

Machine Learning Applied to Image Classification

Zuilho Rodrigues Castro Segundo

August 25, 2021

Abstract

Machine Learning is a field of computer science with severe applications in the modern world. One of the main applications is the use of neural networks in computer vision, recognizing faces in a photo, analyzing x-rays, or identifying an artwork. This paper aims to explore the concepts of machine learning, supervised learning, and neural networks, applying the learned concepts in the CIFAR10 dataset, which is a problem of image classification, trying to build a neural network with high accuracy. To avoid overfitting, we proposed trying to use two different methods of regularization: L2 and dropout. We find that using dropout regularization gives the best accuracy on our model when compared with the L2 regularization

1. Introduction

Machine Learning is the field of computer science that uses data analysis to create a model to predict future examples. It searches patterns in the data analyzed, making decisions with no need for human intervention.

Supervised learning is one of the areas of machine learning where labeled datasets are used to train the models to classify and predict new examples in an accurate way. Using the input data, the model adjusts the parameters in such a way that it can predict new cases¹. An example in a daily application is the predictive text in smartphones.

Supervised learning can be divided into two main types: Classification and Regression. Regression is used to make predictions with numerical values, such as predicting the value of a house based on the number of bedrooms. Classification is used to classify the data into categorical labels, such as defining if some image is of a dog or a cat.

This paper works with a dataset of a multiclassification problem, which has more than two classes as the output. The dataset used in this paper is the CIFAR10 (Alex Krizhevsky, 2009)², its purpose is to classify some small colored images into one of the 10 labels (airplane, automobile, bird, cat, deer, dog, frog, horse, ship, truck).

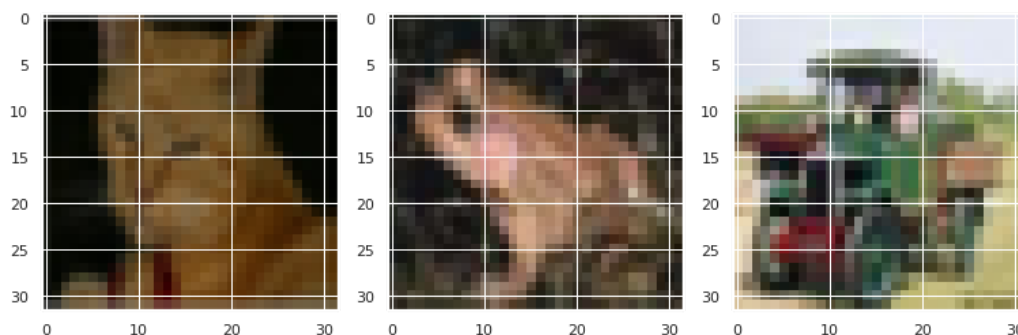


Figure 1 - CIFAR10 images examples (Alex Krizhevsky, 2009)³

¹ "What Is Supervised Learning?," June 30, 2021, <https://www.ibm.com/cloud/learn/supervised-learning>.

² Alex Krizhevsky, "Learning Multiple Layers of Features from Tiny Images," 2009.

³ Krizhevsky.

All the images are the size of 32x32 pixels and are colored, which means there are 3 layers, one for each RGB color. Each of the pixels contains all the 3 layers and is used as one feature of the model, which gives 3072 different features that are numbers between 0 and 255. Based on the features of each image, the computer tries to make a model to classify the images as one of the labels. As our problem is a multiclassification problem with 10 classes, there will be 10 different outputs, that we label as $(\hat{y}_1, \hat{y}_2, \dots, \hat{y}_{10})$, and each of them will be a function of the different features in the input.

$$\hat{y} = \hat{y}(x_1, x_2, \dots, x_{3071})$$

2. Neural networks

There are several types of machine learning methods, each one with a different application. A neural network is one of the types. Inspired by the human brain, neural networks have a set of L layers, which can be divided into 3 types: input (noted as L_0), hidden (noted as L_m), and output (noted as L). In each layer, there is a determined number of nodes (that can be compared to the neurons in the human brain) that are connected to all the nodes in the next layer⁴.

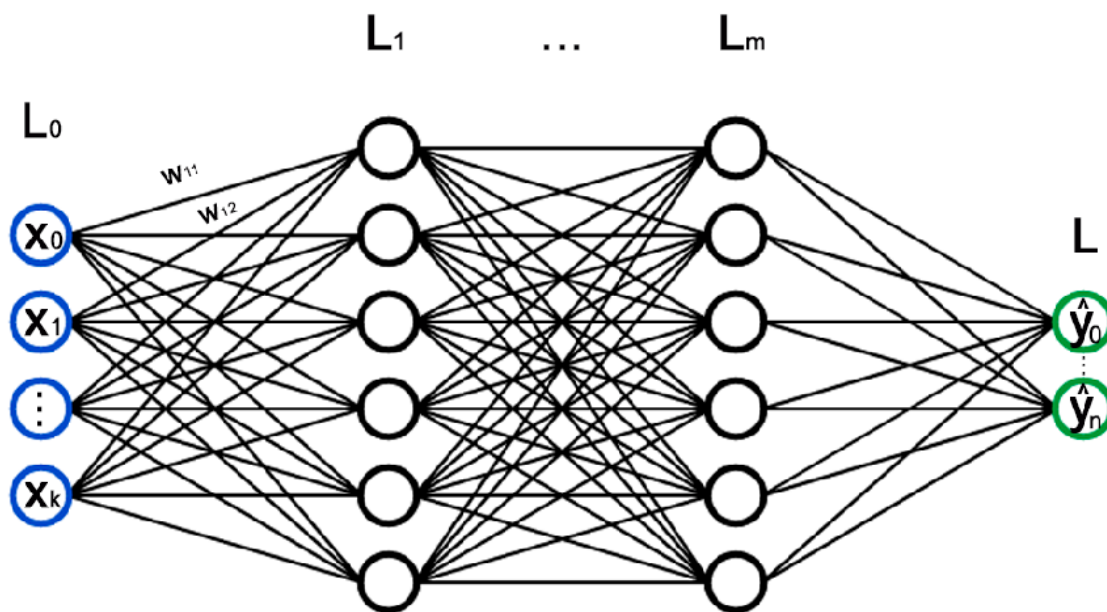


Figure 2 - Example of a neural network.

To make the model, it is necessary to train the network. Using a labeled dataset as input, the network tries to predict the output. To do this, each feature on the input layer is passed to the next layer using the connections that are associated with a weight (w_i), that determines the importance of each feature to discover the intended label. Also, each layer has a bias value (b), to correct any noise in the data. This way, each node of the layer L_m can be determined as:

⁴ "What Are Neural Networks? | IBM," accessed November 16, 2021, <https://www.ibm.com/cloud/learn/neural-networks>.

$$\sum_{i=1}^k w_i x_i + b_{L_m}$$

Where w_i is the parameter associated with the edge that connects the node in the L_{m-1} layer to the L_m layer, x_i is the data associated with the previous node, and b_{L_m} is the bias in the L_m layer.

At the beginning of the training, random values to each weight and bias are selected, and as the training goes on, the values of each ‘w’ and ‘b’ are changed to make the model as accurate as possible. That is done using a cost function that makes the model converge to a minimum of error. In multiclassification problems (problems with more than two classes), the way to measure the error is called a categorical cross-entropy error function. At the end of the training, the neural network selects the parameters (w’s and b’s) that make the categorical cross-entropy error as small as possible.

$$J = - \sum_{i=1}^m y_i \log(\hat{y}_i)$$

Where J is the error, y is the label in the training dataset, and \hat{y} is the prediction of the neural network

This kind of neural network is referred to as “fully connected”. However, considering the usage in an image recognition problem such as CIFAR10, it would take a lot of time to train the network, as the number of features in an image is equal to the number of pixels it has, and to connect all the nodes would take a lot of parameters (w’s and b’s) making it very hard to execute. To solve a computational vision problem there is what is called a Convolutional Neural Network (CNN), which works very well with images. The basic process of a CNN is: Convolution -> Pooling -> Fully connected.⁵

Convolution is a mathematical way to reduce the number of features in an image without losing important information in the process. An image is nothing more than a matrix, where each value in the columns and rows represents the intensity of each pixel. Using a filter, called kernel (that is also a matrix), the image is scanned and the output is a matrix resulting from the convolutional operation of each element (Figure 3). This process helps to find the edges in the images, which makes it easier to recognize the elements.⁶

⁵ Angel Das, “Convolution Neural Network for Image Processing – Using Keras,” Medium, January 11, 2021, <https://towardsdatascience.com/convolution-neural-network-for-image-processing-using-keras-dc3429056306>.

⁶ Sumit Saha, “A Comprehensive Guide to Convolutional Neural Networks – the ELI5 Way,” Medium, December 17, 2018, <https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53>.

$$\begin{vmatrix} a & b & c & d \\ e & f & g & h \\ i & j & k & l \\ m & n & o & p \end{vmatrix} * \begin{vmatrix} 0 & 1 \\ 1 & 0 \end{vmatrix} = \begin{vmatrix} b+e & d+g \\ j+m & l+o \end{vmatrix}$$

Figure 3 - Example of a general convolution

After the convolutional process, the data is now passed by a max-pooling layer. As the convolutional layer, the objective of the max-pooling layer is to reduce the number of features without losing important information. Using a filter, that is a matrix, the max-pooling layer scans the output of the convolutional layer, and returns the highest number in each section of the data (Figure 4). This process is important to achieve the dominant parts of the data, extracting the most important features.

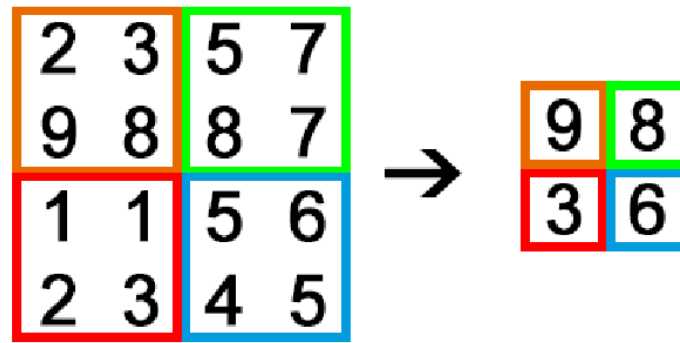


Figure 4 - Example of 2x2 max-pooling

Now, with a reduced number of features, the output data of the pooling layer is used as the input of a fully connected layer, that is trained to classify the images into the labels. In the figure 5 is the complete scheme of a CNN.

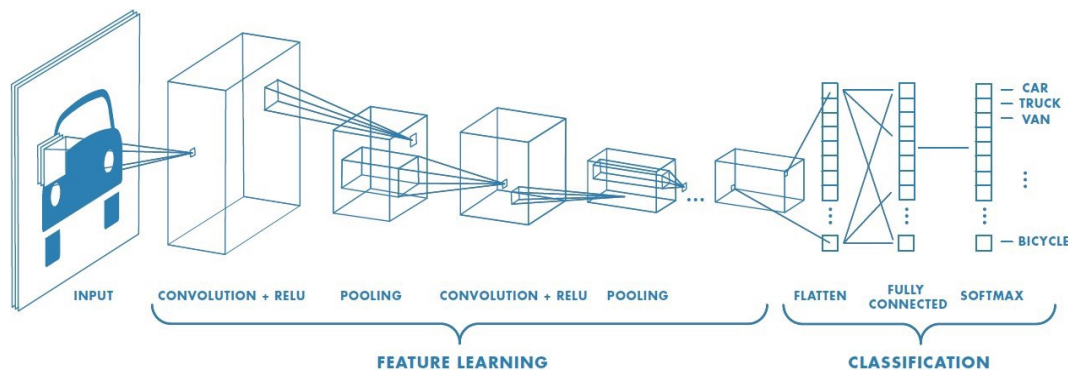


Figure 5 - Complete CNN⁷

⁷ Saha.

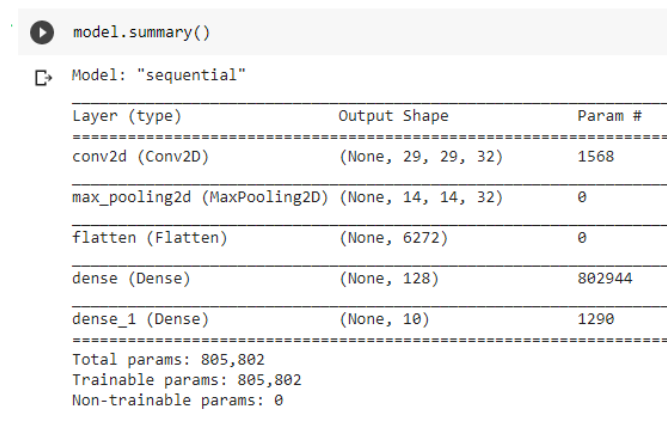
3. Applications in the CIFAR10 dataset

3.1. Base project

As said in section 1, CIFAR10 is an image classification problem and that is why the chosen way to approach it was using a Convolutional Neural Network.

Before starting to build the model it was necessary to set some things. First, it was necessary to split our dataset in two, the training and the validation sets. The purpose of this will be better explained in the next section (3.2 Base Model Evaluation), but ultimately it is important to have a way to evaluate how well the model is doing. Second, it was important to scale the features and make sure the range of the numbers wasn't too large to ensure the weights and bias would not change drastically, making the training process more difficult.

Our initial model used a simple CNN (Figure 6) to see the results and how well it worked. The model was constructed using Google Colab and Keras libraries⁸.



```
model.summary()
```

Model: "sequential"

| Layer (type) | Output Shape | Param # |
|------------------------------|--------------------|---------|
| conv2d (Conv2D) | (None, 29, 29, 32) | 1568 |
| max_pooling2d (MaxPooling2D) | (None, 14, 14, 32) | 0 |
| flatten (Flatten) | (None, 6272) | 0 |
| dense (Dense) | (None, 128) | 802944 |
| dense_1 (Dense) | (None, 10) | 1290 |

Total params: 805,802
Trainable params: 805,802
Non-trainable params: 0

Figure 6 - Model Summary

3.2. Model Evaluation

An important part of the project is to see how well the model works. This process starts before the construction of the network and splits the dataset in two. The training set will be used to train the model, which means the model will adjust the parameters based on the training set. The second part of our data, the validation set, will be used to measure the efficiency of the model.

In the CIFAR10 problem, as the objective is to correctly classify the images, the measure used was the accuracy, a number between 0 and 1, and is a ratio between the number of correct predictions by the number of predictions done. Simply, the higher the accuracy the better the model is working.

⁸ "Google Colaboratory," accessed November 17, 2021, <https://colab.research.google.com/>; Das, "Convolution Neural Network for Image Processing — Using Keras"; "Keras: The Python Deep Learning API," accessed November 17, 2021, <https://keras.io/>.

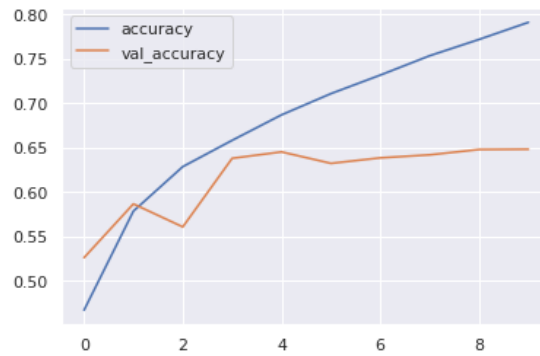


Figure 7 - Accuracy in the base model

In the base model, as can be seen in Image 7, the accuracy in the validation set was 0.6481, which is a good number. However, when looking at the training set, it is possible to see that the accuracy was way higher than in the validation set, a number of 0.7911. This means that the model is overfitting.

Overfitting happens when the model works very well in the training set but is not able to generalize the information and doesn't work in the new data. Some of the reasons for overfitting are a high number of parameters or long training, which makes the model learn irrelevant information from the training set that does not necessarily appear in the new data⁹.

To overcome overfitting, there are some strategies, like lambda regularization and dropout regularization¹⁰.

3.3. Second model - Lambda Regularization (L2)

The first method we tried using to solve the overfitting problem was to use Lambda Regularization.

The lambda regularization introduces a new term to the cost function that helps prevent overfitting¹¹.

$$J_{reg} = J + \lambda \sum_{i=1}^m w_i^2$$

Here, λ is a parameter just as the w 's and b 's. If λ is 0, the cost function will become the categorical cross-entropy, as normal. However, if λ is bigger than 0, the w 's will have to shrink to make the error in the cost function smaller. This way the model reduces its complexity, and as a consequence reduces overfitting.

⁹ "What Is Overfitting? | IBM," accessed November 16, 2021, <https://www.ibm.com/cloud/learn/overfitting>.

¹⁰ "Understanding Regularization in Machine Learning | by Ashu Prasad | Towards Data Science," accessed November 16, 2021, <https://towardsdatascience.com/understanding-regularization-in-machine-learning-d7dd0729dde5>.

¹¹ "L1 and L2 Regularization Methods. Machine Learning | by Anuja Nagpal | Towards Data Science," accessed November 16, 2021, <https://towardsdatascience.com/l1-and-l2-regularization-methods-ce25e7fc831c>.

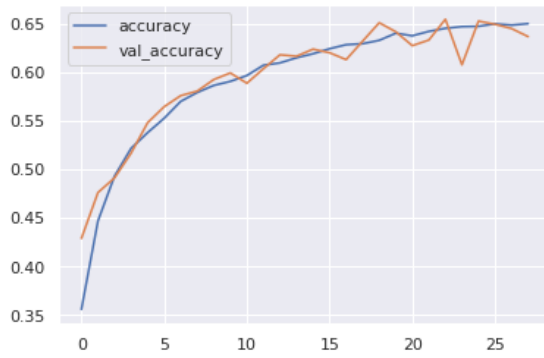


Figure 8 - $\lambda = 0.01$

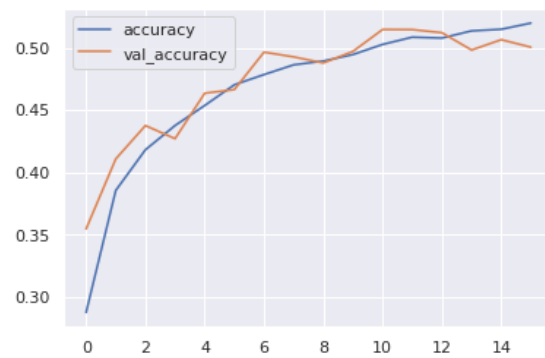


Figure 9 - $\lambda = 0.02$

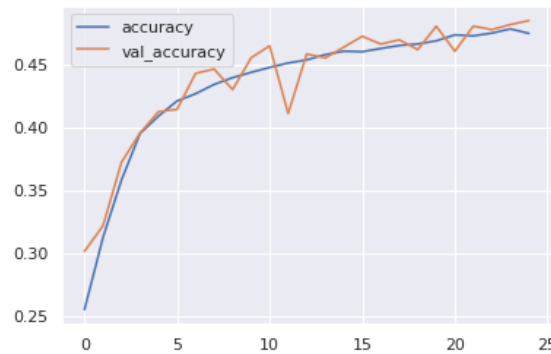


Figure 10 - $\lambda = 0.03$

Analyzing the graphs, it is easy to see that the overfitting was reduced. However, as lambda increased, the accuracy in both the validation and training set reduced drastically. Because of that, there was a need to try another way to prevent overfitting.

3.4. Third model - Dropout regularization

As the application of the lambda regularization reduced the efficiency of the model, we decided to try another approach, using dropout regularization.¹²

Dropout is a method of regularization that works when training the model. During the training phase, some of the nodes in each layer are turned off, giving the layers another configuration during each run (Figure 11). As some of the nodes are “dropped out”, the other nodes on the layer have to compensate for this, giving new weights and importance to different features that help to generalize the model¹³.

¹² Jason Brownlee, “A Gentle Introduction to Dropout for Regularizing Deep Neural Networks,” *Machine Learning Mastery* (blog), December 2, 2018, <https://machinelearningmastery.com/dropout-for-regularizing-deep-neural-networks/>.

¹³ Nitish Srivastava et al., “Dropout: A Simple Way to Prevent Neural Networks from Overfitting,” *Journal of Machine Learning Research* 15, no. 56 (2014): 1929–58.

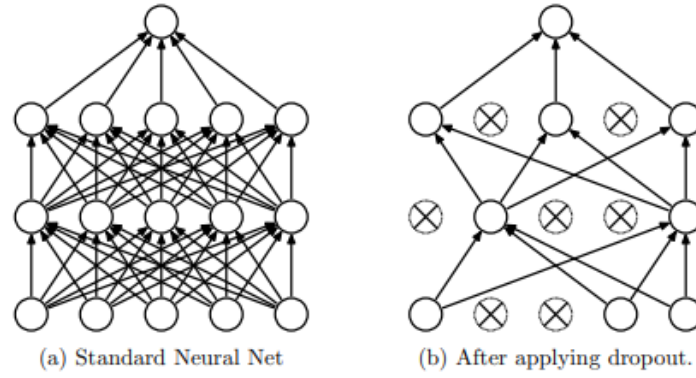


Figure 11 -Dropout network (Srivastava et al., 2014)¹⁴

Using this method, each node in the network is associated with a probability r_i of being “dropped out”, that is the dropout rate (Figure 12).

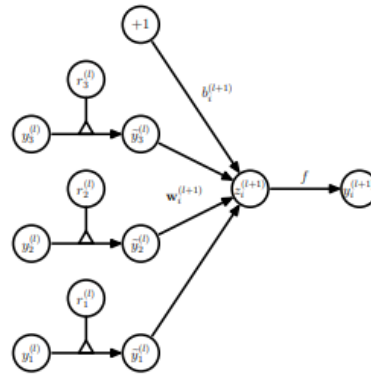


Figure 12 - Dropout in each node (Srivastava et al., 2014)¹⁵

In our CIFAR10 problem, we applied a dropout rate of 0.2¹⁶. Our results can be found in Figures 13 and 14.

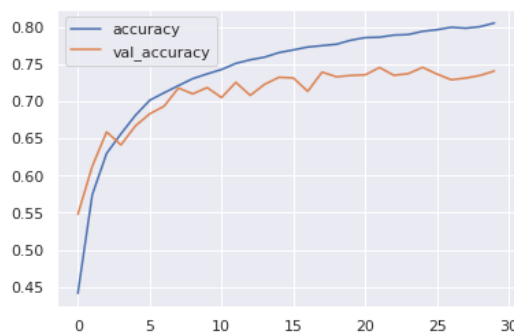


Figure 13 - Dropout accuracy

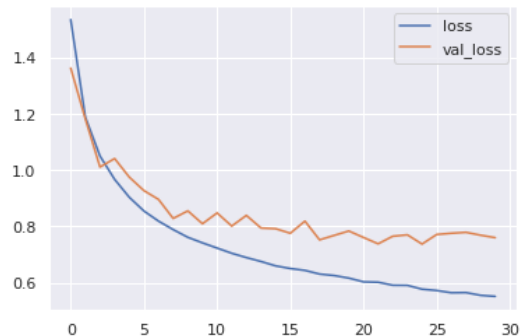


Figure 14 - Dropout Error

¹⁴ Srivastava et al.

¹⁵ Srivastava et al.

¹⁶ Jason Brownlee, “Dropout Regularization in Deep Learning Models With Keras,” *Machine Learning Mastery* (blog), June 19, 2016, <https://machinelearningmastery.com/dropout-regularization-deep-learning-models-keras/>.

As can be seen in the graph (Figure 13, Figure 14), the accuracy and the error are almost the same in the training and validation sets, which shows the overfitting was reduced. Also, it has high accuracy in the validation set, of 0.7406.

4. Conclusion

Machine learning is a field with many applications. This paper worked with a dataset of small images, applying the base concepts in an image classification problem. As can be seen, there are a lot of concepts that can be extracted from this problem and could be applied in more difficult ones, like face recognition or exotic plants recognition. Also, it's important to see the problems that can appear during the development of a neural network, such as overfitting, and that there are many ways to approach it.

Using different methods to correct the overfitting on the CIFAR10 dataset, we showed that using the lambda regularization wasn't the most effective way of doing so. Even while making the model less complex and almost equaling the accuracy in the validation and training sets, the lambda regularization removed the ability to generalize and predict new data. As for using the dropout regularization, even with some difference between the validation and training sets accuracy, it was able to generalize and predict a great part of the validation set. So the method chosen to avoid overfitting was the dropout, as it increased the accuracy.

5. Code

https://colab.research.google.com/drive/1irDxpOGF7grEvLYbjI_xH7LzlTkOg2Hb?usp=sharing

6. References

- Brownlee, Jason. "A Gentle Introduction to Dropout for Regularizing Deep Neural Networks." *Machine Learning Mastery* (blog), December 2, 2018. <https://machinelearningmastery.com/dropout-for-regularizing-deep-neural-networks/>.
- . "Dropout Regularization in Deep Learning Models With Keras." *Machine Learning Mastery* (blog), June 19, 2016. <https://machinelearningmastery.com/dropout-regularization-deep-learning-models-keras/>.
- Das, Angel. "Convolution Neural Network for Image Processing — Using Keras." Medium, January 11, 2021. <https://towardsdatascience.com/convolution-neural-network-for-image-processing-using-keras-dc3429056306>.
- "Google Colaboratory." Accessed November 17, 2021. <https://colab.research.google.com/>.
- "Keras: The Python Deep Learning API." Accessed November 17, 2021. <https://keras.io/>.
- Krizhevsky, Alex. "Learning Multiple Layers of Features from Tiny Images," 2009.
- "L1 and L2 Regularization Methods. Machine Learning | by Anuja Nagpal | Towards Data Science." Accessed November 16, 2021. <https://towardsdatascience.com/l1-and-l2-regularization-methods-ce25e7fc831c>.
- Saha, Sumit. "A Comprehensive Guide to Convolutional Neural Networks — the ELI5 Way." Medium, December 17, 2018. <https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53>.
- Srivastava, Nitish, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. "Dropout: A Simple Way to Prevent Neural Networks from Overfitting." *Journal of Machine Learning Research* 15, no. 56 (2014): 1929–58.

- “Understanding Regularization in Machine Learning | by Ashu Prasad | Towards Data Science.” Accessed November 16, 2021.
<https://towardsdatascience.com/understanding-regularization-in-machine-learning-d7dd0729dde5>.
- “What Are Neural Networks? | IBM.” Accessed November 16, 2021.
<https://www.ibm.com/cloud/learn/neural-networks>.
- “What Is Overfitting? | IBM.” Accessed November 16, 2021.
<https://www.ibm.com/cloud/learn/overfitting>.
- “What Is Supervised Learning?,” June 30, 2021.
<https://www.ibm.com/cloud/learn/supervised-learning>.