Comparative Analysis of Various Neural Networks with Dropout Technique

[1]Chandana C
[1]Assistant Professor
[1]Department of Computer Applications
[1]Jnana Jyothi Degree College, Yelahanka, Bengaluru-560061

***Abstract*—In recent years, neural networks have plays a vital role. One key technique that improves the neural network performance is the dropout technique, which helps to prevent overfitting. This paper presents a comparative study of various neural network architectures with dropout. We discuss different datasets used, specific models implemented, and also analysis the performance of the dropout on various neural network. This analysis Convolutional Neural Networks, Recurrent Neural Networks and Multilayer Perceptrons. Finally, we determine which neural network architecture benefits the most from the dropout technique based on empirical results.**

***Index Terms*—Convolutional Neural Networks, Recurrent Neural Networks, and Multilayer Perceptrons, dropout technique**

I. Introduction

Neural networks are stimulated by the human brain activity also these are computational models consisting of layers of interconnected neurons. Over the past few decades, are identified as a one of the dominant approach for resolving complex problems across various streams for example, natural language processing, and speech recognition. The flexibility and adaptability of neural networks allow to model detailed patterns within data, building them highly effective for a broad range of activity. But even with their great potential, neural networks frequently struggle with overfitting [25] When deep networks have a lot of parameters, this problem becomes more noticeable. Many regularization strategies have been developed to counteract overfitting; dropout has received a lot of attention. Dropout Technique: When a model learns too much from the noise and details in the training set, it is said to be overfitting and performs poorly on fresh data. As a result, there is poor generalization to test data but high accuracy on training data. Many regularization strategies have been developed to address this; Srivastava et al. (2014) introduced the dropout technique, which has proven to be one of the most successful. Dropout forces the network to learn redundant representations by randomly removing neurons during training, which helps to reduce overfitting. During each training iteration, a portion of the network's neurons are randomly set to zero as part of the dropout technique. As a result, the network is encouraged to learn more resilient and universal features by preventing the neurons from co-adapting excessively. All neurons are active during the testing phase, but in order to preserve consistency with the training phase, their outputs are scaled by the dropout rate. It has been demonstrated that this straightforward but effective technique enhances neural networks' generalization and performance on a variety of tasks.

II. RELATED WORK

CNN has already greatly aided in achieving performance in image recognition, including the ability to recognize fingerprints and handwriting [2] P. Baldi. et.al. In reference [1] N. Basit at.al, the writer presented a MapReduce-based method for training CNNs that outperformed sequential programming in terms of performance. CNN performs better in the healthcare industry as well. In [3] M. Halicek, the author built a CNN with six convolutional layers using TensorFlow. It finished identifying HSI from tissue and figure out the likelihood that a head or neck cancer would occur. In [4]

M. Anthimopoulos et al, the author used a CNN trained on a large number of CT scans to interstitial lung diseases. There are many CNN, frameworks available for deep learning implementation; most widely used ones are CudaCovNet [5] A. Krizhevsky et.al, Theano [6], Y. Jia et.al [7], 'Deeplearning4' (DL4J) [8] C. L. NYU et.al works well with Spark as an open-source library. [9] B.
J. Erickson et.al compared various open source libraries using the CNN structure. [10] V. Kovalev et.al examined deep learning frameworks' accuracy and processing speed. Additionally, [11] S. Bahrampour et.al compared to deep learning libraries using various deep learning libraries using various metrics. Since users can train models on Spark cluster, as mentioned in [12] A. Parvat et.al, The Street View House Numbers dataset (as it is explained in [13] N Srivastava et.al, these days, increasing processing efficiency for NN training has already been accomplished through the use of GPU and related parallel algorithms [4]. The author of [14] J. Dean et.al, provided details on the "MapReduce" programming model, which is focused on handling massive data streams. By using MapReduce to build a distributed computing CNN training structure, the author in [2] was able to speed up the train process by 50%. This piece motivated us to building blocks of neural networks [15]

J. Patterson et.al, if properly trained and given sufficient data, this intricate relationship between a network's weights can represent complex systems. The amount of computational time required to optimize the weights increases with network width and depth. However, most datasets in real-world problems are unbalanced and only small amounts are available; examples include bank fraud transactions compared to healthy transactions or rare diseases in medical imaging [16] N. Basit et.al, [17]

A. Vishnu et.al adding weight penalties in in the cost function of the networks such as L1 and L2 J. Schmidhuber et.al [18], and dropout [19] for online learning. [33] Diabetes is frequently the result of these eating habits. According to the Viademonte et.al [23]. Thus, the goal of this study is to improve the ability to recognize the food image of Thai food and use it as a personal assistant to modify eating habits. The detection and classification of images is a rapidly evolving field. Many useful fields and industries can benefit more from the application of image detection and classification technology. Mobile applications with features for image recognition and food health management have created a lot of convenient scenarios. Despite the excellent performance of these methods, computer systems are still unable to distinguish between different types of food, which leads to some results having poor accuracy. As a result, a new method was developed to streamline the system's workflow and make it more precise. Many methods have been developed recently in artificial intelligence and visual technology to help people track their health and use new apps. Hoashi at.al [24]. People's lives can be improved, for instance, by machine learning and photography. Today's photography, however suffers from issues with image sharpness, excessive light in the shot, and screen-captured images. Machine learning algorithm that can take images as input and assign meaningful tasks (learnable weights and biases) to various objects or characteristics in the image is the convolutional neural network CNN will then effectively distinguish one image from the others. One of the most difficult image recognition problems is food recognition. Pictures of food are often very complicated, with many different components that have different textures, colours, and shapes. Numerous methods, including deep learning, have been introduced in food recognition Salim et al. [25].

III. PROPOSED WORK

Convolutional Neural Networks (CNNs): It’s a deep learning model known as CNNs is specially made for dealing with images and the structured data. It captures the spatial hierarchies in the data. A specific proportion of randomly selected neurons are ignored (dropped out) during training in the dropout regularization technique. By lessening the network's sensitivity to the unique weights of individual neurons, this helps avoid overfitting. CNNs automatically takes the hierarchical features from the input data, which are images, by utilizing their operations and layers. working of CNNs as follow:

1. Input Layer: data is sent to input layer. For example, channels will be used to get a colored picture or image.

2. Convolutional Layers Convolution operations use filters and kernels, which are tiny matrices that traverse the input image. Every filter is initialized with random weights. Sliding Window: Convolution is the process by which the filter multiplies and sums the elements at each position as it passes the image. Features maps: Result of this process is a set of feature maps that highlights the various features that each filter picks up on, such as edges, textures, or patterns. computational process

3. Activation Function: To add non-linearity, an activation function is applied after convolution:
The network can learn more intricate patterns as a result.

4. Pooling-Layers: The feature spatial dimensions are decreased while crucial thing is kept by using pooling layers. Max Pooling: Utilizes the highest value found in every patch. Average Pooling: Utilizes the mean value across all patches Pooling facilitates the achievement of spatial invariance, or the ability to recognize objects regardless of where these are in the image, while also lightening the computational load.

5. Layer Stacking: To create deeper networks, several convolutional and pooling layers are stacked. Deeper layers identify more complex features (shapes, objects), whereas early layers may only detect basic features (colours, edges).

6. Distorting The multi-dimensional output to feed into the full connect layers after multiple convolutional and pooling layers.

7. Completely Networked Layers These layers' function similarly to a conventional neural network. All of the neurons in the layer above provide input to every neuron in a fully connected layer: Dense Layer: Activation function is applied after matrix multiplication. The Activation Function Frequently in classification tasks, use softmax for the output layer and ReLU for hidden layers. 8. Layer of Output, the final predictions are produced by the output layer, (soft max activation) functions is used to produce probabilities for each class in classification tasks:

Trainning CNNS: 1. Passage Forward Predictions are generated as data is routed layer by layer through the network. 2 Compute Loss: intended output and the predicted output is measured by the loss function. Typical loss functions consist of: For classification tasks, use cross-entropy loss. 3. Replication. The chain rule to calculate the gradient of the loss function wrt every weight as error is propagated back through network. 4. Weight Revision: The weights adjusted using an optimisation algorithm in order to reduce the loss and the gradient of the loss wrt weights is represented by MSE stands for mean squared error in regression tasks. CNN applications: Semantic Segmentation: Assigning a category to every pixel in an image (e.g., SegNet, U-Net). In medical scannings, such as MRIs and X-rays, is known as medical image analysis. The process of identifies or authenticating a person's face from a picture or video frame. objects Recognition in pictures, classifying Images, Object detection: The process of identifying and localizing objects within an image. Recurrent Neural Networks: Artificial neural networks that process data sequences are known as (RNNs), and these are especially useful for tasks where the sequence of the data is crucial. Feature connection that creates the directed cycles, which enable information persist between input sequence steps. This is a thorough description of RNNs:

1. Sequential Data Handling: RNNs are made to work with sequential data, including video frames, time series, natural language, and other types of data where context and sequence are crucial.

2. Recurrent Connections: These feature recurrent connections that loop back into the network, allowing it to keep track of information about earlier elements in sequence in a hidden state.

3. Hidden State: Based on previous hidden states and the current input the hidden states is updated at every time.

4. Output: Using the current hidden state as a basis, the output at each time step can be calculated: where represents the output bias and is the output weight matrix.
RNN Types-1Vanilla RNN: The most fundamental type of RNN, with the previously mentioned structure. It can have problems with vanishing and exploding gradients, which makes training on lengthy sequences challenging.

5. Enhanced RNN variant is intended to better capture long-term dependency. It maintains a more stable gradient by controlling the information flow through the use of gates (input, forget, and output gates).

6. Gated Recurrent Unit (GRU): An LSTM variation that combine input and forgets gate into a single update gate to streamline the gating mechanism. when Compared to LSTMs, GRUs have lesser parameters and may be simpler to train(RNNs) process sequential data piece by piece while preserving a hidden state that stores information about the elements in the sequence that came before it. This is a thorough description of how RNNs operate: Input Data Representation - RNNs process sequential data, where a vector or tensor is used to represent each successive element. For eg, each word in a sentence can be denoted as a word embedding vector in natural language processing.
Recurrent Connections: Recurrent connections in RNNs enable information to stay consistent between input sequence steps.

Output Computation: Based on the current hidden state, the output at each time step, denoted as, that can be computed. - Numerous tasks, including sequence generation, regression, and classification, can be accomplished with this output. Training: Back propagation through time (BPTT), an extension of back propagation intended for sequences, it is used to train RNNs. At every time step, the goal is to between the target output and the predicted output. Unrolling in Time: RNNs can be conceptually understood as being unrolled over time, with each time step representing a layer in the network that has unfolded. The length of the input sequence determines how many time steps the RNN is effectively unfolded during training, and backpropagation is carried out using this unfolded network. An artificial neural network also called as Multi-Layer Perceptrons (MLPs) is builds upon several layers of nodes, or neurons, each connected to layer above it. Because these are a specific kind of feed forwarding in neural networks, there are no cycles in the connections between the neuron. Information takes forward in multilayer parsers (MLPs) from the input layers to output layers via hidden layers. After taking inputs from the neuron in the previous layers, every neuron in that layer adds a weight to the total, applies a stimulation function to the total, and then transmits the results to the neurons in the layer below. Training- Gradients descent, other supervised learning algorithms are commonly used to train multilayer parsers (MLPs). The network's weights are iteratively changed during training in order to reduce the loss functions that calculate the discrepancy between the network's prediction and the actual labels in the training set. Backpropagation: Essential algorithms for MLP is back propagation enabling effective gradient descent weight adjustment Benefits-Versatility: Multilayer perceptron’s (MLPs) are useful for a change of tasks, such as classification, regression and even reinforcement learning. Non-linear Modelling: MLPs can represent intricate relationships between input and the output because these have non-linear activation functions.

IV. RESULT AND DISCUSSION

A group of handwritten numbers well-known as MNIST (Modified National Institute of Standards and Technology database) is often used dataset to testing and training purposes in machine learning, specially in the areas of computer vision and image processing. Here are MNIST dataset's salient features: This contains of 70,000 digit handwritten images, which grayscale image with the dimensions of 28*28 pixels. The training set takes 60,000 photos. 10 thousand photos taken by the test set. 0 and 9 is used to identify each image. An image intensity (grayscaled) is denoted by an integer that ranges from 0 to 255, where 0 denotes as black and 255 denotes as white. A label (0–9) designating the digit that each image

represents is attached to it. To study how well machine learning algorithms, perform, particularly in classification tasks, MNIST is used as a benchmark. For learning about machine learning and pattern recognition, it acts as an introductory dataset. It is used by scholars to test algorithms and study how well these perform in comparison to current model. By splitting by 255, pixel values are normalized to the interval 0,1. For some purposes, each 28 x 28 picture can be flattened into a 1D array with 784 features. Here is the detailed sample input dataset and data pre-processing steps in table format for CNN, RNN, and MLP models, for every model MNIST dataset is used to encompass the comparative study.

Table 1 Shows the details of dataset

For CNNs-

Here we use MNIST dataset. By randomly setting a portion of the input units to 0 at each training update, Regularization techniqueknown as "dropout" helps avoid overfitting.

Attribute	Value
Dataset	MNIST
Number of images	70,000
Image size	28*28 pixels
Number of class	10(digits 0-9)
Colour mode	grayscale

This shows the sample output of the first 10 rows of precision, recall, f1-score and support *Table2.shows the first 10 rows of precision, recall, f1-score and support*

Precision	recall	F1-score	Support
0.99	0.99	0.99	980
0.99	0.99	0.99	1135
0.98	0.99	0.99	1032
0.99	0.98	0.99	1010
0.99	0.99	0.99	982
0.99	0.98	0.98	892
0.99	0.99	0.99	958
0.99	0.99	0.99	1028
0.99	0.98	0.98	974
0.99	0.98	0.98	1009

Table3 shows the performance matrics of CNNs

For MLPs:

This shows the MLPs model performance on the MNIST dataset, this shows the sample output of the first 10 rows of accuracy, precision, recall, and F1 score

Table4.shows the first 10 rows of precision, recall, f1-score and support

Precision	recall	F1-score	Support
0.99	0.98	0.99	980
0.98	0.99	0.99	1135
0.97	0.99	0.97	1032
0.98	0.98	0.98	1010
0.98	0.98	0.98	982
0.98	0.97	0.98	892
0.98	0.98	0.98	958
0.97	0.98	0.98	1028
0.97	0.97	0.97	974
0.97	0.97	0.97	1009

Table5 shows the performance matrics of MLPs

Matrics	value
Accuracy	0.98
Precision	0.98
Recall	0.98
F1-Score	0.98

For RNNs-

This shows the RNN model's performance on the MNIST dataset, this shows the sample output of the first 10 rows of accuracy, precision, recall, and F1 scores.

Table6.shows the first 10 rows of precision, recall, f1-score and support

Precision	recall	F1-score	Support
0.98	0.99	0.99	980
0.99	0.99	0.99	1135
0.98	0.98	0.98	1032
0.97	0.98	0.98	1010
0.98	0.98	0.98	982
0.98	0.98	0.98	892
0.99	0.98	0.99	958
0.98	0.98	0.98	1028
0.98	0.97	0.98	974
0.97	0.98	0.98	1009

Matrics	value
Accuracy	0.97
Precision	0.98
Recall	0.98
F1-Score	0.98

Table7 shows the performance matrics of MLPs

Matrics	value
Accuracy	0.99
Precision	0.99
Recall	0.98
F1-Score	0.98

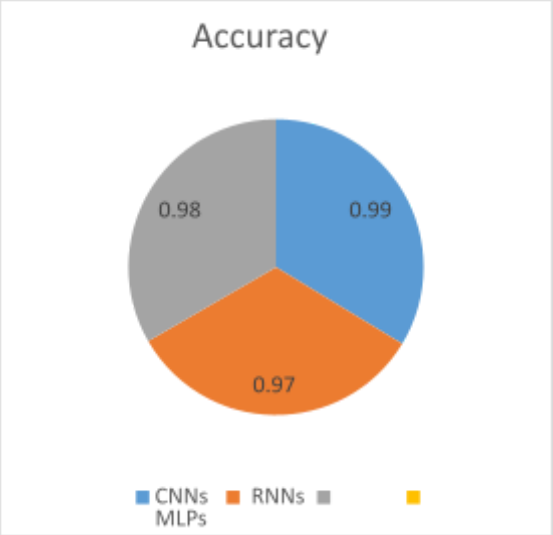

Fig.1 Represents the accuracy of the models

From these analysis, we can get to know the best performing among three models that is CNNs, On the MNIST dataset, CNNs outperform RNNs and MLPs. The main explanation is that convolutional layers in CNNs are specifically made to handle image data, capturing features like edges and shapes as well as spatial hierarchies. For tasks like digit recognition in the MNIST dataset, this makes them very effective. RNNs perform marginally worse when dealing with image data, despite being very effective with sequence data. MLPs perform worse than CNNs but similarly to RNNs on this specific task because, despite their versatility and simplicity these are less effective at capturing spatial hierarchies, because CNNs are so good at extracting and processing spatial features quickly, these are the optimal option for image recognition tasks such as MNIST. CNNs: Utilizing convolutional layers resulted in 0.25 dropout rates, while fully connected layers produced 0.5 dropout rates. RNNs: Recurrent dropout was combined with a dropout rate of 0.2. MLPs: Following dense layers, dropout rates of 0.5 were applied.

V. CONCLUSION

CNNs exhibits the best performance among various various neural network such as RNNs and MLPs with dropout technique using MNIST dataset. RNNs perform marginally worse when dealing with image data, despite being very effective with sequence data. MLPs perform worse than CNNs but similarly to RNNs on this specific task because, despite their versatility and simplicity, these are less effective at capturing spatial hierarchies, because CNNs are so good at extracting and processing spatial features quickly, these are the optimal option for image recognition tasks such as MNIST.

References

[1] N. Basit, Y. Zhang, H. Wu, H. Liu, J. Bin, Y, and A. M. Hendawi, "Mapreduce-based deep learning with handwritten digit recognition case study," 2016 IEEE Intl Conf on Big Data (Big Data), pp. 1690–1699, 2016

[2] P. Baldi and Y. Chauvin, "Neural networks for

[3] M. Halicek, G. Lu, J. V. Little, X. Wang, M. Patel, C. C. Griffith, M. W. El-Deiry, A. Y. Chen, and B. Fei, "Deep convolutional neural networks for classifying head and neck cancer using hyperspectral imaging," Journal of Biomedical Optics, vol. 22, no. 6, p. 060503, 2017

[4] M. Anthimopoulos, S. Christodoulidis, L. Ebner, A. Christe, and S. Mougiakakou, "Lung pattern clas- sification for interstitial lung diseases using a deep convolutional neural network," IEEE trans on medical imaging, vol. 35, no. 5, pp. 1207–1216, 2016

[5] A. Krizhevsky, "cuda-convnet," https://code.google. com/archive/p/cuda-convnet, July 2014 October 27, 2017.

[6] Theano: A Python framework for fast computation of mathematical expressions, arXiv e-prints, May 2016, accessed October 27, 2017. Available: http://arxiv.org/abs/1605.02688

[7] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. [28] Darrell, "Caffe: Convolutional architecture for fast feature embedding," arXiv preprin arXiv:1408.5093, 2014, accessed October 27, 2017

[8] C.L.NYU, "Torch7," http://cilvr.nyu.edu/doku.php?id=code:start, accessed October 27, 2017.

[9] B. J. Erickson, P. Korfiatis, Z. Akkus, T. Kline, and K. Philbrick, "Toolkits and libraries for deep learning," Journal of Digital Imaging, vol. 30, no. 4, pp. 400–405, [Online]. Available: https://doi.org/10.1007/s10278-017-9965-6

[10] V. Kovalev, A. Kalinovsky, and S. Kovalev, "Deep learning with theano, torch, caffe, tensorflow, and deeplearning4j: Which one is the best in speed and accuracy?" 2016.

[11] S. Bahrampour, N. Ramakrishnan, L. Schott, and M. Shah, "Comparative study of caffe, neon, theano, and torch for deep learning," 2016

[12] A. Parvat, J. Chavan, S. Kadam, S. Dev, and V. Pathak, "Asurvey of deep-learning frameworks," 2017 IEEE Intl Conf on in Inventive Systems and Control (ICISC), pp. 1–7.

[13] N. Srivastava, G. E. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a sim- ple way to prevent neural networks from