Redes Neurais Convolucionais

Arthur Abrahão Santos Barbosa
Universidade Federal de Pernambuco
Centro de Informática
Pernambuco, Brasil
aasb2@cin.ufpe.br

Filipe Samuel da Silva
Universidade Federal de Pernambuco
Centro de Informática
Pernambuco, Brasil
fss8@cin.ufpe.br

I. Objetivos

A. Objetivo Geral

Desnvolver um Classificador Multiclasse que reconheça as imagens do dataset CIFAR100 [1].

B. Objetivos Específicos

- Compreender a implementação de uma Rede Neural Convolucional
- Demonstrar a Importância do Aprendizado Profundo e suas aplicações
- Demonstrar a eficiência de três arquiteturas importantes para a história do Deep Learning

II. Justificativa

Este projeto foi escolhido com base no fato deste dataset ser bastante usado para testar redes neurais com imagens coloridas, e pelo fato de ter uma divisão bastante equilibrada dos dados. [1].

Sua função é verificar a qual das classes pertence uma imagem de tamanho 32x32.

III. Base de Dados

O dataset usado durante o projeto é o CIFAR-100. Ele tem 100 classes contendo 600 imagens cada, totalizando 60.000 imagens. Das 600 imagens que cada classe possui 100 delas são separadas para teste e 500 delas para treino. As 100 classes do dataset estão agrupadas em 20 super classes do seguinte modo [1]:

- **aquatic mammals:** beaver, dolphin, otter, seal, whale
- **fish:** aquarium fish, flatfish, ray, shark, trout
- **flowers:** orchids, poppies, roses, sunflowers, tulips
- **food containers:** bottles, bowls, cans, cups, plates
- **fruit and vegetables:** apples, mushrooms, oranges, pears, sweet peppers
- **household electrical devices:** clock, computer keyboard, lamp, telephone, television
- **household furniture:** bed, chair, couch, table, wardrobe
- **insects:** bee, beetle, butterfly, caterpillar, cockroach
- **large carnivores:** bear, leopard, lion, tiger, wolf
- **large man-made outdoor things:** bridge, castle, house, road, skyscraper
- **large natural outdoor scenes:** cloud, forest, mountain, plain, sea
- **large omnivores and herbivores:** camel, cattle, chimpanzee, elephant, kangaroo
- **medium-sized mammals:** fox, porcupine, possum, raccoon, skunk
- **non-insect invertebrates:** crab, lobster, snail, spider, worm
- **people:** baby, boy, girl, man, woman
- **reptiles:** crocodile, dinosaur, lizard, snake, turtle
- **small mammals:** hamster, mouse, rabbit, shrew, squirrel
- **trees:** maple, oak, palm, pine, willow
- **vehicles 1:** bicycle, bus, motorcycle, pickup truck, train
- **vehicles 2:** lawn-mower, rocket, streetcar, tank, tractor

[1]

IV. Análise Exploratória dos Dados

A. Quantidade de Imagens Por Classes

Tanto o dataset de teste quanto o de treino são bastante equilibrados, possuindo a mesma quantidade de imagens para cada classe:

B. Descrição Estatística dos dados

Porque os dados trabalhados são imagens em vez de tabelas, é mais difícil descrevê-las estatisticamente. Portanto as descrevemos do seguinte modo:

1) Média das Imagens por Classe: Um método possível de analisar as imagens estatisticamente é cálcular o valor médio por pixel, e deste modo conseguir uma imagem média que representaria a classe.

- Média da Imagens Não Normalizadas

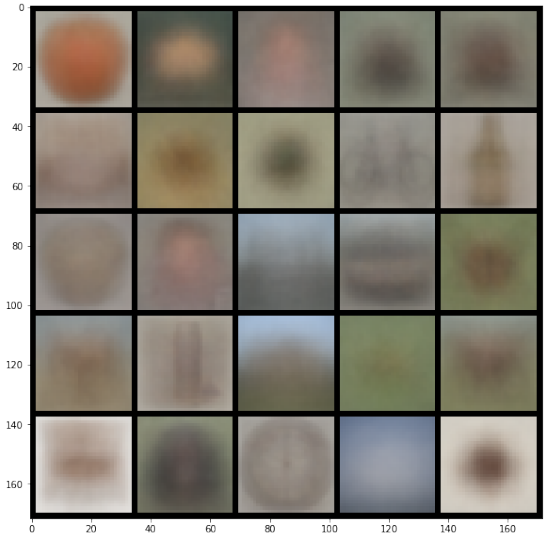

Fig. 1: Da esquerda para a direita, de cima para a baixo: apple, aquarium_fish, baby, bear, beaver, bed, bee, beetle, bicycle, bottle, bowl, boy, bridge, bus, butterfly, camel, can, castle, caterpillar, cattle, chair, chimpanzee, clock, cloud, cockroach

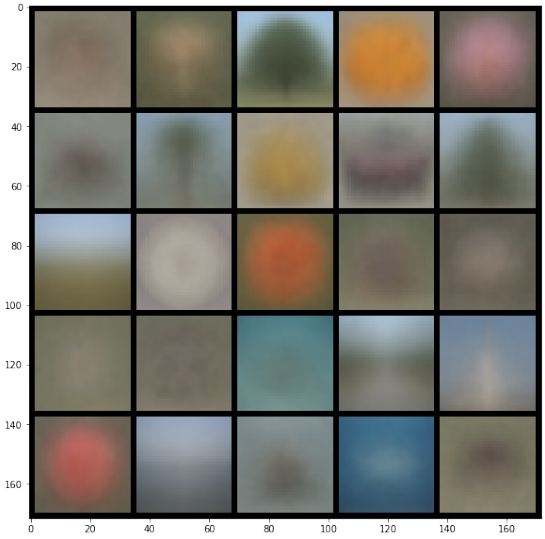

Fig. 3: Da esquerda para a direita, de cima para a baixo: mouse, mushroom, oak_tree, orange, orchid, otter, palm_tree, pear, pickup_truck, pine_tree, plain, plate, poppy, porcupine, possum, rabbit, raccoon, ray, road, rocket, rose, sea, seal, shark, shrew

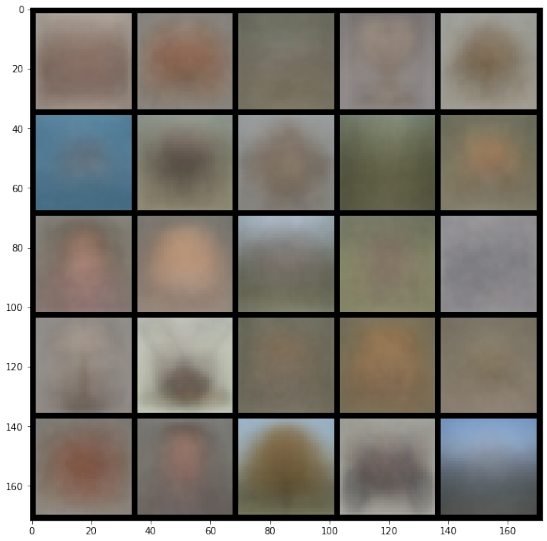

Fig. 2: Da esquerda para a direita, de cima para a baixo: couch, crab, crocodile, cup, dinosaur, dolphin, elephant, flatfish, forest, fox, girl, hamster, house, kangaroo, keyboard, lamp, lawn_mower, leopard, lion, lizard, lobster, man, maple_tree, motorcycle, mountain

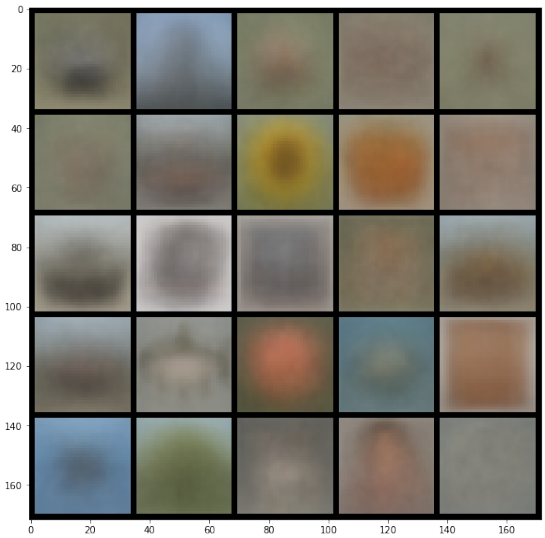

Fig. 4: Da esquerda para a direita, de cima para a baixo: skunk, skyscraper, snail, snake, spider, squirrel, streetcar, sunflower, sweet_pepper, table, tank, telephone, television, tiger, tractor, train, trout, tulip, turtle, wardrobe, whale, willow_tree, wolf, woman, worm

- Média da Imagens Normalizadas

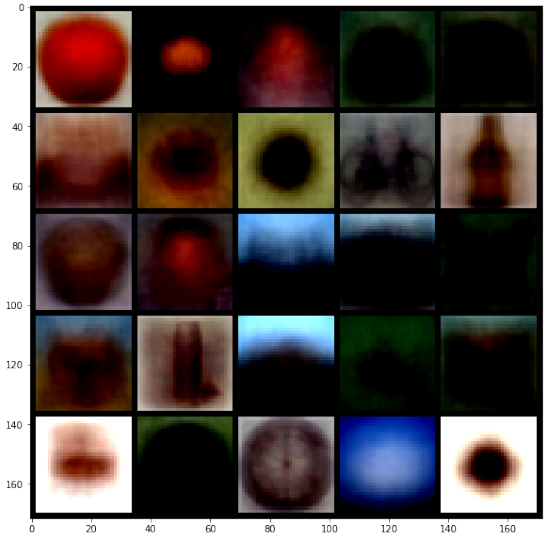

Fig. 5: Da esquerda para a direita, de cima para a baixo: apple, aquarium_fish, baby, bear, beaver, bed, bee, beetle, bicycle, bottle, bowl, boy, bridge, bus, butterfly, camel, can, castle, caterpillar, cattle, chair, chimpanzee, clock, cloud, cockroach

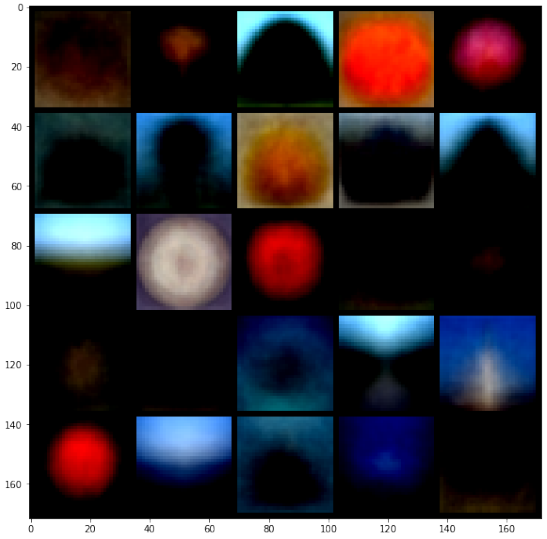

Fig. 7: Da esquerda para a direita, de cima para a baixo: mouse, mushroom, oak_tree, orange, orchid, otter, palm_tree, pear, pickup_truck, pine_tree, plain, plate, poppy, porcupine, possum, rabbit, raccoon, ray, road, rocket, rose, sea, seal, shark, shrew

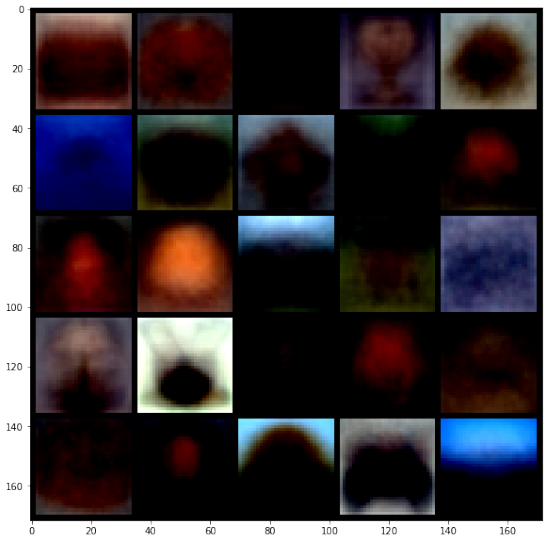

Fig. 6: Da esquerda para a direita, de cima para a baixo: couch, crab, crocodile, cup, dinosaur, dolphin, elephant, flatfish, forest, fox, girl, hamster, house, kangaroo, keyboard, lamp, lawn_mower, leopard, lion, lizard, lobster, man, maple_tree, motorcycle, mountain

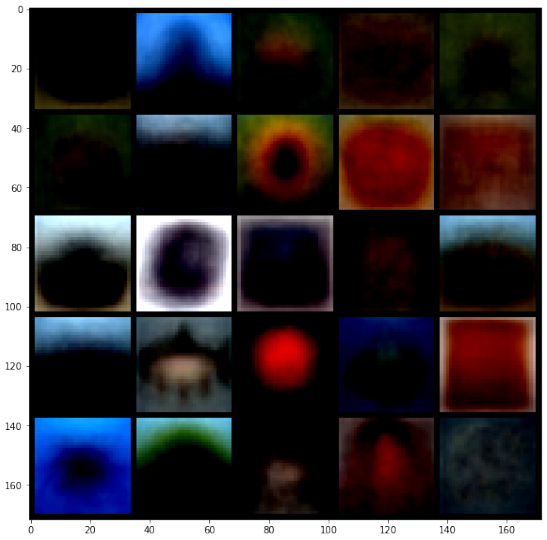

Fig. 8: Da esquerda para a direita, de cima para a baixo: skunk, skyscraper, snail, snake, spider, squirrel, streetcar, sunflower, sweet_pepper, table, tank, telephone, television, tiger, tractor, train, trout, tulip, turtle, wardrobe, whale, willow_tree, wolf, woman, worm

V. Sobre as Métricas Utilizadas

A. *Precision*

Precision é a razão

$$\frac{A_c}{A_c + A_e} \tag{1}$$

onde:

- A_c é o número de amostras corretamente classificadas de uma determinada classe.
- A_e é o número de amostras erroneamente classificadas como sendo desta determinada classe.

Precision é intuitivamente a habilidade do classificador não marcar como pertencente a uma classe uma amostra que não pertence a esta. O melhor valor de Precision é 1 e o pior é zero. [2]

B. Accuracy

Accuracy é a fração de amostras preditas corretamente, e é dada pela seguinte fórmula:

$$\frac{\sum_{1}^{n} A_c}{\sum_{1}^{n} A_t} \tag{2}$$

onde:

- n é o número de classes
- A_c é o número de amostras corretamente classificadas de uma determinada classe.
- A_t é o número de amostras que pertencem a uma determinada classe

[3]

C. Recall-Score

O Recall Score é a razão:

$$\frac{A_c}{A_t} \tag{3}$$

onde:

- A_c é o número de amostras classificadas corretamente de uma determinada classe
- A_t é o número de amostras que pertencem a esta classe

O Recall Score é intuitivamente a habilidade do classificador de encontrar todas as amostras pertencentes a uma classe especifica. O melhor valor do Recall Score é 1 e o pior valor é 0. [4]

D. F1-Score

O F1 Score pode ser interpretado como a média ponderada da precisão e recall. O melhor valor que o F1 score pode alcançar é 1, o pior é 0. A contribuição relativa da precisão e recall para o F1 score são iguais. A fórmula para o F1 score é:

$$F1 = \frac{2 \cdot (precision \cdot recall)}{precision + recall} \tag{4}$$

[5]

E. Confusion Matrix

No caso de classificação multiclasse, uma confusion matrix é dividida em NxN categorias(onde N é o número de classes do problema), cada uma apresentando a quantidade de amostras que se encaixam nesta. A diagonal do meio representa a quantidade de amostras classificadas corretamente e as demais seções da matriz demonstram o número de amostras classificados erroneamente, a quais classes eles pertencem e em quais classes eles foram classificados.

[6]

F. O Nosso Modelo

O modelo que escolhemos não é tão robusto na quantidade de parâmetros, porém é inspirado em soluções modernas de redes convolucionais. Possui quatro camadas convolucionais, além de Batch Normalization e DropOut, que são técnicas para evitar alguns problemas advindos do treinamento com o conjunto de treino e validação com o conjunto de teste. Além da camadas totalmente conectadas, para transformação dos parâmetros de saída das convoluções em valores das classes de saida do problema.

G. LeNet

Um modelo bem simples e que vem dos primórdios da estrutura de rede neural convolucional, proposto por Yann LeCun, em 1989, para lidar com a extração de características das imagens de entrada, aplicando os princípios de redes convolucionais, que são boas para obtenção de características, em dados que têm dependência espacial.

H. AlexNet

Modelo mais moderno, porém já utilizado como base para geração de outros modelos. Possui uma estrutura bem robusta, porém sua grande sacada, é a idéia proposta pelo trio Krizhevsky, Sutskever e Hinton, que utilizou do grande poder das GPUs para realizar o treinamento das redes, com uma rede apropriada para se utilizar desse potencial de processamento.

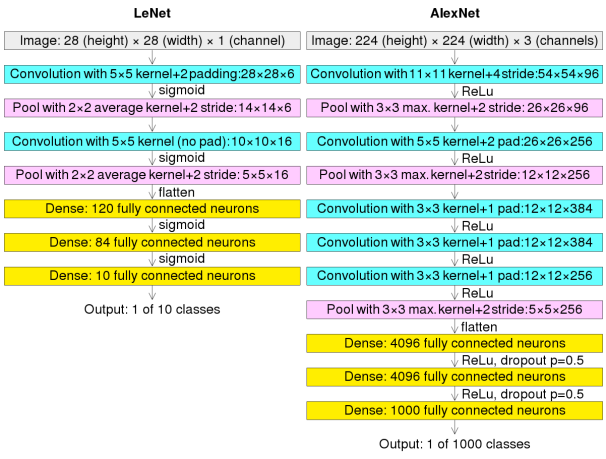

Fig. 9: Diferença entre os modelos LeNet e AlexNet

Os Modelos da LeNet e AlexNet foram modificados de maneira a receber como entrada imagem de resolução 32x32,

sem precisar de um resize. Outra modificação q se mostrou interessante foi a aplicação da Relu ao invés da Sigmoid.

VI. Experimentos

A. CIFAR10

Foram realizados experimentos com o CIFAR10 que é como um subconjunto de classes pertencentes ao CIFAR100, possui 10 categorias de saída, e é util para fazer uma avaliação rápida e com pouco treinamento, garantindo ainda assim, boa performance dos modelos.

B. Similaridade de Pixel

Baseado na abordagem do fastai [7] iremos criar um modelo básico que não utiliza aprendizado de máquina, para ter uma precisão como base para verificar o desempenho dos próximos modelos. Esse método basicamente cálcula uma imagem média para cada classe do conjunto de treino, esta imagem média é basicamente uma imagem formada pela média de cada pixel das imagens de uma classe.

Para fazer a predição esta arquitetura basicamente cálcula a distância de uma imagem a imagem média, e escolhe a classe com a menor distância. Podemos usar o erro quadrático médio ou o valor absoluto das diferenças como valor da distância total entre uma imagem e a média da sua classe.

Porém, ao contrário da abordagem original do fastai para o dataset mnist com apenas duas classes com imagens em preto e branco, que teve uma 'accuracy' razoável com este método, temos que a 'accuracy' que tivemos com o CIFAR 100 que possui muito mais classes, e imagens coloridas, não é melhor do que selecionarmos uma classe ao acaso.

C. Treinando os Modelos

Os dados do dataset que foram separados, agora serão usados para o treinamento e avaliação da performance das redes neurais. Para realizar o treino os dados foram separados em batchs, e todos os três modelos foram treinados durante 50 épocas, para que houvesse um período de aprendizagem razoável de acordo com o tamanho das redes. A cada época de treino e alguns batchs foram selecionados para mostrar a o valor da função loss, e mostrar a evolução do treinamento em diferentes épocas.

Os desempenhos da nossa rede, foram melhores que as duas outras, em 50 épocas de treinamento: O nosso modelo, garantiu a acurácia

- Nosso: de $\frac{5520}{10000}$ = 55.2%
- LeNet: 1%
- AlexNet: de $\frac{4906}{10000}$ = 49.059999999999995%

E um exemplo do desempenho do nosso modelo para cada classe do dataset:

D. Apenas um Epoch

Para o treinamento com apenas uma época, nós restauramos o modelo ao estado sem treinamento, e determinamos o num epochs como 1. Esse tipo de treinamento serve para avaliar a perfomance inicial do modelo ao ser corrigido pelas funções de perda.

	precision	recall	f1-score	support
apple	0.78	0.77	0.77	100
aquarium_fish	0.67	0.71	0.69	100
baby	0.42	0.42	0.42	100
bear	0.46	0.22	0.30	100
beaver	0.32	0.40	0.36	100
bed	0.36	0.55	0.43	100
bee	0.62	0.66	0.64	100
beetle	0.54	0.66	0.59	100
bicycle	0.79	0.76	0.78	100
bottle	0.54	0.74	0.62	100
bowl	0.48	0.32	0.38	100
boy	0.50	0.34	0.40	100
bridge	0.56	0.60	0.58	100
bus	0.51	0.53	0.52	100
butterfly	0.57	0.54	0.55	100
camel	0.34	0.59	0.43	100
can	0.64	0.55	0.59	100
castle	0.73	0.69	0.71	100
caterpillar	0.40	0.49	0.44	100
cattle	0.55	0.41	0.47	100
chair	0.79	0.74	0.76	100
chimpanzee	0.66	0.72	0.69	100
clock	0.66	0.44	0.53	100
cloud	0.76	0.71	0.74	100
cockroach	0.76	0.67	0.71	100
couch	0.46	0.36	0.40	100

Fig. 10: Desempenho 50 Épocas

O nosso modelo, também garantiu melhor desempenho com apenas uma época, com acurácia de 14.85%, enquanto a acurácia da AlexNet foi de 14.85%, já a LeNet apenas 1%.

E. Limitando os Dados

O treinamento com limitação de dados também é uma boa opção para avaliar a capacidade das redes em aprender com poucos exemplos. É possível perceber que os desempenhos de todas as redes acaba caindo, pois existe um princípio, de que, quanto mais exemplos de todas as classes você tiver, mais preciso e mais acurado ele tenderá a ser. Ou seja, ele irá aprender mais sobre aquelas classes. Foi utilizado então, como objetivo de atingir esse experimento, a utilização dos dados de treinamento como dados de teste, e os dados de teste serviram como treinamento. Nesse caso, apesar da perda de desempenho, nosso modelo teve a performance de 30.194%, a AlexNet de 19.054% e a LeNet de 1%

F. Sem Normalização e sem Data Augmentation

Por último, foi realizado o treinamento sem normalização. A normalização é uma estratégia para que o treinamento seja realizado mais rapidamente, e para evitar alguns problemas que podem surgir nas métricas das funções de erro. Os desempenhos sem normalização foram um pouco menores, o que nos mostra que essa estratégia é realmente útil para se aplicar nos treinamentos da rede. A nossa rede obteve, 49.76% de acurácia, a AlexNet 42.1% e a LeNet apenas 0.67%

Esse desempenho da LeNet nos levou a uma modificação.

VII. Análise dos Resultados

A. LeNet com Relu

Nós nos deparamos com um problema, a rede LeNet com a função sigmoid estava tendo resultados muito abaixo, comparado as demais, redes. Abaixo até mesmo considerando a complexidade do tamanho da rede. Portanto, modificamos a rede para uma função ReLu no lugar da sigmoid, e obtivemos bom resultado. Temos que a sua acurácia aumentou para 33.75%. Isso mostra a modernidade da ReLU, como função de ativação.

B. Desempenho da rede com os dados ruidosos

Para avaliar o desempenho das redes, aplicamos um ruido gaussiano nas imagens de teste, e testamos a acurácia da rede, mesmo após a aplicação do filtro que gerou o ruido e tivemos as acurácias de 17.97% do nosso modelo convolucional, 28.02% a LeNet com Relu e 8.18% a AlexNet

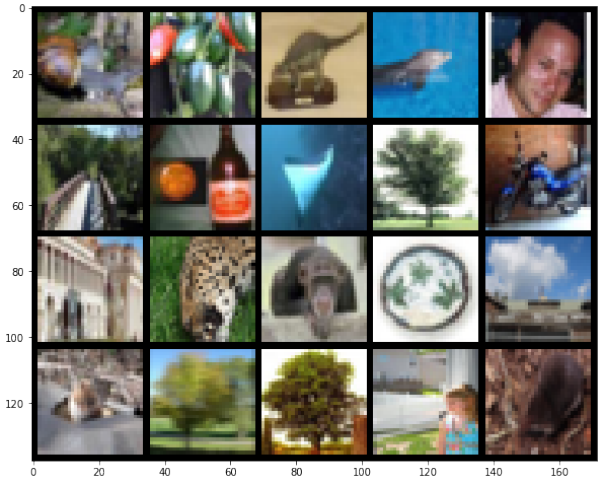

Fig. 11: Imagens Ruidosas

VIII. Conclusões e Discussões

Podemos observar que, para determinados problemas, existem soluções variadas e, geralmente, temos que obter dados específicos do problema para construir uma solução viável. Isso se aplica as redes neurais em geral, e as redes convolucionais, especificamente. Vemos que o tamanho do modelo, não implica em eficácia, e também que as diferentes variações dos modelos, geram novas soluções e melhores. Apesar da grande proposta das redes profundas, lidar com a idéia da máquina extraindo todas as caracteristicas e funcionando com um sistema que apenas inserimos os dados como entrada, vemos que, as redes para resolver diferentes problemas, possuem diferentes tamanhos e formatos, baseados em considerações particulares das condições do problema analisado, ou seja, vemos muitos modelos adaptados aos diferentes tipos e formatos de dados, em contraste com um modelo mais abstrato e generalista. (Com destaque, para algumas propostas modernas de modelos generalistas, que estão atualmente implementados) Apesar disso, a quantidade de aplicações das redes neurais e a facilidade de adaptação das redes para resolver vários tipos de problemas, é o que garante a grande capacidade dessas redes em realmente aprender diversos tipos de questões abstratas, o que significa que elas podem realmente, aprender sobre os dados.

References

[1] A. Krizhevsky. (2009) Cifar-10 and cifar-100 datasets. [Online]. Available: https://www.cs.toronto.edu/~kriz/cifar.html

[2] "sklearn.metrics.precision_score." [Online]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.metrics.precision_score.html

[3] A. Long, "Understanding data science classification metrics in scikit-learn in python," Aug. 2018. [Online]. Available: https://towardsdatascience.com/understanding-data-science-classification-metrics-in-scikit-learn-in-python-3bc3368650

[4] "sklearn.metrics.recall_score." [Online]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.metrics.recall_score.html

[5] sklearn.metrics.f1_score. [Online]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.metrics.f1_score.html

[6] B. Shmueli, "Multi-class metrics made simple, part i: Precision and recall," Jul. 2019. [Online]. Available: https://towardsdatascience.com/multi-class-metrics-made-simple-part-i-precision-and-recall-9250280bddc2

[7] J. Howard and S. Gugger, *Deep Learning for Coders with Fastai and Pytorch: AI Applications Without a PhD*. O'Reilly Media, Incorporated, 2020. [Online]. Available: https://books.google.no/books?id=xd6LxgEACAAJ

[8] A. Krizhevsky, "Learning multiple layers of features from tiny images," University of Toronto, Toronto, Ontario, Tech. Rep. 0, 2009.

[9] Muthu, "Understanding the classification report through sklearn," Jul. 2018. [Online]. Available: https://muthu.co/understanding-the-classification-report-in-sklearn/