University of Ibadan
Journal of Science and Logics in ICT Research

A Review of Fixed Input Size Limitation in Convolutional Neural Networks Models and Proposed Solutions

[1]✉Famurewa K.F., [2]Sakpere W. and [3]Adelodun F.O.

1,2,3Department of Computer Science, Lead City University, Ibadan, Nigeria*
ogedengbekofo@gmail.com, sakpere.wilson@lcu.edu.ng, adelodunfelicia@gmail.com

Abstract
Convolutional Neural Networks (CNNs) are incredibly powerful deep learning techniques that have been applied to computer vision applications to yield innovative results. CNNs are ideal for applications like object identification, image segmentation, and image classification because they can automatically extract pertinent information from the images without human supervision. While CNNs can attain state-of-the-art performance in many applications and domains, most CNNs currently have limitations in training and prediction due to their sensitivity to image size. As a result, image recognition datasets are typically downsized to the input size specification of the CNN models. This study's objective is to examine CNN models and suggest possible solutions to tackle the fixed input size problem that exists in CNN models.

Keywords: *Artificial Intelligence, Machine Learning, Deep Learning, Convolutional Neural Network, Fixed Input Size Limitation.*

1. Introduction

In the context of data analysis and computation, artificial intelligence and machine learning have advanced significantly in recent years, allowing applications to generally operate intelligently [1]. Within the field of artificial intelligence, machine learning allows computers to learn from data without being explicitly programmed. Machine learning depends on input, like training data or domain knowledge, to understand objects, domains, and the connections between them much like the human brain does. One benefit of applying machine learning approaches is that they may be used to construct more reliable and accurate models that can evaluate complex information and offer insights that can help with decision-making [2, 3].

Researchers in the fields of data science and machine learning employ a variety of popular datasets for a variety of reasons. These include, for instance, datasets related to cyber security, smartphones, Internet of Things, agriculture, e-commerce, health, and many other application domains. These data may vary depending on the real-world application and may be structured, unstructured, semi-structured, or metadata. Various machine learning algorithms can be employed based on their learning capabilities to analyze such data in a specific problem area and extract insights or usable knowledge for developing intelligent applications for the actual world. Supervised learning, unsupervised learning, semi-supervised learning, and reinforcement learning are the four basic categories into which machine learning algorithms fall [4].

In order to anticipate future events, supervised learning uses labeled examples to apply prior knowledge to new data. Developing a model that can precisely predict the output for novel, unknown inputs is the aim of supervised learning. The two most popular supervised tasks are "regression," which fits the data, and "classification," which divides the data. Naive Bayes (NB), Linear Discriminant Analysis (LDA), Logistic Regression (LR), K-Nearest Neighbors (KNN), Support Vector Machine (SVM), Decision Tree (DT), Random Forest (RF), Adaptive Boosting (AdaBoost), Extreme gradient boosting (XGBoost), Stochastic gradient descent (SGD), Simple and multiple linear regressions, Polynomial regression, LASSO, and Ridge regression are the most widely used supervised learning algorithms utilized in a wide range of applications. Applications including image and speech

Famurewa K. F., Sakpere W. and Adelodun F. O. (2025) A Review of Fixed Input Size Limitation in Convolutional Neural Networks Models and Proposed Solutions. *University of Ibadan Journal of Science and Logics in ICT Research* (UIJSLICTR), Vol. 13 No. 1, pp. 144 - 152

recognition, natural language processing, financial forecasting or prediction, cost estimation, trend analysis, marketing, time series estimation, drug response modeling, and many more frequently employ supervised learning algorithms.

When a computer is trained on unlabeled data, it must discover underlying patterns or structures on its own without human assistance. This process is known as unsupervised learning. Without any prior ideas on what the final product should be, unsupervised learning aims to extract new knowledge and insights from the data. Clustering, density estimation, feature learning, dimensionality reduction, association rule discovery, and anomaly detection, are among the most popular unsupervised learning tasks. K-means clustering, mean-shift clustering, density-based spatial clustering of applications with noise (DBSCAN), clustering of Gaussian mixture models (GMMs), Agglomerative hierarchical clustering, variance threshold, Pearson correlation, ANOVA, Chi-square, Recursive feature elimination (RFE),

Model-based selection, Principal component analysis (PCA), AIS and SETM, Apriori, Equivalence Class Clustering and bottom-up Lattice Traversal (ECLAT) and ABC-RuleMiner are a few examples of unsupervised learning algorithms [1]. Extraction of generating features, recognition of significant patterns and structures, result groupings, and exploratory uses are all common applications of unsupervised learning. Applications like anomaly detection, recommendation engines, and data visualization frequently employ unsupervised learning.

Since semi-supervised learning uses both labeled and unlabeled data, it can be thought of as a hybridization of supervised and unsupervised learning techniques [5, 6]. It, therefore, lies in the middle of learning "without supervision" and "with supervision". Semi-supervised learning is helpful in situations when labeled data are abundant and unlabeled data are few in the actual world. A semi-supervised learning model's ultimate objective is to produce a better prediction result than one might obtain from the model utilizing just the labeled data. Semi-supervised learning finds application in machine translation, fraud detection, data labeling, and text categorization, among other areas.

Reinforcement learning algorithms interact with their surroundings by taking actions and assessing the results. Trial-and-error learning and delayed rewards are two of the most important components of reinforcement learning. Reinforcement learning involves the computer learning by making mistakes and getting feedback for those mistakes in the form of incentives or punishments. Through exploration of the world and experience-based learning, the algorithm learns to make a series of decisions that maximize a cumulative reward over time. The goal of reinforcement learning is to increase efficiency by enabling software agents and computers to automatically determine the best course of action in a given situation. The agent requires the reinforcement signal, which is straightforward reward feedback, to decide which behavior is best. It is an effective technique for developing AI models that can help improve operational efficiency or automate more complex systems like robots, autonomous driving, recommendation engines, manufacturing, and supply chain management. It is not, however, the best method for resolving elementary issues.

Monte Carlo techniques, Q-learning, Deep Q-learning, policy gradient techniques, and actor-critic techniques are a few instances of reinforcement learning algorithms. Both the data and the learning algorithms play a major role in the final success of a machine learning-based solution and the related applications. Machine learning models may become ineffective or yield poorer accuracy if the data are unsuitable for learning, such as non-representative, low-quality, irrelevant features, or insufficient quantity for training. For a machine learning-based solution and ultimately the development of intelligent applications, it is crucial to handle the various learning algorithms and process data efficiently [1, 2, 3].

1.1 Deep Learning

Deep Learning (DL) is a branch of machine learning that focuses on algorithms that are inspired by the structure and operations of the brain. DL maps the provided input to particular labels using a substantial volume of data. It can function without any rules that were created by humans. The usage of deep neural networks, which comprise numerous layers of interconnected nodes, is a fundamental aspect of deep learning. By identifying hierarchical

patterns and features in the data, these networks can learn complicated representations of the data [7]. Without the need for human feature engineering, deep learning algorithms can automatically learn from and get better at analyzing data [8]. Pre-processing, feature extraction, feature selection, learning, and classification are the sequential processes needed to complete a classification job using traditional machine learning techniques. Moreover, a major influence on the effectiveness of machine learning approaches is feature selection. Inaccurate class distinction may result from biased feature selection. On the other hand, unlike traditional machine learning techniques, deep learning can automate the learning of feature sets for several tasks [9, 10]. Learning and categorization can be accomplished simultaneously with deep learning. On certain tasks, such image classification, deep learning performance has surpassed human performance in recent times [11].

According to current estimations, more than one trillion megabytes of data is being created daily [12]. Deep learning is made feasible by this astounding volume of data generation [13]. The main factor propelling the development of deep learning skills is the enormous increase in data generation. Virtual assistants, self-driving cars, chatbots, facial recognition, speech recognition, and medical science are just a few of the applications that deep learning can serve. Deep learning assists businesses and organizations in creating automated processes that result in better, faster, and less expensive outcomes [12].

Because most DL models are developed by supervised learning, building DL models involves a significant amount of time and resources, including training datasets. In the early days of DL, it was challenging to develop DL models. Nonetheless, two approaches have assisted researchers in getting over the previously described challenges. First, with the recent developments in computer hardware and software, researchers can now easily and effectively implement DL models [14]. Tensor processing units, graphics processing units (GPUs), and central processing units are examples of DL platforms that can be used to implement deep learning models [15]. Secondly, the majority of remote sensing applications demand big data analysis. This large-scale remote sensing data can be used to train deep learning models.

1.2 Deep Learning Approaches

Reinforcement learning, as well as supervised and unsupervised machine learning, can be accomplished with deep learning. Many supervised tasks, such as sentiment analysis, language translation, image classification and identification, and sentiment analysis, employ deep learning techniques such as convolutional neural networks (CNNs), recurrent neural networks (RNNs), and deep neural networks (DNNs) [8]. Long short-term memory (LSTM) techniques and gated recurrent units (GRUs) are included in the RNN category. Semi-supervised learning process relies on datasets that have been partially tagged. Deep reinforcement learning (DRL) and generative adversarial networks (GANs) are occasionally used in this method.

Furthermore, LSTMs and GRUs are examples of RNNs that are used for partially supervised learning. For text document classification tasks, semi-supervised learning is the best option because it is not easy to gather a large number of classified text documents. For unsupervised tasks including clustering, dimensionality reduction, and anomaly detection, deep learning methods such as generative adversarial networks (GANs), auto-encoders, and limited Boltzmann machines are employed. Furthermore, a variety of applications have utilized RNNs for unsupervised learning, including GRUs and LSTM techniques [11]. Robotics and gaming, among other activities, are reinforced by deep reinforcement learning algorithms such as Deep Q networks and Deep Deterministic Policy Gradient (DDPG) [8].

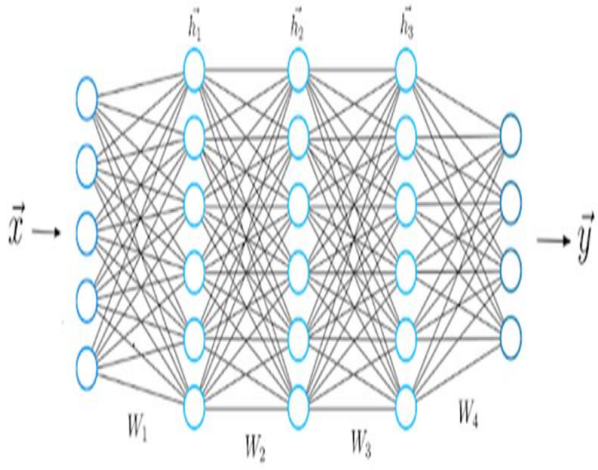

Figure 1: Architecture of Neural Network

a) Recursive neural networks (RvNNs)

Recursive neural networks can classify the outputs using compositional vectors and make predictions in a hierarchical structure. The main source of inspiration for the development of RvNN is recursive auto-associative memory (RAAM). RvNN is constructed for handling objects with randomly formed structures, such as trees or graphs. This method takes a variable-size recursive data structure and turns it into a fixed-width distributed representation. An introduced back-propagation through structure (BTS) learning system is used to train the network. The BTS system may accommodate a treelike structure and uses the same methodology as the general-back propagation algorithm. The network is trained to reproduce the input-layer pattern at the output layer by using Auto-association. RvNN is very effective in the area of Natural Language Processing [11].

b) Recurrent Neural Network (RNN)

Recurrent Neural Network is one of the types of neural networks that can handle sequential data, including time series and plain language. The recurrent neural network (RNN) is ideally suited for applications like speech recognition, natural language processing, and language translation and they can keep an internal state that records information about the prior inputs [8].

c) Convolutional Neural Networks (CNNs)

CNNs are incredibly powerful deep learning algorithms. They have been applied to computer vision applications and they yield innovative results [16]. CNNs are ideal for applications like object identification, image segmentation, and classification because they can automatically extract pertinent features from the images without human intervention. CNNs are made to handle data in numerous arrays, and they work particularly well for this purpose. Convolutional layer, pooling layer, fully connected layer, and softmax layer are all present in a typical CNN architecture. Convolutional layers are utilized to improve spatial features from the input data. To add nonlinearity to the model, an activation function is added to the convolution-layer output. Pooling layers lessen the spatial sizes of the input data, consequently decreasing the number of trainable parameters in the following layers and aiding them to focus on a wider variety of input patterns. As the input is processed in hidden layers, the amount of appropriate high-level features that are extracted rises. Lastly, data classification is done using the fully connected and softmax layers [14].

Furthermore, the output layer uses a few loss functions to determine the expected error that the CNN model will produce among the training data. The discrepancy between the expected and actual output is shown by this error. It will then be optimized via CNN's learning process. Nevertheless, loss function uses two parameters to compute the error. The first parameter is the CNN estimated output, also known as the prediction. The second parameter is the actual output, sometimes called the label. Different kinds of loss functions are used for different kinds of problems. These consist of the Hinge Loss Function, Euclidean Loss Function, and Cross-Entropy or Softmax Loss Function.

In deep learning, the idea of loss function indicates how poorly the model is performing at that given moment. It is necessary to train the network using the loss to ensure better performance. In essence, the loss function must be minimized since greater performance of the model is indicated by a smaller loss function. Optimization is the technique of maximizing or minimizing any mathematical expression [17].

An optimizer is a function or algorithm that modifies the neural network's parameters, like learning rates and weights. As a result, it aids in raising accuracy and decreasing overall loss. With millions of parameters in most deep learning models, selecting the appropriate weights for the model is a difficult issue. It highlights how important it is to select an appropriate optimization technique for a certain application [18].

Optimization can be done in a variety of ways, and each approach has benefits and drawbacks. Gradient descent is the most popular optimization technique used with neural networks. This technique entails continuously modifying the network's parameter values until performance increases. Different approaches and strategies can be used to optimize different kinds of problems. Gradient descent may not always be feasible, necessitating the employment of an alternative technique [19]. Stochastic Gradient Descent (SGD), Mini Batch Stochastic Gradient Descent (MB-SGD), Momentum, Nesterov Accelerated Gradient, Adaptive Gradient (AdaGrad), AdaDelta, RMSprop, and Adam (Adaptive Moment Estimation) are examples of

common gradient descent variations. To determine the ideal model parameters for increased performance, each optimizer includes unique update rules, learning rates, and momentum [11].

1.4 Convolutional Neural Network Architectures for Image Classification

Deep learning frameworks that are most widely used are CNN architectures. Not all CNNs are made equal, but they have demonstrated amazing progress in solving the image recognition problem by delivering a new degree of scalability and precision [20]. While it is feasible to create a CNN from scratch, several architectures that have been created and made available to the public can also be utilized. Additionally, some of these networks provide pre-trained models that are simple to modify for a certain use case [21].

Depending on how well they can train, CNN's architectures, which include LeNet, AlexNet, ResNet, ZFNet, MobileNets, VGGNet, GoogLeNet, ENet, Xception, DenseNet, Shuffile Net, Squeeze Net, and Inception, can be employed in a variety of application. Table 1 presents a brief overview of CNN architectures and their input size specification.

Table 1: CNN Architectures and Their Input Size Specification

Convolution Neural Network Architecture	**Image size**
AlexNet	227 × 227 × 3
CapsuleNet	28 × 28 × 1
Competitive squeeze and excitation network	32 × 32 × 3
DenseNet	224 × 224 × 3
FractalNet	32 × 32 × 3
GoogLeNet	224 × 224 × 3
Highway	32 × 32 × 3
HRNetV2	224 × 224 × 3
Inception-ResNetv2	229 × 229 × 3
Inception-V3	229 × 229 × 3
Inception-V4	229 × 229 × 3
MobileNet-v2	224 × 224 × 3
NIN	32 × 32 × 3
Residual attention neural network	40 × 40 × 3
ResNet	224 × 224 × 3
Squeeze-and-excitation networks	229 × 229 × 3 224 × 224 × 3 320 × 320 × 3
VGG	224 × 224 × 3
WideResNet	32 × 32 × 3
ZfNet	224 × 224 × 3

2. Fixed Input Size Limitation Associated with CNN

The images that we can obtain generally come in different sizes. The development of convolution neural network (CNN)-based computer vision technologies has been made easier by the abundance of these images. Target recognition, image classification, and many other computer vision tasks have benefited greatly from CNN-based computer vision technology [22]. CNNs are primarily made up of two layers: the convolution layer and the fully-connected layer which makes most CNNs have a limit in terms of training and prediction. The fully-connected layer demands uniformity in the size of all input data. A fixed size is not required in the convolutional layer because the number of parameters is independent of the input size. It can process inputs of any size and generate a feature map whose size is proportionate to the input size. Conversely, the number of inputs and outputs of the neurons directly affects how many parameters the fully connected layer has.

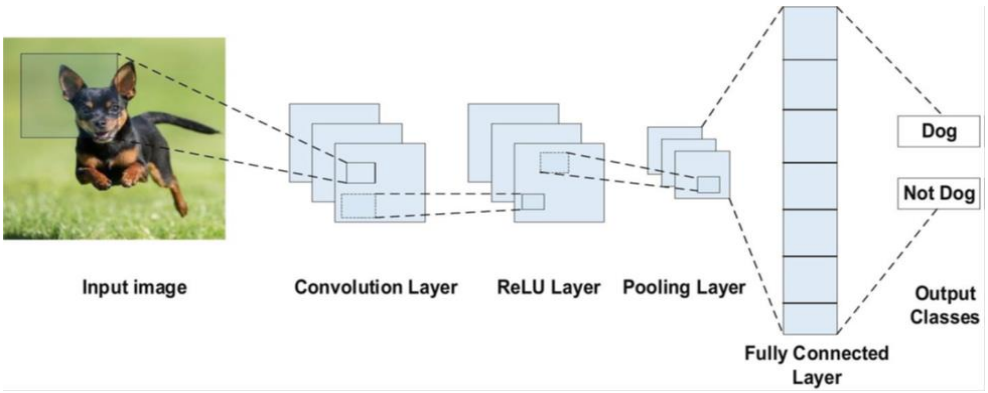

Figure 2: Convolutional Neural Network Architecture

The final output layer has a set number of neurons, which is equal to the number of classes. Iterative updates can only be used to learn parameters in networks where the number of parameters is fixed. As Table 1 above illustrates, many CNNs require fixed-size input because of the fully connected layer [23].

CNNs can produce state-of-the-art results in a wide range of tasks and domains, but an intrinsic shortcoming is their sensitivity to image size, which necessitates matching evaluation inputs to training image size thereby restricting practical use cases [24]. Convolution neural networks that are trained on a certain image size exhibit poor performance when evaluated on different image sizes [25]. Additionally, since consistency between input size is a necessary condition for typical Neural Network architectures and a really helpful condition for successfully training them, image recognition datasets are made up of a set of images that have been preprocessed to have the same height and width. Therefore, image recognition datasets are usually resized to 256x256 or something similar depending on the input size specification of the CNN architecture, In fact, employing larger images creates new training hurdles and problems with memory consumption [26].

2.1 Research Gap

The issue that many CNNs require all input data to be of the same size, despite the reality that images typically have different sizes, served as the impetus for this work [22]. The fixed input size requirement of existing CNN models for image classification tasks was deemed acceptable by the majority of researchers who have worked on these models. The researchers obtained their images from either public or private domains or both. They resized the images using warping or cropping techniques, trained the CNN model, classified the images, assessed the model's performance, and came to the conclusion that their model outperforms existing models without taking the fixed input size limitation into account. Furthermore, there is paucity of literature on the fixed input size limitation of CNN model. Consequently, the purpose of this work is to examine CNN models and propose potential solutions to tackle the fixed input size problem that exists in convolutional neural network models.

3. Methodology

The researchers reviewed existing literature on machine learning, deep learning approaches, and architectures, CNN architectures for image classification, and solutions to fixed input size problems in CNN models with the help of books, conference proceedings, and electronic databases such as Sciencedirect, Researchgate, and Google Scholar. The majority of the research articles used were published between 2018 and 2023. In our searches, we employed the following keywords: Artificial Intelligence, Machine Learning, Deep Learning, Convolutional Neural Network, and Fixed Input Size Limitation.

4. Solution to Fixed Input Size Limitation in CNN Models

A limited number of researchers have examined alternative approaches for utilizing large images in CNN systems that receive smaller input images. There are three strategies for employing large photos in CNN architectures that accept smaller input images: resizing the input image, increasing the size of the model, and processing images in batches. It is possible to adjust the input image to match the CNN's needed input size. One method of resizing an input image is to downscale it, which lowers the image's spatial resolution without affecting its two-dimensional representation. But in this instance, the well-known aliasing issue arises, where high-frequency variations such as shifting light and dark hues will translate into low-frequency variations such as continuous dark and light. Cropping the image to the desired size is another way to minimize its size. In order to preserve the central content, which is typically more important and beneficial for the image, the image is typically cropped from the center.

Maintaining a constant input size and adjusting the CNN architecture appropriately is another way to use large images in CNN models that accept smaller input images. Convolutional layers in particular can shrink an input image's size. Therefore, it is possible to create an architecture that can handle larger images by adding some more convolutional layers before the CNN convolutional layers. Reducing the total memory burden and processing the huge input images in batches is the third way to solve this issue.

More specifically, the inability of the entire training set to fit into memory occurs when employing large images. Mini-Batch Gradient Descent provides the solution to this problem. This involves processing a subset of images each time we loop through the dataset. Thus, alteration of the batch size can be done to accommodate the large images in the memory [27]. Local features are lost while training images in batches. Moreover, labels for each batch must be created when training on image batches, which might take a lot of time. [28].

In their article, [22] also mentioned that a popular approach for images of varying sizes is to crop or warp the image to force all of the images into a single, uniform size, and then do CNN training and learning. However, during the cropping procedure, a large number of pixels containing valuable information are lost, which will affect the training accuracy. In many situations, cropping is not appropriate [23]. Geometric distortion results from warping distortion, which destroys structural information like an object's angle and proportion. This makes it challenging to learn about many scientific data sets with geometric labeling or high accuracy requirements. Warping might not provide too much of a hindrance in learning tasks for many natural image datasets. However, as computer vision advances, CNN is processing an increasing amount of scientific data. This implies that many application scenarios will be sensitive to warp, and the demand for warped image preprocessing will not be met.

A certain kind of CNN architecture known as fully convolutional exists in the literature that it is capable of operating with varied input shapes. The architectural feature that sets it apart from the rest is that it fixes the shape of the system by combining all the spatial information into a single value before feeding fully connected layers. The capacity to feed images with varying shapes might be resolved by the fully convolutional layers, but learning patterns at such large scales will be extremely challenging for CNN [26]. The Global Average Pooling (GAP) concept was introduced in 2014 [29]. To get around the input's fixed-size limit, GAP can be used to replace the model's fully connected layer. The fully-connected layer is still crucial to many computer vision jobs, even though GAP has shown promise in certain areas.

The network's fixed-size restriction was eliminated by introducing a spatial pyramid pooling (SPP) layer, also referred to as spatial pyramid matching or SPM. In particular, the SPP layer was placed above the final convolutional layer. The fixed length outputs produced by the SPP layer after pooling the features are passed into the fully connected layers. In other words, they obviate the necessity for cropping or warping at the outset by doing some information aggregation at a lower level of the network structure between convolutional layers and fully-connected layers. A versatile method for managing various image sizes is to train a CNN with a spatial pyramid pooling layer [30].

5. Conclusion

This paper reviewed fixed input size limitations in convolutional neural network models and proposed possible solutions to address the problem. The solutions for using large images in CNN architectures are to: resize the input image by cropping or warping the image to force all images into one uniform size; add more convolutional layers before the convolutional layers of the CNN architecture; process the large input images in batches to reduce the overall memory load; using a fully-convolutional architecture that allows using variable input shapes; replacing the fully-connected layer of the model with Global Average Pooling (GAP) to avoid the fixed-size limit in the input image and lastly; replacing the last pooling layer after the last convolutional layer of the CNN model with a spatial pyramid pooling layer (SPP) to eliminate the fixed-size input limit regardless of the size or proportion of the input image.

References

[1] Iqbal, H. S. (2021). Machine Learning: Algorithms, Real-World Applications and Research Directions. *SN Computer Science*, 2:160, 1-21, https://doi.org/10.1007/s42979-021-00592-x

[2] Amapiano. (2023). Introduction to Machine Learning 2023: PDF Download. A-Brief-Introduction-to-Machine-Learning-for-Engineers. https://www.citimuzik.com/2023/03/introduction-to-machine-learning-2023-pdf-download.html.

[3] Anubhav, K.P., Dileep, K.M., Víctor D.J.M., Mohan, B.R, & Achyutha, P. (2023). Machine Learning Approach for Prediction of the

Online User Intention for a Product Purchase. *International Journal on Recent and Innovation Trends in Computing and Communication*, Volume: 11 Issue: 1s, page 41-51, DOI: https://doi.org/10.17762/ijritcc.v11i1s.5992

[4] Mohammed, M., Khan, M.B., & Bashier, M.B.E. (2016). Machine learning: algorithms and applications. CRC Press.

[5] Han, J., Pei, J., & Kamber, M. (2011). Data mining: concepts and techniques. Amsterdam: Elsevier.

[6] Sarker, I.H., Kayes, ASM., Badsha, S., Alqahtani, H., Watters, P., & Ng, A. (2020). Cybersecurity data science: an overview from machine learning perspective. *J Big Data*, 7(1):1–29.

[7] Jason, B. (2020). What is Deep Learning? Machine Learning Mastery. https://machinelearningmastery.com/what-is-deep-learning/

[8] Saumyasaxena. (2023). Introduction to Deep Learning. https://www.geeksforgeeks.org/introduction-deep-learning/

[9] LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. *Nature*, 2015; 521(7553):436–44.

[10] Shrestha, A., & Mahmood, A. (2019). Review of deep learning algorithms and architectures. *IEEE Access*, 7:53040–65.

[11] Laith, A., Jinglan, Z., Amjad, J.H., Ayad, A., Ye, D., Omran, A., Santamaría, J., Mohammed, A.F., Muthana, A., & Laith, F. (2021). Review of deep learning: concepts, CNN architectures, challenges, applications, future directions. *Journal of Big Data*, Pages 1 – 74. https://doi.org/10.1186/s40537-021-00444-8

[12] Coursera. (2023). What Is Deep Learning? Definition, Examples, and Careers. https://www.coursera.org/articles/what-is-deep-learning

[13] Techjury. (2022). How Much Data Is Created Every Day in 2022? https://techjury.net/blog/how-much-data-is-created-every-day/#gref.

[14] Chen, Y.N., Fan, K.C., Chang, Y.L., & Moriyama, T. (2023). Special Issue Review: Artificial Intelligence and Machine Learning Applications in Remote Sensing. *Remote Sens*. 2023, 15, 569. https://doi.org/10.3390/rs15030569. Page 1-10. https://www.mdpi.com/journal/remotesensing

[15] Wang, Y., Wei, G.Y., & Brooks, D. (2019). Benchmarking TPU, GPU, and CPU Platforms for Deep Learning. *arXiv* 2019, arXiv:1907.10701. Available online: https://arxiv.org/pdf/1907.10701.

[16] Vidya, M. M. (2023). Covid-19 Prediction Using Machine Learning. International Research. *Journal of Modernization in Engineering Technology and Science*, 5(5), 1180-1185, DOI: https://www.doi.org/10.56726/IRJMETS38449

[17] Nagesh, S.C. (2020). Optimization Algorithms in Neural Networks. KDnuggets. https://www.kdnuggets.com/2020/12/optimization-algorithms-neural-networks.html

[18] Ayush, G. (2023). A Comprehensive Guide on Optimizers in Deep Learning. Analytics Vidhya. https://www.analyticsvidhya.com/blog/2021/10/a-comprehensive-guide-on-deep-learning-optimizers/

[19] Faisal, M., Shabir, A., & Taeg, K.W. (2023). An Efficient Optimization Technique for Training Deep Neural Networks. *Mathematics 2023*, 11(6), 1360; https://doi.org/10.3390/math11061360

[20] Ajitesh, K. (2023). Different Types of CNN Architectures Explained: Examples. Analytics Yogi. https://vitalflux.com/different-types-of-cnn-architectures-explained-examples/

[21] Derrick, M. (2022). Image Classification with Convolutional Neural Networks (CNNs). KDnuggets. https://www.kdnuggets.com/2022/05/image-classification-convolutional-neural-networks-cnns.html

[22] Yili, Q., Yaobin, K., & Wei, Y. (2018). A Solution for Input Limit in CNN Due to Fully Connected Layer. IEEE Xplore.

[23] He, K., Zhang, X., Ren, S., & Sun, J. (2015). Spatial Pyramid Pooling in Deep Convolutional Networks for Visual Recognition .*[J]*. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, 37(9):1904-1916.

[24] Elad., H., Berry, W., Itay, H., Tal, B., Torsten, H., & Daniel, S. (2019). Mix & Match: Training Convnets with Mixed Image Sizes for Improved Accuracy, Speed and Scale Resiliency. *arXiv*, 1-13, https://github.com/eladhoffer/convNet.pytorch

[25] Hugo, T., Andrea, V., Matthijs, D., & Herv´e, J. (2019). Fixing the train-test resolution discrepancy. *CoRR*, abs/1906.06423. http://arxiv.org/abs/1906.06423.

[26] Ferran, P., Dario, G., & Jesus, L. (2023) Training CNNs using high-resolution images of variable shape. 6th BSC Severo Ochoa Doctoral Symposium. Page 60-61.

[27] Panagiotis, A. (2023). How to Handle Large Images to Train CNNs? https://www.baeldung.com/cs/

[28] Pinckaers, J.H.F.M., & Litjens, G.J.S. (2018). Training convolutional neural networks with megapixel images. *1st international conference on Medical Imaging with Deep Learning (MIDL 2018), Amsterdam*. Pages 1-3.

29] Lin, M., Chen, Q., & Yan, S. (2013). "Network In Network". *Computer Science*, 2013.

[30] Kaiming, H., Xiangyu, Z., Shaoqing, R., & Jian, S. (2015). Spatial Pyramid Pooling in Deep Convolutional Networks for Visual Recognition. *arXiv*:1406.4729v4 [cs.CV] 1-14