

Bài: Train model giải normal captcha với google colab

Xem bài học trên website để ủng hộ Kteam: [Train model giải normal captcha với google colab](#)

Mọi vấn đề về lỗi website làm ảnh hưởng đến bạn hoặc thắc mắc, mong muốn khóa học mới, nhằm hỗ trợ cải thiện Website. Các bạn vui lòng phản hồi đến Fanpage [How Kteam](#) nhé!

Mọi người vẫn nghĩ AI là điều gì đó rất to lớn, khó khăn. Nay Kteam sẽ giúp bạn tiếp cận với AI từng bước theo cách đơn giản nhất. Đến người không biết lập trình còn có thể làm được.

Bài này chúng ta sẽ cùng nhau train một model AI giúp nhận diện chữ từ hình. Cụ thể là dạng text captcha.

Như hình này:



Chúng ta cần phải đưa ra kết quả text là: 11438 để nhập vào ô captcha.

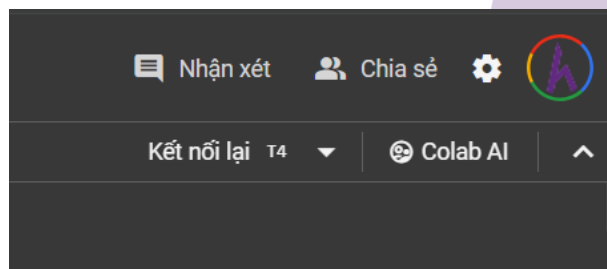
Vậy để có thể train một model cho phép OCR captcha như thế này sẽ phải làm gì khi bạn không biết cả lập trình?

Ở bài này chúng ta sẽ làm quen sử dụng google colab và thực hiện step by step để giải được captcha.

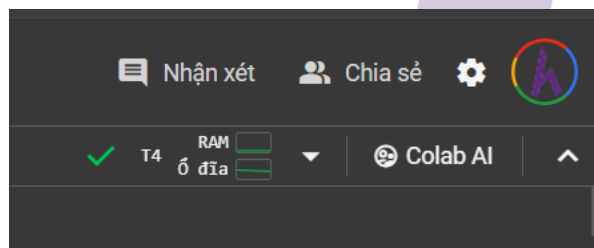
vào link này:

[Train model OCR giải normal captcha](#)

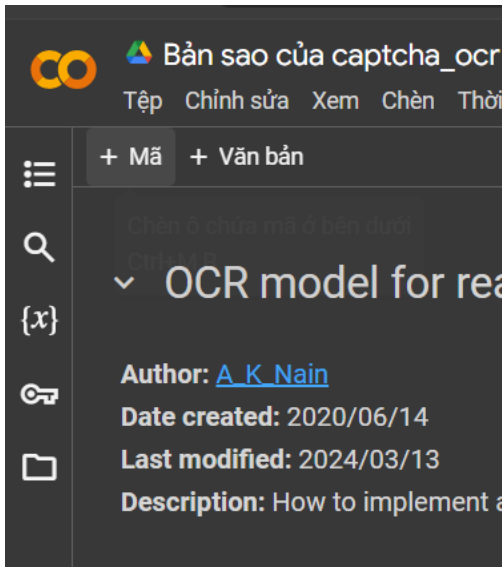
Nhấn vào kết nối tài nguyên ở góc trên phải màn hình:



Khi bạn đã kết nối được với 1 tài nguyên thì sẽ trông như thế này:



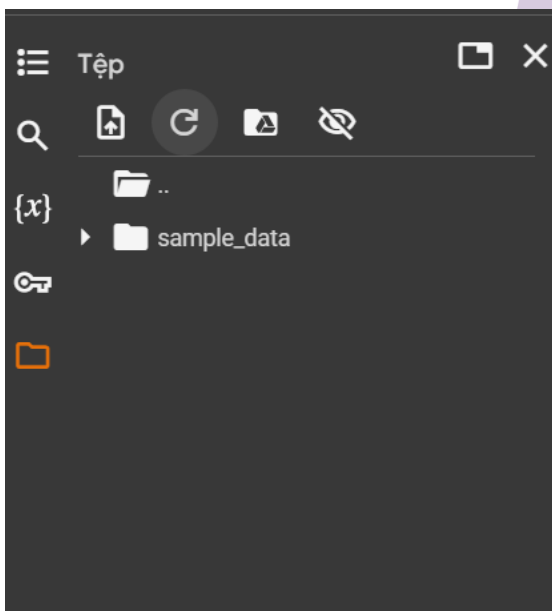
Sau đó nhìn qua phía tay trái của màn hình:



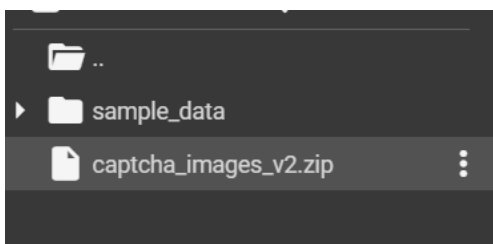
Nhấn vào icon folder



Dialog folder sẽ show ra:




Nắm kéo file captcha_images_v2.zip vào dialog để upload.




Bạn có thể tự chuẩn bị data mẫu theo cấu trúc của file captcha_images_v2.zip. Bạn có thể tải folder mẫu ở phần Library ở cuối bài này.

Bắt đầu nhấn vào từng icon  ở từng phần mã.

 `pip install --upgrade keras`

Setup



```
import os

os.environ["KERAS_BACKEND"] = "tensorflow"

import numpy as np
import matplotlib.pyplot as plt

from pathlib import Path
from collections import Counter

import tensorflow as tf
import keras
from keras import ops
from keras import layers
```

Thực hiện tất cả các icon như vậy. Việc này chính là việc chạy đoạn mã đó.

Ở phần training bạn sẽ thấy loss và val_loss khá cao.

Training

```
[ ]
# TODO restore epoch count.
epochs = 1000
early_stopping_patience = 100
# Add early stopping
early_stopping = keras.callbacks.EarlyStopping(
    monitor="val_loss", patience=early_stopping_patience, restore_best_weights=True
)

# Train the model
history = model.fit(
    train_dataset,
    validation_data=validation_dataset,
    epochs=epochs,
    callbacks=[early_stopping],
)
```

```
Epoch 1/1000
26/26 ————— 12s 459ms/step - loss: 205.1698 - val_loss: 196.9651
Epoch 2/1000
26/26 ————— 6s 249ms/step - loss: 204.7010 - val_loss: 198.0834
Epoch 3/1000
26/26 ————— 10s 232ms/step - loss: 204.3675 - val_loss: 198.6482
Epoch 4/1000
26/26 ————— 10s 208ms/step - loss: 204.3421 - val_loss: 199.3812
Epoch 5/1000
26/26 ————— 10s 209ms/step - loss: 203.8372 - val_loss: 200.5355
```

Nhưng đừng lo. Sau khi train tầm 100 lượt Epoch thì loss sẽ còn dưới 1 thôi:

```
[ ] Epoch 143/1000
26/26 ————— 6s 247ms/step - loss: 0.2712 - val_loss: 61.6251
Epoch 144/1000
26/26 ————— 9s 212ms/step - loss: 0.4417 - val_loss: 66.7017
Epoch 145/1000
26/26 ————— 10s 213ms/step - loss: 0.2956 - val_loss: 69.3453
Epoch 146/1000
26/26 ————— 6s 248ms/step - loss: 0.3014 - val_loss: 77.4799
Epoch 147/1000
26/26 ————— 10s 249ms/step - loss: 0.5597 - val_loss: 80.9648
Epoch 148/1000
26/26 ————— 6s 211ms/step - loss: 0.2869 - val_loss: 80.0241
Epoch 149/1000
26/26 ————— 10s 211ms/step - loss: 0.3647 - val_loss: 109.4136
Epoch 150/1000
26/26 ————— 11s 233ms/step - loss: 1.5601 - val_loss: 87.9703
Epoch 151/1000
26/26 ————— 11s 253ms/step - loss: 2.1019 - val_loss: 82.5539
Epoch 152/1000
26/26 ————— 6s 214ms/step - loss: 0.3664 - val_loss: 82.2390
Epoch 153/1000
26/26 ————— 10s 211ms/step - loss: 1.4864 - val_loss: 78.3941
```

Bạn có thể nhấn lại icon start để stop tiến trình nếu đợi lâu.

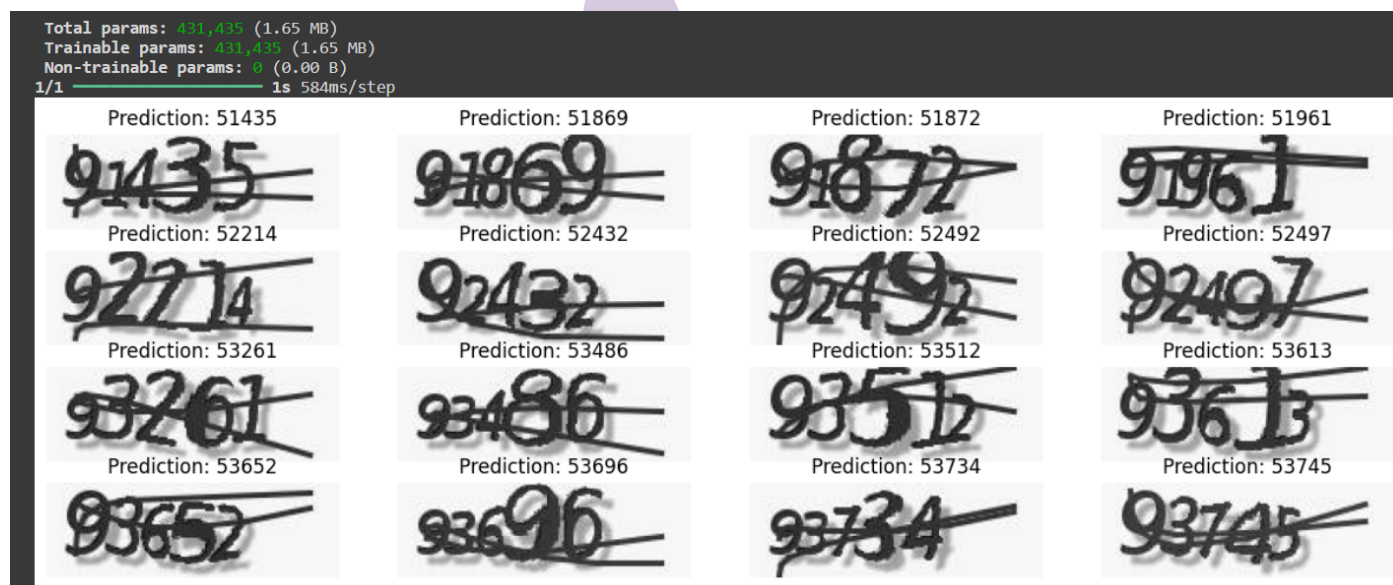
Cuối cùng có thể test model đã train ở phần Inference

▼ Inference

You can use the trained model hosted on [Hugging Face Hub](#) and try the demo on [Hugging Face Spaces](#).

```
def ctc_decode(y_pred, input_length, greedy=True, beam_width=100, top_paths=1):
    input_shape = ops.shape(y_pred)
    num_samples, num_steps = input_shape[0], input_shape[1]
    y_pred = ops.log(ops.transpose(y_pred, axes=[1, 0, 2]) + keras.backend.epsilon())
    input_length = ops.cast(input_length, dtype="int32")
    if greedy:
```

Và đây là kết quả của 450 hình data đầu vào:



Bạn có thể tăng số data đầu vào lên 1000-2000 thì độ chính xác sẽ tăng lên đáng kể.

Có thể train nhiều lần để loss dưới 1 và val_loss giảm thật nhiều. Khi loss và val_loss nhỏ thì chính xác sẽ tăng.

Cuối cùng bạn có thể chạy code Lưu model để tạo ra file model.h5. nhấp phải vào model ở bảng folder để tải về và tùy ý sử dụng.

Cảm ơn các bạn đã theo dõi bài viết. Hãy để lại bình luận hoặc góp ý của mình để phát triển bài viết tốt hơn. Đừng quên **"Luyện tập – Thử thách – Không ngại khó"**.