp càng tốt (≤5% là ổn) |
| WER (Word Error Rate) | Càng thấp càng tốt |
| Nếu CER\_train ≈ CER\_test | Mô hình **tốt**, không bị overfitting |
| Nếu CER\_train << CER\_test | Mô hình **bị overfitting** |

**❗ Mẹo:**

Nếu bạn có data/my-lang/list.eval đã tách sẵn lúc train thì bạn có thể:

bash

CopyEdit

lstmeval --model my-lang\_checkpoint \

--eval\_listfile data/my-lang/list.eval \

--traineddata data/my-lang/my-lang.traineddata

Bạn có sẵn tập test và ground truth chưa? Nếu cần mình giúp viết script tự động cho bạn luôn.