Dropout :An Effective Approach to Prevent Neural Networks from Overfitting

Abstract

Overfitting remains a significant challenge in training neural networks, often leading to poor generalization on unseen data. Dropout has emerged as a powerful regularization technique to mitigate overfitting by randomly deactivating neurons during training, thereby preventing co-adaptation of features and encouraging diverse representations. This paper explores the theoretical foundations and practical implementations of dropout across various neural network architectures, including Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs). Through empirical analysis on benchmark datasets such as CIFAR-10, MNIST, and others, dropout is shown to improve model robustness and accuracy significantly. The study also compares dropout with alternative regularization methods, such as weight constraints and batch normalization, highlighting its effectiveness in diverse scenarios. Despite its success, dropout's performance is influenced by hyperparameter tuning and dataset characteristics. The paper concludes by discussing limitations, such as computational overhead, and proposes directions for optimizing dropout for specific applications, including dynamic dropout rates and hybrid regularization techniques.

Keywords

Overfitting, Dropout, Neural Networks, Regularization, Convolutional Neural Networks (CNNs), Recurrent Neural Networks (RNNs).

1- Introduction

Deep neural networks have become indispensable across a wide array of domains, including computer vision, natural language processing, and quantum machine learning, delivering groundbreaking performance. However, their complexity and over-parameterization often lead to overfitting, where models memorize the training data at the expense of

generalization to unseen inputs. Addressing this limitation is critical for deploying neural networks in real-world applications.

Dropout, a regularization technique introduced by Srivastava et al. (2014), has emerged as one of the most effective solutions to prevent overfitting. By randomly deactivating a fraction of neurons during training, dropout forces the network to learn robust and diverse feature representations. This process can be interpreted as training an ensemble of subnetworks, significantly improving generalization and reducing the risk of co-adaptation among neurons[1][2][3].

Recent innovations have enhanced dropout's utility. FocusedDropout targets foreground features by selectively dropping units unrelated to the classification target, enhancing network focus and accuracy in tasks like image classification and object detection[4]. Energy-based dropout (EDropout) combines regularization with network pruning by optimizing neuron selection based on energy loss, achieving substantial reductions in parameter count while maintaining accuracy[2][5]. Furthermore, integrating dropout into quantum convolutional neural networks (QCNNs) has highlighted its potential in emerging paradigms like quantum machine learning, though with unique adaptations to handle quantum entanglement[3].

Dropout's flexibility extends to various neural network layers, including visible, hidden, convolutional, and recurrent layers. It has been integrated into long short-term memory (LSTM) networks for sequence processing and convolutional networks for spatial data, showcasing its versatility. Techniques such as Monte Carlo dropout enable uncertainty quantification, critical for applications requiring probabilistic predictions, while adaptive dropout approaches dynamically adjust dropout rates for optimal performance[5][1].

Beyond conventional regularization, dropout also plays a pivotal role in model compression and efficient training. Strategies like applying dropout selectively to high-weight neurons or integrating it with additional regularization methods like weight decay have proven effective in balancing robustness and performance[4][5]. These advancements highlight dropout as not just a tool for mitigating overfitting but as a central component in designing efficient, adaptive, and generalizable neural networks.

This paper delves into dropout's foundational principles, recent developments, and applications across classical and quantum machine learning. By exploring its transformative impact on neural network training and generalization, we aim to underscore dropout's role as a cornerstone technique in modern deep learning.

Background theory

2. Theoretical Concept of Dropout

Dropout neuronal units are developed using stochastic volatile memristive devices to address overfitting and nonideal synapse issues in neural networks. These units help achieve high classification accuracy by mitigating the negative effects of overfitting and nonideality in synaptic performance. The stochastic and volatile switching performances of the devices contribute to these benefits[6].

Dropout is used to interpret a DNN as a Bayesian model, allowing for uncertainty evaluation in predictions. The paper proposes a sampling-free method to evaluate uncertainty by converting a network trained with dropout into a Bayesian neural network with variance propagation. This approach is computationally efficient and statistically reliable, extending beyond feed-forward networks to include recurrent networks like LSTM[7].

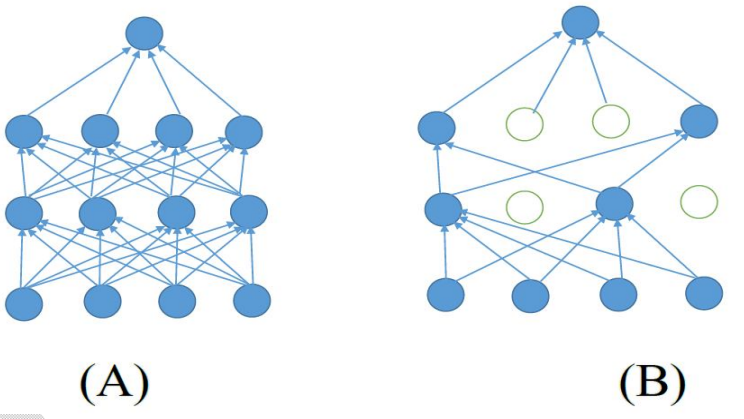
In recent times, deep learning architectures have significantly evolved and expandedleading to improvements in various tasks, such as classification, object detection, and segmentation,etc. But, he utilization of larger and more complex deep learning architecturescomes with the downside of increased risks of overfitting during training. In order totackle the challenges in deep learning, researchers have put forward various regularizationtechniques with dropout being particularly prominent among them.The standard dropout method can be used to relieve the overfitting issue, whichrandomly drops neurons from the neural network during training.For example, consider a neural network with an individual hidden linear layer of N units.Then the activation function Softmax is identical, taking the geometric arithmetic mean of the final output of the 2^N workable networks under standard Softmax . This method iscloser to a machine learning approach, such as the bagging method , which has trainedthe instance separately and the output inference has used the arithmetic mean.Let us consider a single linear layer within a neural network. This layer is commonlyreferred to as a linear layer because it incorporates the behavior of a linear

activationfunction, f (x) = x. In this (Figure 1) layer, the final neuron (which serves as the outputof the layer) is obtained by calculating the weighted sum of all the inputs. While thissimplified mathematical explanation holds empirically for certain non-linear networks, itis important to note that the estimation of the model involves minimizing a loss function[8].The ordinary least square (OLS) loss,

$$L_n = \frac{1}{2}\left(t - \sum_{i=1}^{n} w_i I_i\right)^2 \quad (1)$$

$$L_D = \frac{1}{2}\left(t - \sum_{i=1}^{n} \delta_i w_i I_i\right)^2 \quad (2)$$

Equation (1) represents the loss function used in general neural networks, whileEquation (2) corresponds to the loss function specific to dropout networks. In the dropoutnetwork, the dropout rate is denoted by *d*, which follows a Bernoulli distribution withparameter p. This implies that *d* takes value 1 with probability p and 0 otherwise. In thegiven network, the input is denoted by "I" and the weight associated with each input isrepresented by "w".



8]

Figure 1. An illustration of a standard neural network (**A**) and after applying dropout (**B**).

During network training, the gradient descent approach is utilized for backpropagation.This results in the computation of the gradient of the dropout network, denoted byEquation (2), which subsequently feeds into the regular network, as shown in Equation (1)[8].

$$\frac{dL_D}{dw_i} = -t\delta_i I_i + w_i \delta_i^2 I_j^2 + \sum_{j=1, j\neq i}^{n} w_j \delta_i \delta_j I_i I_j \quad (3)$$

Now, a relationship can be established between the gradient of the dropout networkand the gradient of a regular network. Based on Equation (1), we can assume thatw0 = p _ w, where p represents the probability dropout variable. Therefore, this equationindicates that the weights in the regular network are scaled by the dropout probability, p[8].

$$L_D = \frac{1}{2}\left(t - \sum_{i=1}^{n} p_i w_i I_i\right)^2 \tag{4}$$

Taking the derivative of Equation (4),

$$\frac{dL_D}{dw_i} = -t p_i I_i + w_i p_i^2 I_j^2 + \sum_{j=1, j\neq i}^{n} w_j p_i p_j I_i I_j \tag{5}$$

Now, move on to the next step. When we calculate the expectation of the gradient forthe dropout network, we obtain the following expression:

$$\begin{aligned} L\left[\frac{dL_D}{dw_i}\right] &= -t p_i I_i + w_i p_i^2 I_j^2 + w_i Var(\delta_i) I_i^2 + \sum_{j=1, j\neq i}^{n} w_j p_i p_j I_i I_j \\ &= \frac{dL_N}{dw_i} + w_i Var(\delta_i) I_i^2 \\ &= \frac{dL_N}{dw_i} + w_i p_i (1 - p_i) I_i^2 \end{aligned} \tag{6}$$

According to Equation (6), the expectation of the gradient with dropout is equivalentto the gradient of the regular neural network LN when the weights are scaled by p, denotedas $w0 = p \_ w$[8].

3. Contribution of Dropout Regularization Methods

explores implicit self-regularization in deep neural networks using dropout and its effects as a form of regularization that is inherently introduced through the training process. It discusses the various stages of self-regularization observable through dropout during the training process, which contributes to the robustness of the network against overfitting and how it can influence the generalization capabilities of the network[9].

In this section, we discuss the contribution of several commonly used dropout implementsbased on their performance under the dropout operation. The advantages ofdropout regularization methods play a vital role in consolidating knowledge, highlightingkey benefits and

inspiring further research in the field of deep learning. The primary aim ofthis section is to highlight the advantages of dropout regularization methods and providea comprehensive and consolidated overview of the benefits. In the following section, weprovide a concise overview of the benefits that these methods offer[8].

3.1. Effectiveness Improving

- **SpatialDropout**: Improves dropout in convolutional layers by dropping entire feature map channels, which can be more effective than traditional dropout for spatially correlated features[10].
- **DropBlock**: Further enhances effectiveness by dropping contiguous regions within the feature maps, encouraging the network to adapt to the absence of more significant portions of data, improving generalization over traditional methods[10].

3.1.1. Data augmentation

Utilizes various data manipulation techniques to enhance the volume and diversity of training data, which helps in improving the model's robustness and accuracy by simulating a wider array of possible input scenarios [11].

3.1.2. Preventing overfitting

Dropout and its Variants (SpatialDropout, DropBlock): These techniques randomly deactivate a subset of neurons during training to prevent them from co-adapting too much, which helps in avoiding overfitting by ensuring that the network generalizes well to new, unseen data [10][4].

3.1.3. Enhancing data representation.

Techniques like **DropBlock** not only prevent overfitting but also force the network to learn more robust features by not relying on any specific set of neurons, thus enhancing how data is represented internally within the network[10].

3.1.4. Preventing over-smoothing

FocusedDropout: Targets over-smoothing by selectively retaining features directly related to the target variable, thereby preserving essential information while still benefiting from the regularization effects of dropout[4].

3.2. Efficiency Improving

Dropout techniques can improve both the effectiveness and efficiency of a modelacross several processes[8].

3.2.1 Model Compression

Several dropout methods have been utilized for model compressionpurposes. By using these methods, the model structure can be madeeasier to compress after the random dropout of neurons, such as by performing neuralpruning. Model summarization methods are used to reduce the number of modelparameters, which can lead to improved training efficiency and reduced overfitting[8].

3.2.2. Model uncertainty estimation

Dropout techniques help in estimating model uncertainty by introducing stochasticity in the neural network's predictions, making the network's output sensitive to the absence of certain neurons, which can be interpreted as a measure of uncertainty[12].

3.2.3. Accelerate GCN training

In a graph convolutional network, the node feature informationsampling process has been proposed by GraphSAGE (sample and aggregate) ,which efficiently accelerates GCN training. The training purpose requires only someneighbor nodes to execute the training procedure. After conducting our work, weobserved similarities between our approach and the FastGCN and AS-GCN methods[8].

4. Categorization of Dropout Regularization Approaches

During the period from 2012 to 2022, we categorized each dropout regularizationapproach based on its year of introduction and development. The timeline of these dropoutregularization approaches is visually depicted in Figure 2. The graphical representationdepicts the temporal continuity of dropout regularization techniques, providing a clearoverview of their scenario over the years.Figure 3 illustrates the taxonomy of dropout approaches, categorized into three maincategories: internal structure change, data augmentation, and input information. Eachcategory further consists of several subcategories. By organizing the dropout approaches inthis taxonomy, we aim to provide a clear and systematic overview of the

different types ofdropout techniques based on the changes they induce in the internal structure, their usagein data augmentation, and their impact on input information.This taxonomy serves as a valuable reference for nderstanding the various dropoutapproaches employed in deep learning. The following section provides an indepth analysisof the taxonomic classification[8].

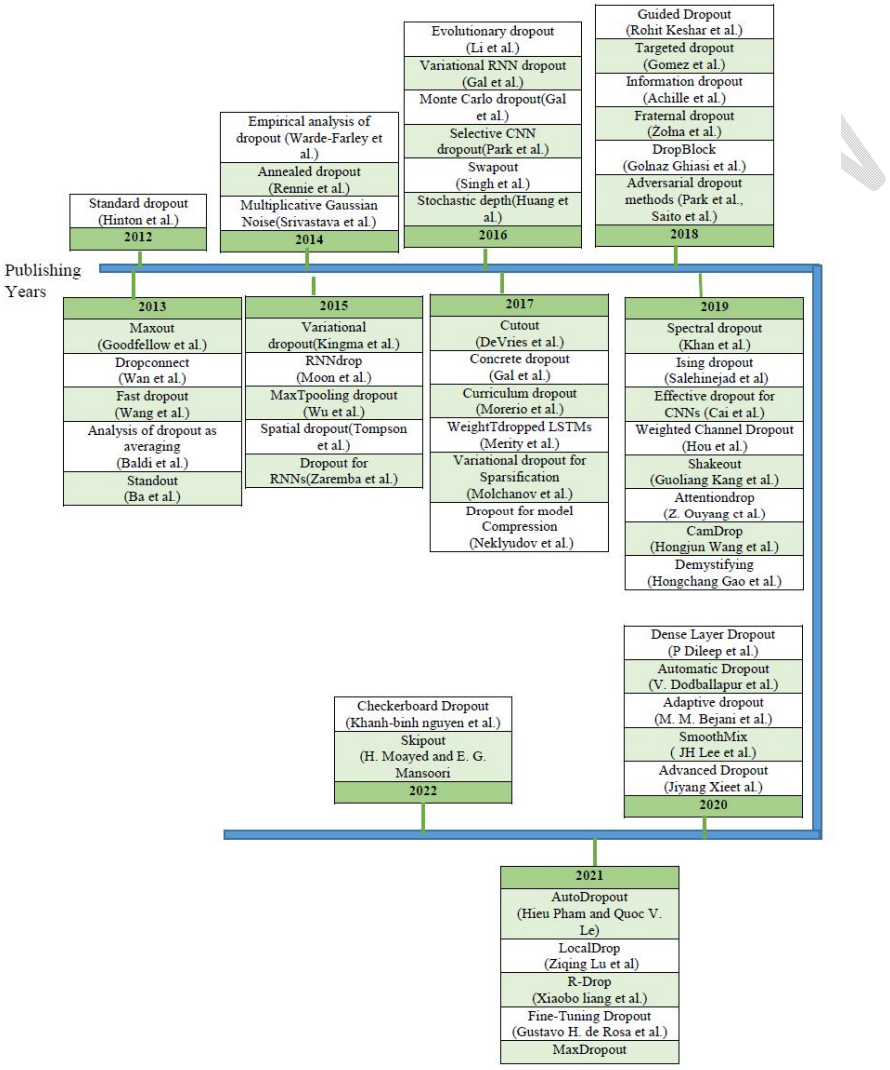


Figure 2. A flow diagram illustrating the progression of dropout methods over the years .

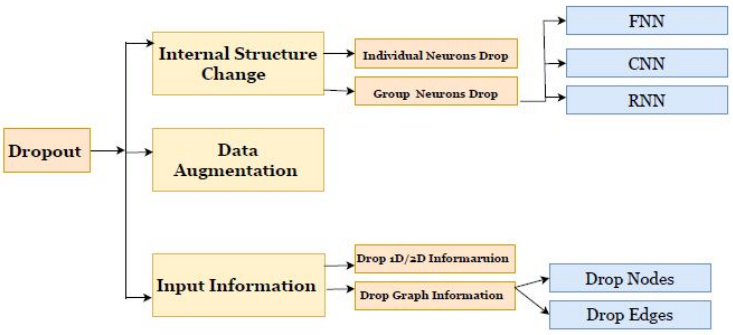


Figure 3. Taxonomy diagram according to dropout approaches.

5. Dropout Based on Internal Structure Changes

Various methods have been suggested for regularization in deep learning, beyond just dropout. Our review paper introduces the concept of internal dropout-based regularization, which alters weights and kernels during training but does not modify the input values. Traditional individual neuron dropout, as previously noted, involves randomly deactivating certain neurons during the training process. Each neuron is assigned a dropout probability, leading to its output being zeroed. This technique reduces dependence on particular neurons, aiding in the prevention of overfitting and promoting the learning of more stable and generalizable features. Conversely, group neuron dropout adopts a different strategy by deactivating whole groups or subsets of neurons at once, rather than individual ones. These groups might be organized by various criteria, such as their spatial location or their functional role within the network. We explore these and other prevalent methods in more detail in the subsequent subsection[8].

5.1. FNN and CNN-Based Neurons Drop Approaches

In this section, we examine a range of well-known neuron dropout strategies applicable to both feedforward neural networks (FNNs) and convolutional neural networks (CNNs)[8].

5.1.1 DropBlock

DropBlock is a structured form of dropout for convolutional layers where units in a contiguous region of a feature map are dropped together. This method is more effective than traditional dropout for CNNs as it handles the spatial correlation between units in convolutional layers [10] .

Deep neural networks generally achieve high performance when their parameters areover-optimized and trained with significant amounts of noise and regularization techniqueslike weight decay and dropout. In the course of this research, a method known as

Drop-Block is a form of a structured dropout that involves removing all units located within acontinuous region of a feature map (tensor). The size of the dropped regions (Figure 4) isdetermined by a hyperparameter, and during each training iteration, the dropped regionsare randomly selected.[8].

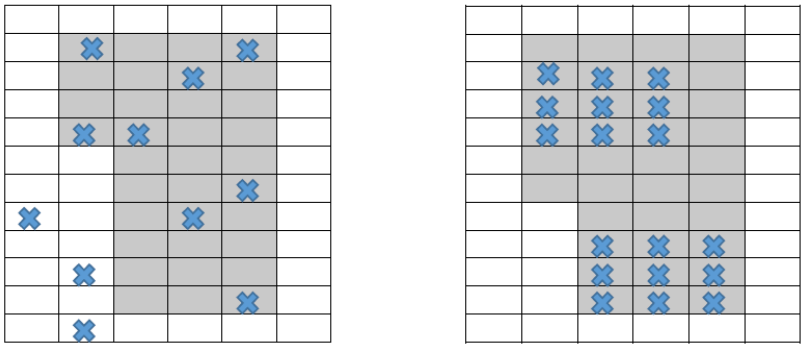


Figure 4. DropBlock structure visualization.

Moreover, gradually increasing the number of units discarded during the trainingprocess leads to greater accuracy and makes the model less sensitive to variations inhyperparameters, such as the learning rate and batch size. The image classification taskusing AmoebaNet-B and ResNet-50 models has been shown to improve the accuracy ofthe model compared to other variations of dropout. Based on the experimental results,using dropout techniques such as AmoebaNet-B and ResNet-50 can improve the accuracyof image classification tasks compared to other dropout variations. Specifically, on theImageNet dataset, the baseline accuracy of 2% for ResNet-50 was beaten by using cutoutand AutoAugment and AmoebaNet-B achieved an improvement of around 0.3% accuracy[8] .

5.1.2 MaxDropout

While Dropout randomly removes the neurons in the training phase, Maxdropout deactivatesthe neurons based on their activations. It first normalizes the tensor's values and then setsto 0 every single output greater than a given threshold *p*, so the higher this value, the most likelyit to be deactivated. The original work shows it can improve ResNet18 results on CIFAR-0 andCIFAR-100 datasets, and it also outperforms Dropout on the WideResNet-28-10 model [11]**.**

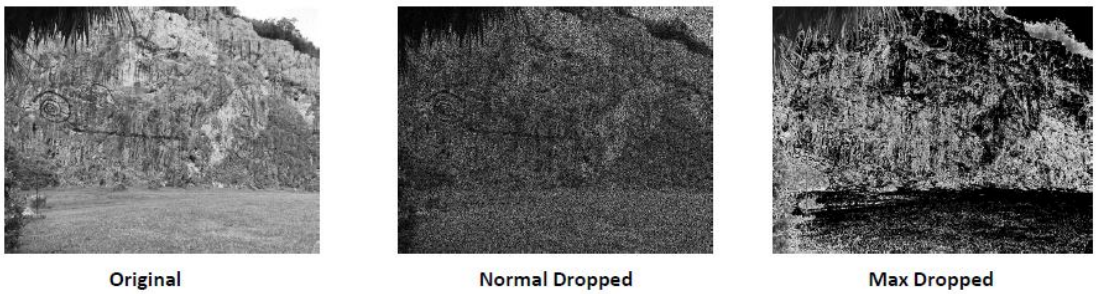


Figure 5. This is a visualization comparing the effects of normal dropout and MaxDropout on anoriginal image[8].

5.1.3 AutoDrop Dropout

AutoDropout automates the learning of dropout patterns via reinforcement learning, effectively adapting dropout rates and patterns to optimize network training. Though computationally intensive, it represents an advanced approach to dynamically adjusting dropout to enhance model training outcomes [12].

5.1.4 Attentiondrop

This method involves adaptively dropping features based on attention information. It prioritizes features that are deemed less critical for the current task, thus forcing the network to rely on a broader set of features and potentially improving generalization[4] **.**

5.2 Dropout with Recurrent Neural Networks (RNNs)

One of the primary obstacles in implementing dropout with Recurrent Neural Networks (RNNs) is their recurrent nature, where the same weights are reused across different time steps. This recurrence can lead to issues when dropout causes varying units to be excluded at each time step, potentially leading to erratic and unstable behaviors that degrade RNN performance. Yarin Gal and ZoubinGhahramani have developed a robust method for integrating dropout regularization into RNNs. Their paper presents a theoretically sound approach to applying dropout, with a mathematical model that incorporates noise addition to the hidden units. V. Pham and colleagues have shown that this technique of dropout regularization can enhance RNNs' effectiveness in tasks like handwriting recognition by applying dropout exclusively to feedforward connections and not to recurrent ones. Practically, they introduced dropout through a dedicated layer that maintains input values except at the dropped points. Another concept, "recurrent dropout," uses a "masking" tensor that affects the hidden state before activation. This tensor is randomly generated during each training cycle, based on a specified dropout rate. [8].

6 . AVOID OVERFITTING

To combat overfitting in deep neural networks, techniques like early stopping and dropout are employed. Early stopping prevents the model from learning too closely from the training dataset by halting the training process when improvements on a validation set cease. This method avoids the model's tendency to fit the noise and peculiarities in the training data rather than generalizing from it. On the other hand, dropout enhances the generalization of neural networks by randomly deactivating certain neurons during the training process. This randomness forces the network to learn more robust features that can perform well on unseen data, as it cannot rely on the presence of any particular neuron. Both techniques are crucial in developing neural network models that are not only deep and complex but also capable of generalizing well to new, unseen data[13].

7. Literature review

Kaiyu Zhang(2020) addressed the challenges of small sample sizes in fault diagnosis by proposing a compact convolutional neural network (CNN) augmented with a multiscale feature extraction unit. This unit used 1D convolutional kernels of varying scales to extract diverse features, reducing network depth and mitigating overfitting. The compact CNN, consisting of shallow layers and a softmax classifier, processed these features efficiently while employing dropout to enhance generalization. The approach was evaluated on three datasets, demonstrating high accuracy and strong generalization, outperforming traditional CNNs and SVMs. Future work included automating hyperparameter selection, improving generalization for dynamic conditions, and addressing multi-label classification[14] .

Ozan _Irsoy (2021) introduced a methodology for adapting the Hierarchical Mixture of Experts (HMoE) model, which organized multiple experts into a tree structure with gating functions at internal nodes and experts at the leaves. A novel dropout technique was applied to internal nodes, dropping entire subtrees to reduce overfitting and produce smoother outputs. The model was trained using a binary tree with stochastic gradient descent (SGD) and the Adam optimizer,

employing cross-entropy loss for classification and squared loss for regression. Evaluations on datasets such as synthetic regression data, MNIST, CIFAR-10, and SSTB demonstrated that dropout reduced overfitting, improved generalization, and outperformed L1 and L2 regularization for deep trees. Qualitative analysis highlighted the diverse representations and ensemble effects created by dropout, showcasing its ability to balance bias and variance and enhance model performance[15].

Albert Senen (2023) analyzed the convergence of dropout-based stochastic training in neural networks, provided theoretical guarantees for stationary points, and highlighted the influence of dropout probability on convergence rates. It showed that dropout mitigated overfitting, with less severe impacts on wide networks compared to deep ones[16].

Joshua Shunk (2022)introduced The NSDropout methodology had involved splitting the dataset into training, validation, unseen validation, and testing subsets. Training had been used to update weights, while validation had identified overfitting. A portion of the training data had been reserved for unseen validation to refine dropout criteria. Using a standard feed-forward architecture, NSDropout had selectively removed noisy neurons based on activation deviations calculated from training and validation data. Dropout masks had been iteratively refined, and backpropagation had updated gradients only for retained neurons. Techniques like SGD, momentum, and L2 weight decay had stabilized training, with dropout proportions tuned per layer (0.5–0.7 for input, 0.2–0.4 for hidden layers). Validation had dynamically influenced neuron retention, ensuring unbiased optimization. Evaluation on MNIST, Fashion-MNIST, and CIFAR-10 had shown that NSDropout had enhanced generalization, reduced reliance on large datasets, and achieved high accuracy, particularly with small datasets. Iterative mask refinement and hyperparameter tuning had been key to its robustness[17] .

Aakash Ravindra Shinde(2024)proposed a methodology for mitigating overfitting in quantum convolutional neural networks (QCNNs). Datasets such as Medical MNIST, BraTS, and Stellar were prepared by splitting them into training, testing, and validation subsets, with dimensionality reduction techniques like principal component analysis (PCA) applied. QCNNs were designed with data encoding, convolutional layers, and pooling layers, which were optimized during

training. Two post-training methods were tested: gate dropout, which proved ineffective due to accuracy losses, and Post-Training Parameter Adjustment (PTA), which effectively reduced overfitting by adjusting variational parameters. Validation accuracy and the testing-validation accuracy gap demonstrated that PTA successfully enhanced model performance and generalization. [18].

Ho-Chan Kim(2020) focused on reducing overfitting in neural networks using the MNIST dataset, tested multi-layer neural networks and convolutional neural networks (CNNs), including simple and deep CNNs. Techniques such as regularization, dropout, and data augmentation were applied to mitigate overfitting. Experiments evaluated the effects of weight decay, dropout, and training dataset size, using test and validation accuracy to measure generalization. Results showed that deep CNNs achieved the highest accuracy and were the most effective at reducing overfitting. Increasing the training dataset size had a greater impact on mitigating overfitting compared to regularization or dropout, highlighting the importance of data quantity in improving generalization[19].

Afshin GHOLAMYa (2023) explored the use of deep learning for geoscience problems, including solving inverse problems and predicting events like earthquakes and volcanic eruptions. Dropout training was applied to improve training speed and reduce overfitting by training sub-networks on different data portions and averaging their results. The geometric mean was identified as the optimal averaging method, outperforming the arithmetic mean. A theoretical analysis explained its success based on criteria like commutativity and continuity. Experiments validated the geometric mean's effectiveness, showing superior generalization and prediction accuracy. The study concluded that the geometric mean was the best choice for dropout training, enhancing deep learning applications in geosciences[20] .

MinghuiLiu(2022) FocusedDropout was a regularization method designed to enhance classification performance and prevent overfitting in CNNs. It was tested on CIFAR10, CIFAR100, and Tiny ImageNet across architectures like ResNet, DenseNet, and VGGNet. The method identified target-related features by selecting the reference channel with the highest average activation value and retaining relevant units using a binary mask. To prevent overfitting, FocusedDropout was applied to 10% of training batches per epoch with increased weight decay. The training incorporated data augmentation and hyperparameter optimization. Evaluation

demonstrated improved generalization and accuracy, outperforming methods like Dropout and SpatialDropout. Class Activation Mapping (CAM) confirmed its effectiveness in enhancing focus on target regions[4] .

Nebojsa Bacanin(2020) proposed a hybridized enhanced bat algorithm (BA-OM) to address overfitting in CNNs by automating and optimizing dropout probability selection. Combining the bat algorithm (BA) with artificial bee colony (ABC) mechanisms, BA-OM balanced global search and local optimization to refine dropout rates. Experiments on MNIST and CIFAR-10 datasets demonstrated the method's effectiveness, achieving test accuracies of 71.76% on CIFAR-10 with a dropout rate of 0.671 and 99.19% on MNIST with a dropout rate of 0.5216. BA-OM outperformed other metaheuristic techniques, improved generalization, and streamlined dropout rate optimization. Future work was planned to explore broader applications and additional datasets[21] .

Yuanyuan Chen(2021)developed The Adaptive Sparse Dropout (AS-Dropout) method was developed to address overfitting in deep neural networks (DNNs) by combining dropout with sparsity, ensuring only 2–5% of neurons remained active during training. Unlike traditional dropout, AS-Dropout adaptively calculated activation probabilities based on neuron activation values, prioritizing those with higher activation. The method involved normalizing activation values, applying a sigmoid function to compute probabilities, and dynamically selecting active neurons. It was tested on MNIST, COIL-100, and Caltech-101 using fully connected networks, VGGNet19, and ResNet50. AS-Dropout consistently outperformed traditional dropout, standout, and sparseout methods, achieving superior generalization and effectively mitigating overfitting, particularly in fully connected networks. However, its performance in convolutional neural networks was less satisfactory, requiring further refinement for broader applications. The method showed significant potential, especially for small datasets[22].

Claudio Filipi (2021)introduced MaxDropout, a novel regularization method that enhanced generalization in deep neural networks by deactivating the most active neurons in hidden layers, allowing less active neurons to learn more informative features. It used L2 normalization to scale activation values and selected neurons for retention based on a randomly chosen dropout rate. MaxDropout was applied by replacing standard Dropout layers or combining it with regularizers like Cutout. Experiments on CIFAR-10 and CIFAR-100 datasets using ResNet18 and

WideResNetarchitectures showed that MaxDropout outperformed standard Dropout, achieving better generalization and reduced overfitting, with further improvements when combined with Cutout. The method demonstrated strong potential for image classification tasks, with plans for broader applications and implementation in additional frameworks[23].

SanghunLee(2020) revisited spatial dropout to address the limitations of conventional element-wise dropout in convolutional layers, which failed due to correlations from weight sharing and local connectivity. Spatial dropout, which dropped entire feature maps, effectively mitigated neuron co-adaptation while maintaining structural integrity. Experiments on CIFAR-10 and CIFAR-100 datasets with VGG, Wide-ResNet, and DenseNet-BC architectures showed significant improvements, particularly with DenseNet-BC achieving a 3.32% error rate on CIFAR-10 using only 3 million parameters. Training employed SGD with Nesterov momentum over 300–600 epochs. Results demonstrated spatial dropout's superior regularization and efficiency, with future work aimed at mathematical enhancements and integration with pre-training[24] .

Zhuang Liu(2022) investigated dropout's potential to address underfitting and overfitting by introducing two techniques: early dropout, which was applied during initial training to improve data fitting in underfitting models, and late dropout, which was introduced later to enhance generalization in overfitting models. Experiments with Vision Transformers (ViT), Mixer, and ConvNeXt architectures on ImageNet-1K, COCO, and ADE20K datasets utilized the AdamW optimizer, cosine learning rate schedules, and dropout rates between 0.1 and 0.7. Metrics such as top-1 accuracy, training loss reduction, and gradient variance alignment highlighted the effectiveness of these techniques. Early dropout consistently reduced training loss and improved test accuracy in underfitting scenarios, while late dropout improved generalization in overfitting cases. Models trained with these methods outperformed baselines in object detection, segmentation, and classification tasks, showcasing dropout's potential for large-scale neural network training and encouraging further research on scheduling strategies[25] .

Ioannis E. Livieris (2021) developed a novel Dropout Weight-Constrained Recurrent Neural Network (DWCRNN) to address the challenge of forecasting cryptocurrency prices, which were highly nonlinear and fluctuating. By combining weight-constrained RNNs with dropout techniques, the model effectively reduced overfitting and improved generalization. It utilized

daily price data for BTC, ETH, XRP, LTC, and the CCi30 index from January 2017 to June 2019, with 27 months for training and 3 months for testing, and applied data transformations for stationarity. The model was evaluated using MAE and RMSE across forecasting horizons of 7, 14, and 21 days and outperformed state-of-the-art algorithms like LSTM, BiLSTM, CNN, and regression models. Optimal dropout rates (10%-20%) were identified, and statistical tests validated its superior accuracy. While the model showed promise, limitations included unclear interactions between weight constraints and dropout techniques and the need for dynamic re-training to adapt to evolving data. Future research was proposed to focus on real-world trading system applications to assess profitability[12] .

BihiSabiri(2022) proposed two techniques, early stopping and dropout, to address overfitting in deep learning models. Early stopping prevented overfitting by halting training when validation loss stopped improving, using strategies such as predefined epochs and validation monitoring. Dropout enhanced generalization by randomly deactivating neurons during training, encouraging the network to learn diverse features. Experiments on datasets like MNIST, SONAR, and Diabetes demonstrated that dropout improved accuracy and reduced loss across datasets, with accuracy on the Diabetes dataset increasing from 75% to 81% and loss decreasing from 81% to 45%. Early stopping further minimized overfitting by stopping training at optimal points, improving generalization while reducing training time. Both techniques effectively improved model robustness, with future research aimed at dynamic dropout adjustments, network compression, and sparse representations to reduce complexity[13] .

Kishan K C (2021) introduced a Bayesian Joint Inference Framework to tackle overfitting and uncertainty calibration in neural networks by jointly inferring network depth and applying dropout regularization. The framework used a beta process to model network depth, enabling theoretically infinite hidden layers, and a conjugate Bernoulli process to control neuron activations. Structured Stochastic Variational Inference (SSVI) efficiently approximated the marginal likelihood for optimization. Experiments on synthetic, UCI, and image datasets, including MNIST and CIFAR-10, demonstrated the framework's superiority in accuracy, uncertainty calibration, and overfitting prevention, outperforming state-of-the-art methods like vanilla dropout and stochastic depth. It dynamically adjusted network depth and neuron activations to create compact, efficient structures and proved effective in continual learning

tasks, reducing catastrophic forgetting on rotated and permuted MNIST datasets. The framework achieved well-calibrated predictions, with future work aimed at enhancing dynamic adaptation and scalability for real-world applications[26].

Xiaobo Liang (2021)proposed R-Drop, a consistency training strategy to address inconsistencies caused by dropout between training and inference stages. R-Drop minimized the bidirectional Kullback-Leibler (KL) divergence between the output distributions of two sub-models generated via dropout during training, adding this as a regularization term to the negative log-likelihood loss. Experiments across 18 datasets and five tasks, including translation (e.g., WMT14), summarization (e.g., CNN/DailyMail), and classification (e.g., CIFAR-100, ImageNet), demonstrated its effectiveness. Tested on models like Transformer, BART, RoBERTa-large, and Vision Transformer (ViT), R-Drop achieved state-of-the-art results, such as BLEU scores of 30.91 and 43.95 on WMT14 English→German and English→French, respectively, and improved model generalization and robustness against overfitting. While computational cost increased due to dual forward passes, future work aimed to explore pre-training stages, apply R-Drop to other architectures like CNNs, and optimize efficiency[27] .

B. H. Pansambal (2023) focused on addressing overfitting in neural networks by applying dropout regularization at different layers, including visible and hidden layers, to improve performance and generalization. Dropout worked by randomly deactivating a percentage of neurons during training, reducing co-adaptation and enabling better generalization. The approach iteratively formed "thinned networks" by retaining subsets of neurons based on dropout rates (e.g., 20%, 30%, 40%, 50%) and aggregating their outputs. Experiments showed that a dropout rate of 0.30 achieved the highest accuracy (87.93%) for visible layers, while 0.40 yielded 85.17% for hidden layers. While dropout effectively reduced overfitting and enhanced accuracy, it increased computational costs due to iterative training. Future work aimed to integrate dropout with other regularization techniques, such as L2 regularization, to improve efficiency and reduce computational overhead[5] .

KarshievSanjar (2021) introduced **Weight Dropout**, a regularization technique for deep neural networks that deactivated weight connections instead of entire neurons to reduce overfitting and improve generalization. It was implemented in convolutional neural networks (CNNs) using a pre-trained ResNet-50 model and employed a dynamic Bernoulli-based dropout mask that was

updated at each training step. Batch gradient descent (BGD) was used for optimization, processing features through batch normalization, activation functions, and weight dropout layers. Evaluations on MNIST, CIFAR-10, and skin lesion segmentation datasets showed superior performance, with validation accuracy of 94.51% (MNIST), 91.68% (CIFAR-10), and leading segmentation metrics (pixel accuracy: 96.27%, Dice coefficient: 88.59%, IoU: 51.48%). Despite its effectiveness, the approach increased computational complexity and risked losing critical information. Future work aimed to enhance scalability and efficiency by refining the dropout process with more informed selection criteria[28].

PanissaraThanapol (2021) addressed overfitting and poor generalization in CNNs trained with limited data by exploring combinations of data augmentation, dropout, and batch normalization techniques. Using a contracted CIFAR-10 dataset with only 10% of the original samples, augmentation methods such as rotation, width and height shifts, shear transformations, and random erasing were tested individually and in combination. Dropout (omission rates: 0.5–0.8) was applied with augmentation, while batch normalization was used to stabilize activations, though it was not combined with dropout due to adverse effects. Injecting augmentation during training, particularly at 30 epochs, yielded the best results. The highest test accuracy (61.5%) was achieved by combining width and height shift augmentation with dropout injected at 30 epochs. Random erasing showed poor performance, while batch normalization with augmentation provided marginal gains. Limitations included dataset specificity and increased computational costs, with future work aimed at applying these strategies to other datasets and optimizing CNNs for limited data scenarios[29] .

Table 1-Comparison table

author	Dataset	Advantage	Disadvantage	Limitation	Algorithm	Result
Kaiyu Zhang	- CWRU dataset: For small sample and cross-load diagnosis. - SQ dataset: For variable speeds and fault severities. - Escalator	- Mitigates overfitting while preserving feature richness. - Enhances generalization with shallow architecture and dropout.	- Requires time-consuming hyperparameter optimization.	- Limited generalization under dynamic operational conditions. - Does not address multi-label classification or unbalanced datasets.	- Compact CNN with multiscale feature extraction. - 1D convolutional kernels for multiscale analysis. - Gradient descent with	- Demonstrated high accuracy and strong generalization across datasets. - Outperformed traditional CNNs and SVM-based

	dataset: For real-world industrial applications.	- Achieves high accuracy with limited samples and varying speeds.			exponentially decreasing learning rate.	methods.
Ozan _Irsoy	Synthetic regression data, MNIST, CIFAR-10, SSTB	- Reduces overfitting by distributing decision-making across the tree. - Produces smoother outputs. - Improves generalization across datasets. - Outperforms L1/L2 regularization, especially for deep trees.	- Adapting dropout to hierarchical structures may add complexity compared to flat architectures.	- Effectiveness depends on selecting appropriate dropout rates. - Results may vary for deep trees and different datasets.	Hierarchical Mixture of Experts (HMoE) with custom dropout applied at internal gating nodes. - Training: SGD with Adam optimizer. - Loss functions: cross-entropy (classification), squared loss (regression).	- Dropout effectively reduces overfitting (smaller training-validation error gaps). - Higher dropout rates yield smoother outputs. - Improves generalization across tasks and datasets.
Albert Senen	Wide and deep neural networks, arborescence-shaped networks.	Reduces overfitting; ensures convergence to stationary points; effective in wide networks.	Convergence impaired by exponential dependence on depth.	Dropout probability significantly affects convergence; bounds may not fully represent real-world behaviors.	SGD with dropout, compact projections, explicit risk function for arborescence networks.	Guaranteed convergence to stationary points; slower in deep networks; better performance in wide networks with fewer dropout layers.
Joshua Shunk	MNIST, Fashion-MNIST, CIFAR-10; Dataset split into training, validation, unseen validation, and testing subsets. Training data further divided for unseen	Improves generalization and reduces overfitting by selectively dropping noisy neurons; reduces dependence on large training	Increased training time compared to standard neural networks and traditional dropout methods; requires more computational effort due to iterative refinement of	Training takes up to four times longer than standard networks; additional hyperparameter (p) adds complexity; sorting operations for mask creation consume significant computational	1. Split dataset into subsets. 2. Calculate neuron activation averages for each class. 3. Identify and drop noisy neurons using dropout masks. 4. Perform forward	Superior generalization and accuracy across datasets; achieved near-perfect accuracy with as few as 750 MNIST examples; significantly lower error

	validation.	datasets; achieves superior accuracy with small datasets; dynamically updates dropout masks.	dropout masks.	resources.	propagation with thinned layers. 5. Perform backpropagation using SGD. 6. Refine dropout masks iteratively. 7. Evaluate on unseen validation and testing datasets.	rates compared to traditional dropout and other regularization techniques.
Aakash Ravindra Shinde,	Medical MNIST, BraTS, Stellar datasets split into training, testing, and validation subsets. Dimensionality reduction (e.g., PCA) was applied to adapt data to circuit capacity.	Successfully reduced overfitting with PTA, improved validation accuracy, enhanced generalization across datasets, and preserved quantum circuit integrity.	Gate dropout method was ineffective, causing significant accuracy drops. PTA requires careful parameter selection and adjustment.	Gate dropout highlights QCNN vulnerability; PTA is dependent on manual or empirical determination of adjustment thresholds.	1. Prepare datasets and apply dimensionality reduction. 2. Train QCNN with data encoding, convolutional, and pooling layers. 3. Use PTA to adjust selected parameters. 4. Validate and test performance using accuracy metrics.	PTA reduced overfitting, improved validation accuracy, and increased generalization across datasets, with consistent success in mitigating overfitting.
Ho-Chan Kim	MNIST dataset with 60,000 training images and 10,000 test images, each depicting a 28x28 grayscale digit. Data augmentation techniques like flipping and rotation were used.	Deep CNNs consistently achieved the highest accuracy and better generalization. Increasing training data size significantly reduced overfitting compared to regularization or dropout.	Techniques like regularization and dropout were less effective than increasing training data size or using advanced architectures like deep CNNs.	The study focused solely on the MNIST dataset, limiting generalization to other datasets or tasks. Regularization and dropout were not as impactful in reducing overfitting.	1. Prepare datasets with augmentation. 2. Test multi-layer neural networks and CNNs (simple and deep). 3. Apply regularization, dropout, and data augmentation. 4. Measure performance	Deep CNNs outperformed other models, achieving the highest accuracy. Increasing dataset size was more effective in reducing overfitting than regularization or dropout.

					using test and validation accuracy.	
Afshin GHOLAMYa,	Historical geoscience data, including seismic activity and volcanic eruption patterns, used for inverse problem-solving and prediction tasks.	The geometric mean improved training efficiency, reduced overfitting, and enhanced generalization and prediction accuracy in dropout training.	The study mainly focused on theoretical and experimental validation of the geometric mean, potentially overlooking other promising combination methods.	Limited to geoscience applications, the findings may not generalize across other fields or datasets.	1. Apply deep learning models with dropout training. 2. Train sub-networks on different data portions. 3. Combine results using the geometric mean. 4. Evaluate performance through prediction accuracy and generalization metrics.	The geometric mean outperformed other methods like the arithmetic mean, providing better generalization and prediction accuracy for dropout-related deep learning tasks.
Minghui Liu	CIFAR10, CIFAR100, and Tiny ImageNet datasets used for classification tasks.	Improved classification performance and generalization; effectively focuses on target features while preventing overfitting.	Requires careful tuning of hyperparameters, such as dropout participation rate and weight decay, for optimal performance.	FocusedDropout relies on spatial invariance and may not generalize well to non-image data or tasks where spatial alignment is less critical.	1. Select the reference channel with the highest average activation value. 2. Identify target-related units using spatial invariance. 3. Generate a binary mask to retain target units. 4. Apply FocusedDropout to 10% of batches with increased weight decay. 5. Train models with	Achieved improved accuracy and generalization across CNN architectures; outperformed baseline methods like Dropout, SpatialDropout, and DropBlock; enhanced target focus, as verified by Class Activation Mapping (CAM).

					data augmentation and hyperparameter optimization.	
Nebojsa Bacanin	MNIST and CIFAR-10 datasets. MNIST: Two convolutional layers, two pooling layers, one fully connected layer. CIFAR-10: Three convolutional layers, three pooling layers, two fully connected layers. Datasets were divided into training, validation, and testing subsets.	Automated and optimized dropout probability selection; balanced exploration and exploitation using hybridized bat algorithm (BA-OM); achieved superior test accuracy and generalization compared to other methods.	Requires a complex combination of metaheuristic techniques, which increases computational complexity.	Results are limited to MNIST and CIFAR-10 datasets; broader generalization to other datasets and tasks is yet to be tested.	1. Use a hybridized enhanced bat algorithm (BA-OM). 2. Alternate between BA's global search and ABC's local optimization. 3. Evaluate dropout probabilities based on test accuracy. 4. Optimize dropout rates iteratively.	Achieved test accuracy of 71.76% on CIFAR-10 with a dropout rate of 0.671 and 99.19% on MNIST with a dropout rate of 0.5216. Outperformed other metaheuristic techniques like PSO, FA, and CS in optimizing dropout rates.
Yuanyuan Chen	MNIST, COIL-100, and Caltech-101 datasets. Neural network architectures included fully connected networks, VGGNet19, and ResNet50.	Effectively mitigates overfitting by combining dropout with sparsity; adaptively calculates activation probabilities, improving generalization; particularly effective for small datasets.	Requires careful tuning of the shift parameter to control neuron activation probabilities.	Less effective in convolutional neural networks compared to fully connected networks; further refinement is needed for broader applicability.	1. Normalize activation values to a defined range. 2. Compute activation probabilities using a sigmoid function. 3. Retain 2–5% of neurons dynamically based on calculated probabilities. 4. Evaluate	AS-Dropout outperformed traditional dropout, standout, and sparseout methods, achieving better generalization and effectively preventing overfitting, particularly in fully connected networks.

					performance using recognition errors and training-testing error analysis.	
Claudio Filipi	CIFAR-10 and CIFAR-100 datasets, comprising 60,000 images with varying class distributions. Evaluated on ResNet18 and WideResNet architectures.	Encourages less active neurons to learn more informative features, improving generalization and reducing overfitting; outperforms standard Dropout and enhances performance when combined with Cutout.	Requires careful tuning of the dropout rate threshold; additional computational complexity due to L2 normalization and selection of neurons to deactivate.	Limited to image classification tasks; broader applications like object detection and speech recognition need further testing.	1. Apply L2 normalization to neuron activation values. 2. Identify the most active neurons. 3. Retain neurons below a threshold determined by a random dropout rate. 4. Multiply the activation tensor by a generated mask to deactivate selected neurons. 5. Incorporate MaxDropout into training with data augmentation methods.	MaxDropout outperformed standard Dropout on CIFAR-10 and CIFAR-100 datasets, achieved better generalization, and further improved performance when combined with Cutout.
Sanghun Lee	CIFAR-10 and CIFAR-100 datasets (60,000 32×32 color images)	Effectively mitigates neuron co-adaptation while maintaining structural integrity of convolutional layers. Improves regularization, reduces error rates, and increases model efficiency. Outperforms standard	No explicit disadvantages mentioned; possibly increased training time.	Evaluated only on CIFAR-10 and CIFAR-100 datasets and specific architectures. Further mathematical analysis and testing with pre-training are needed.	Spatial dropout replaces standard dropout, dropping entire feature maps in CNNs. Training used Stochastic Gradient Descent (SGD) with Nesterov momentum (0.9) and weight decay (0.0001).	Achieved a 3.32% error rate on CIFAR-10 with DenseNet-BC using only 3 million parameters. Demonstrated superior regularization and performance compared to standard dropout.

		dropout, achieving significant error reduction.				
Zhuang Liu	ImageNet-1K (initial evaluations), COCO, ADE20K (downstream tasks).	Early dropout reduces underfitting by improving data fitting and lowering training loss. Late dropout mitigates overfitting and enhances generalization accuracy in large models.	Additional hyperparameter tuning required for dropout rates and activation epochs.	Effectiveness depends on specific model architectures and dataset characteristics. Early dropout may not help models that are already overfitting.	Early dropout: Applied during initial training and deactivated later. Late dropout: Introduced in later training phases. Both utilize dropout with variable rates (0.1 to 0.7).	Early dropout reduced training loss and improved test accuracy in underfitting regimes. Late dropout enhanced generalization accuracy for overfitting models, outperforming baselines.
Ioannis E. Livieris	Daily price data for BTC, ETH, XRP, LTC, and CCi30 index from January 2017 to June 2019. Training data covered 27 months, and testing data spanned 3 months.	DWCRNN significantly reduced overfitting and improved generalization. It captured reliable patterns while filtering out noise, outperforming other models in forecasting accuracy.	Requires extensive tuning of dropout rates and hyperparameters. The interaction between weight constraints and dropout techniques remains unclear.	Effectiveness in real-world trading systems was not tested. The model's adaptability to dynamic market changes and new data needs further research.	Dropout Weight-Constrained Recurrent Neural Network (DWCRNN), combining weight constraints and dropout techniques applied to feed-forward connections to improve forecasting.	DWCRNN outperformed state-of-the-art models (e.g., LSTM, CNN, SVR) with lower MAE and RMSE across all datasets and forecasting horizons, demonstrating superior accuracy and efficiency.
BihiSabiri	MNIST, SONAR, and Diabetes datasets.	Dropout reduced overfitting, improved generalization, and forced the network to learn	Dropout can hinder performance during training as it suppresses neuron activations. Early	Dropout requires careful tuning of probabilities and may not completely eliminate overfitting.	Dropout randomly deactivates neurons during training with specified probabilities	Dropout improved accuracy on the Diabetes dataset from 75% to 81% and reduced loss from

		diverse features. Early stopping minimized overfitting and saved training time.	stopping may halt training prematurely, risking underfitting.	Early stopping depends heavily on validation set monitoring.	(e.g., 0.25 to 0.5). Early stopping halts training when validation loss stops improving.	81% to 45%. On MNIST, accuracy increased from 98% to 99%, and loss decreased from 5% to 3%.
Kishan K C	Synthetic datasets, UCI datasets, and image datasets like MNIST, FashionMNIST, SVHN, and CIFAR-10.	Dynamically adjusts network depth and neuron activations based on data, achieving compact, efficient structures. Enhances accuracy, uncertainty calibration, and robustness to overfitting.	Computational complexity due to infinite hidden layer modeling and the need for approximations like Structured Stochastic Variational Inference (SSVI).	Requires further improvements for real-world scalability and dynamic adaptation to evolving datasets.	Bayesian Joint Inference Framework using a beta process for network depth and conjugate Bernoulli process for neuron activations, optimized via SSVI.	Achieved superior performance in accuracy and uncertainty calibration, dynamically adjusted network depth, and reduced catastrophic forgetting in continual learning tasks.
Xiaobo Liang	18 datasets across 5 tasks: neural machine translation (e.g., WMT14), abstractive summarization (e.g., CNN/DailyMail), language understanding (GLUE benchmark), language modeling (e.g., Wikitext-103), and image classification (e.g., CIFAR-100, ImageNet).	Improved model generalization and reduced overfitting. Achieved state-of-the-art results across multiple datasets. Enhanced consistency between training and inference.	Increased computational cost due to dual forward passes per input sample.	Tested only during fine-tuning; performance during pre-training was not explored.	R-Drop: Minimizes the bidirectional Kullback-Leibler (KL) divergence between output distributions of two sub-models generated via dropout during training.	Achieved state-of-the-art BLEU scores of 30.91 (WMT14 English→German) and 43.95 (WMT14 English→French) and robust improvements across all tasks and datasets.

B. H. Pansambal	Not explicitly specified in the paragraph, but the methodology focuses on neural networks with visible and hidden layers for general use across datasets.	Reduced overfitting, improved generalization, and enhanced accuracy for larger networks by randomly deactivating neurons and creating diverse sub-networks.	Increased computational costs due to iterative sub-network creation during training.	Requires careful selection of dropout rates and batch sizes to achieve optimal performance; computational overhead remains significant for larger networks.	Dropout regularization: Randomly deactivates a subset of neurons during training based on a specified dropout rate, iteratively forming "thinned networks."	Achieved best accuracy of 87.93% for visible layers with a 0.30 dropout rate and 85.17% for hidden layers with a 0.40 dropout rate, effectively reducing overfitting.
KarshievSanjar	MNIST, CIFAR-10, and a skin lesion segmentation dataset.	Reduced overfitting and improved generalization. Achieved higher validation accuracy and superior performance in segmentation tasks compared to baseline models.	Increased computational complexity due to dynamic dropout mask updates.	Risk of losing critical information by deactivating connections, which could impact performance in certain scenarios.	Weight Dropout: Randomly deactivates weight connections using a Bernoulli distribution, dynamically updating the dropout mask during training.	Achieved 94.51% validation accuracy on MNIST, 91.68% on CIFAR-10, and top performance on segmentation tasks (pixel accuracy: 96.27%, Dice coefficient: 88.59%, IoU: 51.48%).
PanissaraThanapol	CIFAR-10 dataset, with five contracted versions created (each containing 10% of the original samples) to simulate limited data scenarios.	Improved generalization and reduced overfitting by combining data augmentation, dropout, and batch normalization techniques.	Increased computational overhead due to complex augmentation and dropout combinations.	Results were dataset-specific and may not generalize to other scenarios with limited data. Batch normalization could not be combined with dropout effectively.	Combined data augmentation (e.g., width and height shifts, rotation, shear), dropout (omission rates: 0.5–0.8), and batch normalization. Injected augmentation during training (best at 30 epochs).	Achieved the highest test accuracy of 61.5% by combining width and height shift augmentation with dropout, injected at 30 epochs during training.

7. discussion

The discussion highlights various advanced regularization techniques to combat overfitting and improve generalization in deep learning models. FocusedDropout demonstrated superior performance by selectively retaining target-related features in CNNs, outperforming traditional methods. Similarly, the DWCRNN effectively forecasted cryptocurrency prices, surpassing state-of-the-art models, though it required further exploration of dynamic re-training and weight constraints. Early stopping and dropout techniques improved robustness by halting training at optimal points and deactivating neurons randomly. Weight Dropout enhanced generalization by targeting weight connections but increased computational complexity. Lastly, data augmentation combined with dropout yielded the best results for limited data scenarios, despite dataset-specific limitations and higher computational costs. These studies underline the effectiveness of tailored approaches while identifying areas for future refinement and broader applications.

8. Conclusion

Dropout has proven to be a highly effective regularization technique for mitigating overfitting and enhancing generalization in neural networks. By introducing stochasticity during training, dropout prevents co-adaptation of neurons, fostering robust and diverse feature learning. Empirical evidence demonstrates its effectiveness across a variety of architectures, including CNNs, RNNs, and Transformers, and on datasets such as CIFAR-10, MNIST, and ImageNet. Advanced dropout variants, such as FocusedDropout, SpatialDropout, and WeightDropout, have further refined its applicability, addressing specific architectural challenges and enhancing performance in targeted scenarios.Despite its widespread adoption, dropout is not without limitations. Challenges such as computational overhead, sensitivity to hyperparameter tuning, and dataset-specific effectiveness persist. Recent advancements, including adaptive dropout rates

and hybrid approaches, show promise in overcoming these challenges and improving scalability. Furthermore, integrating dropout with other regularization methods, such as data augmentation and batch normalization, has yielded notable results, particularly in limited data scenarios.The study of dropout underscores its pivotal role in modern deep learning, both as a standalone method and in combination with other techniques. Future research should focus on refining adaptive mechanisms, exploring broader applications, and enhancing computational efficiency to ensure dropout remains a cornerstone in building robust, generalizable neural networks.

References :

[1] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *J. Mach. Learn. Res.*, vol. 15, pp. 1929–1958, 2014.

[2] H. Salehinejad and S. Valaee, "EDropout: Energy-Based Dropout and Pruning of Deep Neural Networks," *IEEE Trans. Neural Networks Learn. Syst.*, vol. 33, no. 10, pp. 5279–5292, 2022, doi: 10.1109/TNNLS.2021.3069970.

[3] H. Moayed and E. G. Mansoori, "Skipout: An Adaptive Layer-Level Regularization Framework for Deep Neural Networks," *IEEE Access*, vol. 10, pp. 62391–62401, 2022, doi: 10.1109/ACCESS.2022.3178091.

[4] M. Liu *et al.*, "FocusedDropout for Convolutional Neural Network," *Appl. Sci.*, vol. 12, no. 15, pp. 1–14, 2022, doi: 10.3390/app12157682.

[5] B. H. Pansambal and A. B. Nandgaokar, "Integrating Dropout Regularization Technique at Different Layers to Improve the Performance of Neural Networks," *Int. J. Adv. Comput. Sci. Appl.*, vol. 14, no. 4, pp. 716–722, 2023, doi: 10.14569/IJACSA.2023.0140478.

[6] H. M. Huang *et al.*, "Implementation of Dropout Neuronal Units Based on Stochastic Memristive Devices in Neural Networks with High Classification Accuracy," *Adv. Sci.*, vol. 7, no. 18, pp. 1–9, 2020, doi: 10.1002/advs.202001842.

[7] Y. Mae, W. Kumagai, and T. Kanamori, "Uncertainty propagation for dropout-based Bayesian neural networks," *Neural Networks*, vol. 144, pp. 394–406, 2021, doi: 10.1016/j.neunet.2021.09.005.

[8] I. Salehin and D. K. Kang, "A Review on Dropout Regularization Approaches for Deep Neural Networks within the Scholarly Domain," *Electron.*, vol. 12, no. 14, 2023, doi: 10.3390/electronics12143106.

[9] C. H. Martin and M. W. Mahoney, "Implicit self-regularization in deep neural networks: Evidence from random matrix theory and implications for learning," *J. Mach. Learn. Res.*, vol. 22, pp. 1–73, 2021.

[10] G. V Ghiasi Google Brain Tsung-Yi Lin Google Brain Quoc Le Google Brain, "DropBlock," *Adv. Neural Inf. Process. Syst.*, vol. 31, no. NeurIPS, pp. 1–11, 2018, [Online]. Available: http://papers.nips.cc/paper/8271-dropblock-a-regularization-method-for-convolutional-networks.pdf?nsukey=HLn8UaxvJaHPDy8mHpdvjrwWBefCIyz%2BRD%2FBpj79onCYXbfVRTv02IMU7ULMsKY4DHDI4qnEV5nqgYw9efJQ6bUF5uHvysay0JLtqhkEYVxMBIGiAdZWiZccVun3wYnN7IfWbWb37NIhQN7opY

[11] C. F. G. Dos Santos and J. P. Papa, "Avoiding Overfitting: A Survey on Regularization Methods for Convolutional Neural Networks," *ACM Comput. Surv.*, vol. 54, no. 10 s, 2022, doi: 10.1145/3510413.

[12] I. E. Livieris, S. Stavroyiannis, E. Pintelas, T. Kotsilieris, and P. Pintelas, "A dropout weight-constrained recurrent neural network model for forecasting the price of major cryptocurrencies and CCi30 index," *Evol. Syst.*, vol. 13, no. 1, pp. 85–100, 2022, doi: 10.1007/s12530-020-09361-2.

[13] B. Sabiri, B. El Asri, and M. Rhanoui, "Mechanism of Overfitting Avoidance Techniques for Training Deep Neural Networks," *Int. Conf. Enterp. Inf. Syst. ICEIS - Proc.*, vol. 1, no. May, pp. 418–427, 2022, doi: 10.5220/0011114900003179.

[14] K. Zhang, J. Chen, T. Zhang, and Z. Zhou, "A Compact Convolutional Neural Network Augmented with Multiscale Feature Extraction of Acquired Monitoring Data for Mechanical Intelligent Fault Diagnosis," *J. Manuf. Syst.*, vol. 55, no. April, pp. 273–284, 2020, doi: 10.1016/j.jmsy.2020.04.016.

[15] O. İrsoy and E. Alpaydın, "Dropout regularization in hierarchical mixture of experts," *Neurocomputing*, vol. 419, pp. 148–156, 2021, doi: 10.1016/j.neucom.2020.08.052.

[16] A. Senen-Cerda and J. Sanders, "Almost Sure Convergence of Dropout Algorithms for Neural Networks," no. 2012, 2020, [Online]. Available: http://arxiv.org/abs/2002.02247

[17] J. Shunk, "Neuron-Specific Dropout: A Deterministic Regularization Technique to Prevent Neural Networks from Overfitting & Reduce Dependence on Large Training Samples," pp. 1–19, 2022,

[Online]. Available: http://arxiv.org/abs/2201.06938

[18] A. R. Shinde, C. Jain, and A. Kalev, "Soft-Dropout: A Practical Approach for Mitigating Overfitting in Quantum Convolutional Neural Networks," *arXiv Prepr. arXiv2309.01829*, pp. 1–10, 2023, [Online]. Available: https://arxiv.org/abs/2309.01829%0Ahttps://arxiv.org/pdf/2309.01829

[19] H.-C. Kim and M.-J. Kang, "A comparison of methods to reduce overfitting in neural networks," *Int. J. Adv. smart Converg.*, vol. 9, no. 2, pp. 173–178, 2020, [Online]. Available: http://dx.doi.org/10.7236/IJASC.2020.9.2.173

[20] A. Gholamy, J. Parra, V. Kreinovich, O. Fuentes, and E. Anthony, "How to best apply deep neural networks in geosciences: Towards optimal 'averaging' in dropout training," *Unconv. Methods Geosci. Shale Gas Pet. 21st Century*, vol. 0, pp. 11–21, 2023, doi: 10.3233/AERD230004.

[21] N. Bacanin, E. Tuba, T. Bezdan, I. Strumberger, R. Jovanovic, and M. Tuba, "Dropout Probability Estimation in Convolutional Neural Networks by the Enhanced Bat Algorithm," *Proc. Int. Jt. Conf. Neural Networks*, no. 3, 2020, doi: 10.1109/IJCNN48605.2020.9206864.

[22] Y. Chen and Z. Yi, "Adaptive sparse dropout: Learning the certainty and uncertainty in deep neural networks," *Neurocomputing*, vol. 450, pp. 354–361, 2021, doi: 10.1016/j.neucom.2021.04.047.

[23] C. F. G. do Santos, D. Colombo, M. Roder, and J. P. Papa, "MaxDropout: Deep neural network regularization based on maximum output values," *Proc. - Int. Conf. Pattern Recognit.*, pp. 2671–2676, 2020, doi: 10.1109/ICPR48806.2021.9412733.

[24] S. Lee and C. Lee, "Revisiting spatial dropout for regularizing convolutional neural networks," *Multimed. Tools Appl.*, vol. 79, no. 45–46, pp. 34195–34207, 2020, doi: 10.1007/s11042-020-09054-7.

[25] Z. Liu, Z. Xu, J. Jin, Z. Shen, and T. Darrell, "Dropout Reduces Underfitting," *Proc. Mach. Learn. Res.*, vol. 202, pp. 21715–21729, 2023.

[26] K. C. Kishan, R. Li, and M. Gilany, "Joint Inference for Neural Network Depth and Dropout Regularization," *Adv. Neural Inf. Process. Syst.*, vol. 32, no. NeurIPS, pp. 26622–26634, 2021.

[27] X. Liang *et al.*, "R-Drop: Regularized Dropout for Neural Networks," *Adv. Neural Inf. Process. Syst.*, vol. 13, no. NeurIPS, pp. 10890–10905, 2021.

[28] K. Sanjar, A. Rehman, A. Paul, and K. Jeonghong, "Weight dropout for preventing neural networks

from overfitting," *2020 8th Int. Conf. Orange Technol. ICOT 2020*, pp. 10–13, 2020, doi: 10.1109/ICOT51877.2020.9468799.

[29] P. Thanapol, K. Lavangnananda, P. Bouvry, F. Pinel, and F. Leprevost, "Reducing Overfitting and Improving Generalization in Training Convolutional Neural Network (CNN) under Limited Sample Sizes in Image Recognition," *InCIT 2020 - 5th Int. Conf. Inf. Technol.*, pp. 300–305, 2020, doi: 10.1109/InCIT50588.2020.9310787.