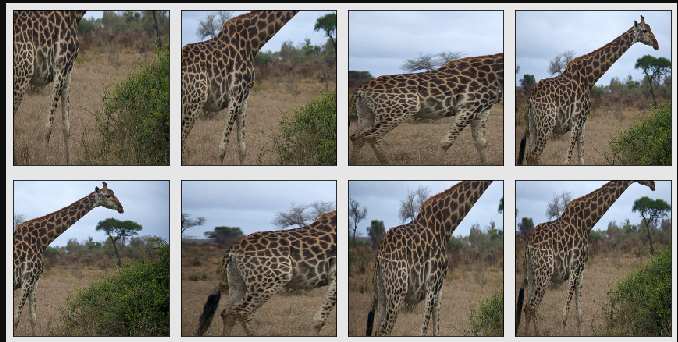
**Đánh giá**

1. **Cifar 10**

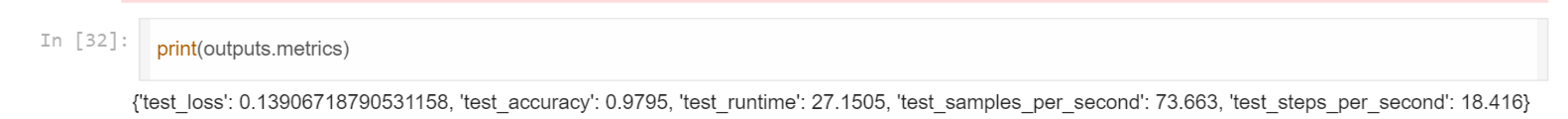
* Đây là datasets có kích thước (60000, 32, 32, 3). Feature 1 là ảnh, feature 2 là tên của ảnh(nhãn). Nhãn là categorical, gồm 10 lớp: airplane, automobile, bird, cat, deer, dog, frog, horse, ship, và truck.
* Ở đây mô hình sẽ lấy tập train từ dòng 0 đến 5000, tập test lấy từ dòng 0 đến 2000 để thử nghiệm.
  1. **ViT(mô hình ViT-B/16)**
  2. đọc dữ liệu
  3. Xử lý dữ liệu đầu vào
* Chuyển các giá trị pixel của ảnh từ khoảng [0:255] thành [0:1]
  1. chia tập train và test
  2. chia tập train thành 2 tập train và valid, valid 10%
  3. Sử dụng mô hình tiền huấn luyện của Google với dữ liệu google/vit-base-patch16-224-in21k (giúp tiết kiệm thời gian tiền huấn luyện, in21k là data của imageNet21k, có data phong phú, đa dạng, giúp mô hình chính xác hơn).
  4. Chuẩn hóa dữ liệu đầu vào
* RandomResizedCrop(size): Phép biến đổi cắt ngẫu nhiên một phần của hình ảnh với kích thước ngẫu nhiên, sau đó thay đổi kích thước ảnh đó để có kích thước nhất định. Điều này giúp mô hình học được các đặc trưng từ các phần khác nhau của ảnh và làm cho mô hình trở nên đa dạng hơn.



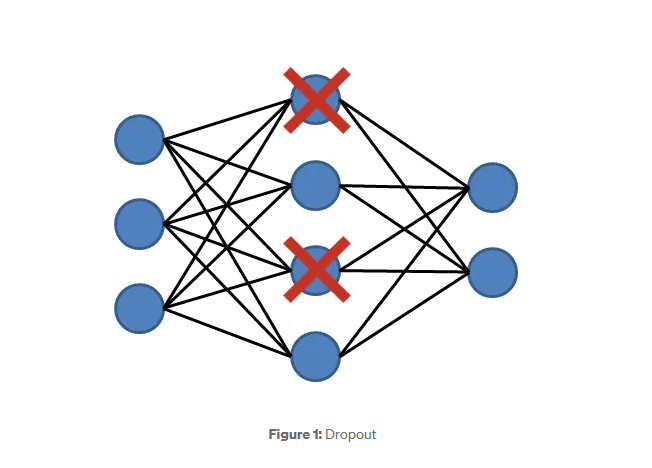
* RandomHorizontalFlip(): Phép lật ngẫu nhiên ảnh theo chiều ngang. Điều này giúp tăng cường dữ liệu bằng cách tạo ra các biến thể của ảnh mà không ảnh hưởng đến nhãn.



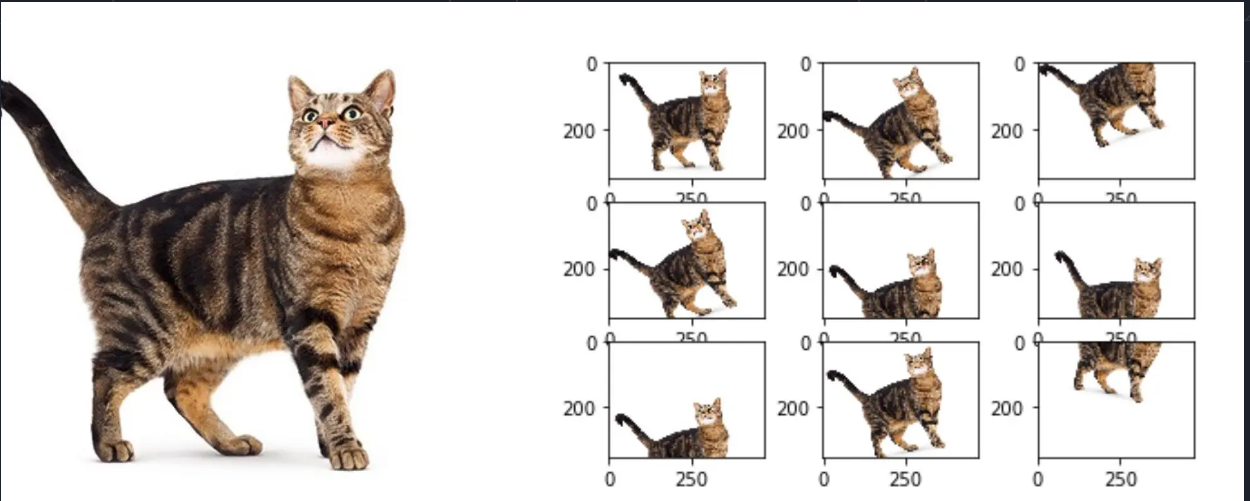
* ToTensor(): Chuyển đổi ảnh từ định dạng hình ảnh sang tensor. Đây là bước cần thiết để cung cấp dữ liệu cho mạng nơ-ron.
* normalize: Phép chuẩn hóa ảnh theo giá trị trung bình và độ lệch chuẩn đã được xác định trước. Điều này giúp cân bằng dữ liệu và tăng tốc độ hội tụ của mô hình(sử dụng trung bình và độ lệch chuẩn của mô hình tiền huấn luyện của Google)
  1. Định nghĩa mô hình
  2. Huấn luyện
  3. Kiểm tra kết quả

****

* 1. **CNNs(mô hình VGG)**
     1. Đọc dữ liệu
     2. Xử lý dữ liệu đầu vào
* Chuyển các giá trị pixel của ảnh từ khoảng [0:255] thành [0:1]
  + 1. Chia tập train và test
    2. Sử dụng tập test làm valid
    3. Sử dụng 3 lớp VGG, sử dụng dropout regularization, Data Augmentation và Batch Normalization
* Dropout regularization là một kỹ thuật phổ biến được sử dụng để ngăn chặn overfitting trong quá trình huấn luyện mạng nơ-ron.
* Trong quá trình huấn luyện, mỗi nút trong lớp ẩn của mạng nơ-ron sẽ có một xác suất dropout, nghĩa là có một tỷ lệ xác định của các nút sẽ bị "tắt" ngẫu nhiên trong mỗi lần lan truyền ngược.
* Việc này giúp mô hình trở nên kháng overfitting hơn bằng cách làm cho mạng nơ-ron không thể phụ thuộc quá nhiều vào một số nút cụ thể và học được các đặc trưng tổng quát hơn từ dữ liệu.



* Data augmentation là một kỹ thuật được sử dụng để tăng cường tập dữ liệu huấn luyện bằng cách tạo ra các biến thể của các mẫu dữ liệu hiện có thông qua các phép biến đổi không ảnh hưởng đến nhãn.
* Kỹ thuật này giúp mô hình học được các đặc trưng chung và tổng quát hơn từ dữ liệu, giảm nguy cơ overfitting và cải thiện khả năng tổng quát hóa của mô hình trên dữ liệu mới.



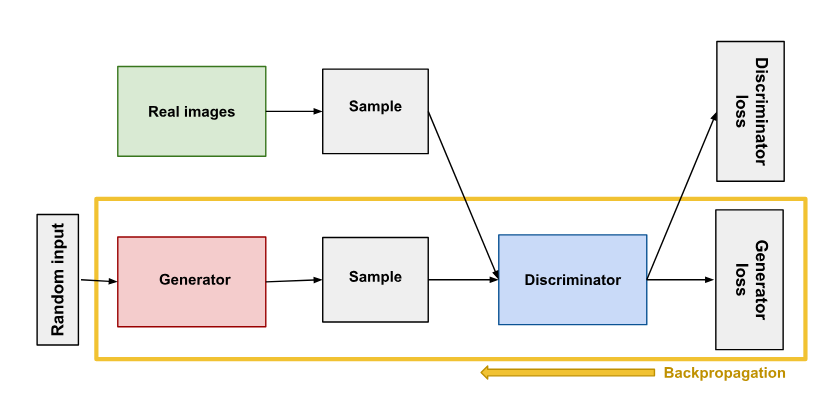
* Batch Normalization là một kỹ thuật chuẩn hóa đầu vào của các lớp ẩn trong mạng nơ-ron. Ý tưởng chính của Batch Normalization là chuẩn hóa đầu vào của mỗi lớp ẩn trong mạng nơ-ron sao cho chúng có phân phối trung bình (mean) xấp xỉ 0 và độ lệch chuẩn (standard deviation) xấp xỉ 1.
* **Hyperparam:**

+ Dropout probability: Xác suất loại bỏ ngẫu nhiên một nút trong quá trình huấn luyện. Thường được thiết lập trong khoảng từ 0.1 đến 0.5.

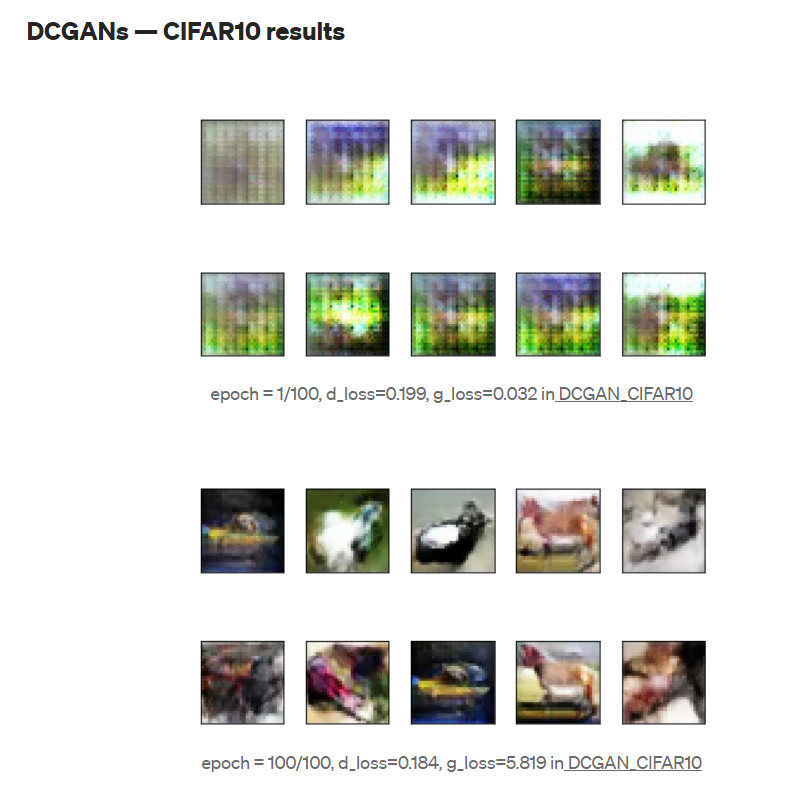
+ Các siêu tham số cụ thể cho từng phép biến đổi, chẳng hạn như kích thước cắt, góc xoay.

+ Số lượng biến thể (augmentations) được tạo ra từ mỗi ảnh gốc trong quá trình huấn luyện.

* + 1. Huấn luyện
    2. Kiểm tra kết quả
* Độ chính xác đạt 88.620%
  1. **GAN(mô hình DCGAN)**
* Kiến trúc mạng GAN bao gồm 2 mô hình: Generator và Discriminator

****

* 1. Đọc dữ liệu
  2. Xử lý dữ liệu đầu vào
* Chuyển các giá trị pixel của ảnh từ khoảng [0:255] thành [-1:1]
  1. Chia tập train và test
  2. Tạo discriminator
  3. Tạo generator
  4. Tạo GAN(kết hợp 2 discriminator và generator)
  5. Huấn luyện
  6. Kiểm tra kết quả

****

* **Hyperpram:**

+ Kích thước không gian latent (Latent Space Dimension): Số lượng chiều của không gian latent, thường được thiết lập trước và ảnh hưởng đến chất lượng và đa dạng của ảnh được tạo ra.

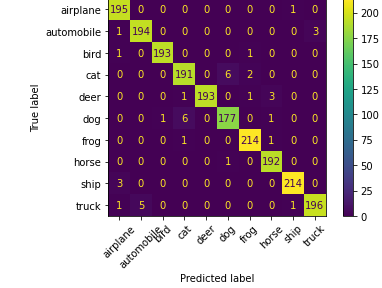
+ Kích thước batch (Batch Size): Số lượng ảnh được sử dụng trong mỗi lần cập nhật trọng số của mạng.

+ Số lớp Convolutional và Transposed Convolutional: Số lượng và kích thước của các lớp convolutional và transposed convolutional trong mô hình, cũng như số lượng các bộ lọc và kích thước của chúng.

**Kết quả**

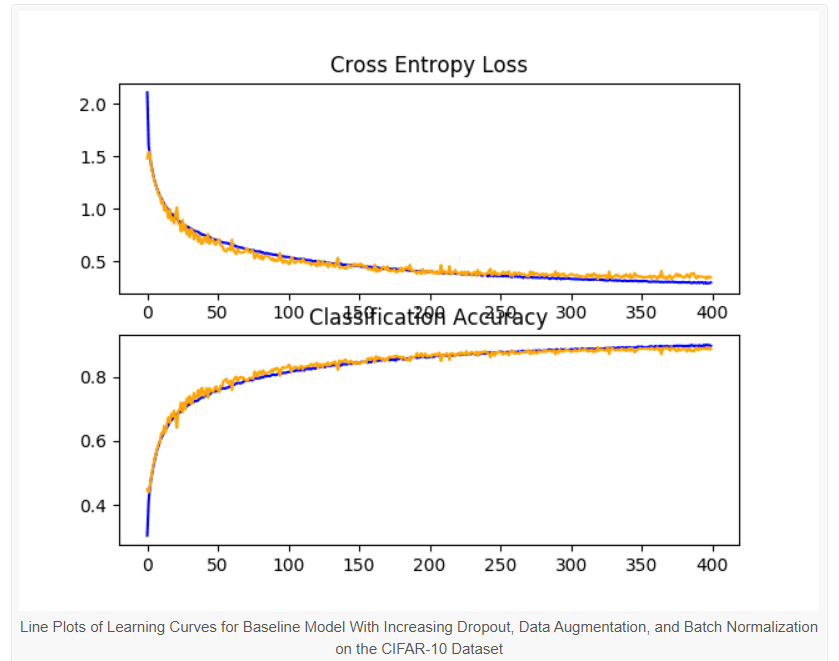
1. **ViT-B/16**

* Độ chính xác 97,95%
* Confusion metric

****

1. **VGG**

* Độ chính xác 88.62%



* Đường màu xanh thể hiển cho tập valid
* Đường màu vàng thể hiển cho tập train
* Biểu đồ này thể hiển trong lúc huấn luyện(400 epoch)
* Biểu đồ “Cross entropy loss” cho thấy 2 đường này sát nhau, thể hiện mô hình không bị overfitting và ngược lại
* Biểu đồ “Classification Accuracy” thể hiện độ chính xác của train và valid qua từng epoch



1. **DCGAN**

* Chỉ số chỉ thể hiện độ chính xác ở mức độ khách quan.
* Muốn xem được độ hiệu quả của mô hình chỉ có thể đánh giá thông qua từng epoch một và in ra ảnh để xem mức độ hoàn thiện của ảnh.

**Kết luận**

1. **ViT-B/16**

* **Hiệu Suất Trên Tập Huấn Luyện**: Mô hình có thể đã đạt được độ chính xác cao trên tập huấn luyện, nhưng cần phải kiểm tra xem có hiện tượng overfitting hay không.
* **Hiệu Suất Trên Tập Kiểm Tra**: Đánh giá mô hình trên tập kiểm tra CIFAR-10 để đánh giá khả năng tổng quát hóa của nó. Độ chính xác trên tập kiểm tra cung cấp thông tin quan trọng về hiệu suất thực tế của mô hình trên dữ liệu mới.
* **Thời Gian Huấn Luyện**: Kiểm tra thời gian cần thiết để huấn luyện mô hình, bao gồm cả thời gian tiền xử lý dữ liệu và huấn luyện thực sự. Thời gian huấn luyện càng ngắn thì mô hình càng có khả năng triển khai nhanh chóng.

1. **VGG**

* **Hiệu Suất Đáng Kể**: Mô hình VGG có hiệu suất tốt trên CIFAR-10 vì nó có khả năng học được các đặc trưng phức tạp từ dữ liệu hình ảnh.
* **Đơn Giản và Dễ Triển Khai:** Kiến trúc của VGG tương đối đơn giản, dễ dàng triển khai và hiểu, giúp cho việc huấn luyện và sử dụng mô hình trở nên thuận tiện.
* **Tiềm Năng Tối Ưu Hóa**: Mặc dù VGG có thể có kích thước lớn với số lượng tham số đáng kể, nhưng có thể tối ưu hóa để giảm độ phức tạp và tăng tốc độ huấn luyện.
* **Thích Hợp Cho Dữ Liệu Nhỏ**: CIFAR-10 là một tập dữ liệu nhỏ với các hình ảnh có độ phân giải thấp, VGG có thể thích hợp với tập dữ liệu này mà không gặp vấn đề về overfitting.
* **Khả Năng Tái Sử Dụng**: Mô hình VGG đã được huấn luyện trước hoặc mô hình có thể tái sử dụng từ các nguồn mở có thể được sử dụng và tinh chỉnh cho các tác vụ phân loại hình ảnh trên CIFAR-10.

1. **DCGAN**

* **Tạo Ảnh Phân Loại Tốt**: DCGAN sinh ra ảnh có chất lượng tương đương hoặc gần tương đương với ảnh thật từ tập dữ liệu CIFAR-10, giúp tạo ra ảnh phân loại chất lượng cao.
* **Tạo Ảnh Đa Dạng**: DCGAN sinh ra các biến thể của ảnh từ tập dữ liệu CIFAR-10, giúp tạo ra ảnh mới và đa dạng cho các ứng dụng sáng tạo.
* **Cần Số Lượng Dữ Liệu Lớn**: DCGAN thường yêu cầu một lượng dữ liệu đáng kể để huấn luyện hiệu quả, đặc biệt là trên tập dữ liệu như CIFAR-10 với nhiều lớp và phân loại.
* **Kiểm Soát Overfitting:** Việc kiểm soát overfitting trong quá trình huấn luyện là một thách thức đối với DCGAN, đặc biệt là khi sử dụng trên tập dữ liệu nhỏ như CIFAR-10

1. [**https://github.com/NielsRogge/Transformers-Tutorials/blob/master/VisionTransformer/Fine\_tuning\_the\_Vision\_Transformer\_on\_CIFAR\_10\_with\_the\_%F0%9F%A4%97\_Trainer.ipynb**](https://github.com/NielsRogge/Transformers-Tutorials/blob/master/VisionTransformer/Fine_tuning_the_Vision_Transformer_on_CIFAR_10_with_the_%F0%9F%A4%97_Trainer.ipynb)
2. [**https://machinelearningmastery.com/how-to-develop-a-cnn-from-scratch-for-cifar-10-photo-classification/**](https://machinelearningmastery.com/how-to-develop-a-cnn-from-scratch-for-cifar-10-photo-classification/)
3. [**https://mafda.medium.com/gans-deep-convolutional-gans-with-cifar10-part-8-be881a77e55b**](https://mafda.medium.com/gans-deep-convolutional-gans-with-cifar10-part-8-be881a77e55b)
4. [**https://towardsdatascience.com/coding-neural-network-dropout-3095632d25ce**](https://towardsdatascience.com/coding-neural-network-dropout-3095632d25ce)
5. [**https://ubiai.tools/what-are-the-difficulties-associated-with-data-augmentation**](https://ubiai.tools/what-are-the-difficulties-associated-with-data-augmentation)
6. [**https://paperswithcode.com/method/randomresizedcrop**](https://paperswithcode.com/method/randomresizedcrop)