COMP 551 ASSIGNMENT 4

Antony Zhao, Naiwrita Sen, Shichen Li

December 7, 2023

Abstract

We investigated dropout's impact on vision tasks, paralleling Srivastava et al.'s study in 'Dropout: A Simple Way to Prevent Neural Networks from Overfitting'[5]. Without the original code, we created our own version and emphasized hyperparameter tuning. Our findings, using MNIST, SVHN, CIFAR-10, and CIFAR-100 datasets, reaffirm dropout's effectiveness in reducing overfitting in neural networks.

1 Introduction

The paper "Dropout: A Simple Way to Prevent Neural Networks from Overfitting" is a pivotal work in Machine Learning and Artificial Intelligence, introducing 'dropout' to combat overfitting in neural networks. Overfitting, where models excel on training data but falter on new data, limits practical applications. Dropout, a method of randomly omitting neurons during training, promotes robust learning and better generalization to new data, marking a significant departure from previous, more complex anti-overfitting techniques.

Our project aims to delve into the dropout method outlined and reproduce the results in the paper to explore the practical implementation and effectiveness of the method. For our investigation, we chose a subset of the datasets used in the paper: MNIST, SVHN, CIFAR10, and CIFAR100. For our analysis, we developed our own implementation and conducted several experiments with different models by tuning hyperparameters such as the dropout rate and varying the architecture of the neural network. Our results showed that dropout enhanced the accuracy across all four datasets and helped prevent overfitting at large epochs.

2 Datasets

2.1 MNIST

The MNIST dataset ([1]), a cornerstone in machine learning, specifically image processing, consists of 70,000 grayscale images (28x28 pixels) of digits 0-9. It is divided into a training set of 60,000 images and a test set of 10,000. Notable for its normalization and alignment, MNIST is crucial for advancing neural networks and deep learning techniques, contributing to seminal works like "Batch Normalization" [3] and "Deep Residual Learning" [2].

2.2 SVHN (Street View House Numbers)

The Street View House Numbers (SVHN) dataset is a collection of over 600,000 color images of house numbers from Google Street View, used for machine learning in automatic digit recognition [4]. Unlike the simpler MNIST dataset, SVHN offers real-world complexity with varied digit colors, sizes, orientations, and backgrounds. It provides two formats: original images with character-level bounding boxes, and 32x32 pixel centered, cropped digit images. This diversity challenges image recognition systems with conditions more representative of real-world scenarios.

2.3 CIFAR-10 and CIFAR-100

The CIFAR-10 and CIFAR-100 datasets, detailed in Krizhevsky (2009), are essential for image classification and object recognition. CIFAR-10 contains 60,000 images in ten categories (like 'plane', 'car', 'bird'), split into 50,000 training and 10,000 test images. Each category is distinct. CIFAR-100, similar in size to CIFAR-10,

has 100 classes with 500 training and 100 test images per class, offering a more complex challenge due to its broader range of objects such as flowers and vehicles.

3 Dropout Algorithm Implementation

We implemented the dropout algorithm by creating a class CustomDropout which can be instantiated as a PyTorch nn module. This class creates a mask with the same dimension as the input. Its elements are drawn from a Bernoulli distribution whose possibility argument can be set by passing in a p argument when creating the instance of this class. Then, the class will return the element-wise product between the input and this mask, which will set a portion of the input to zero according to the dropout factor. We tested this algorithm on a fully connected network which is introduced in the dropout paper, this model has two hidden layers with 800 neurons each. The result on the MNIST dataset is as shown in Figure 1, and a zoomed in view on the last 10 epochs is as shown in Figure 2. The test set accuracy is as shown in Table 1.

Method	Test Accuracy
CNN without dropout	97.21 %
CNN with dropout	**97.82 %**

Table 1: Results of Our Dropout Implementation on MNIST Dataset

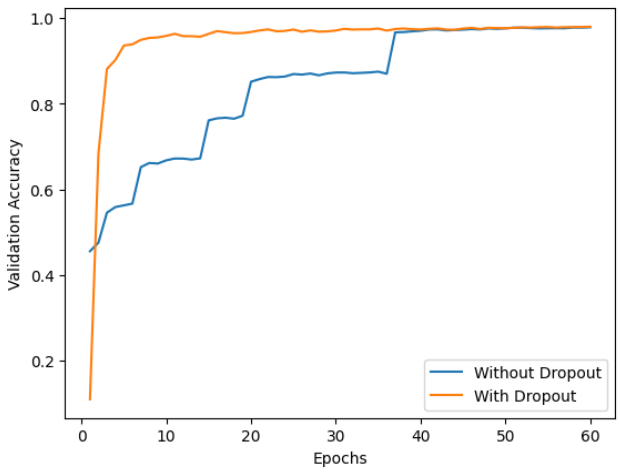

Figure 1: Validation set accuracy comparison.

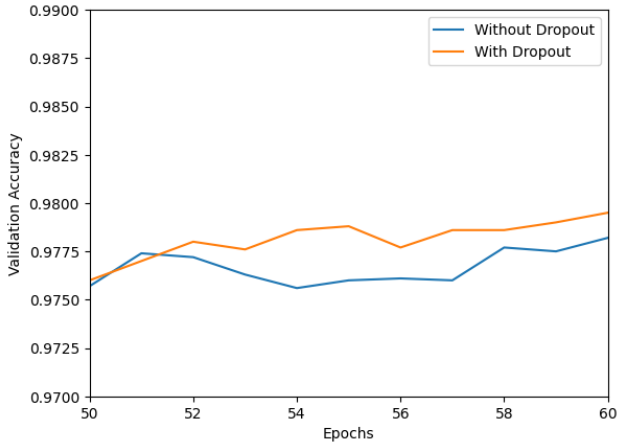

Figure 2: Zoomed in on last 10 epochs.

4 Impact of Dropout on Different Datasets

- **4.1 Experimental Results on MNIST**

  Our first experiment was to test the dropout method against a fully connected network with 2 hidden layers, each with 800 neurons. We implemented the network with and without dropout. As shown in Table 2, our findings revealed that introducing dropout increased the accuracy achieved on test set. With dropout, the test accuracy was **98.26%** compared to an accuracy of **97.73%** obtained without the dropout method. The validation set accuracy versus epochs is as shown in Figure 3, also, a zoomed in version on the 40th to 50th epochs is shown in Figure 4

Method	Test Accuracy
Neural Net Without Dropout	97.73 %
Neural Net With Dropout	**98.26 %**

Table 2: Accuracy of Different Models on MNIST

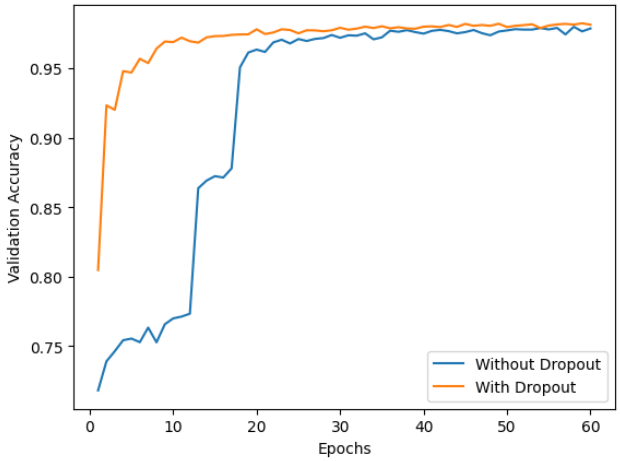

Figure 3: MNIST validation set accuracy

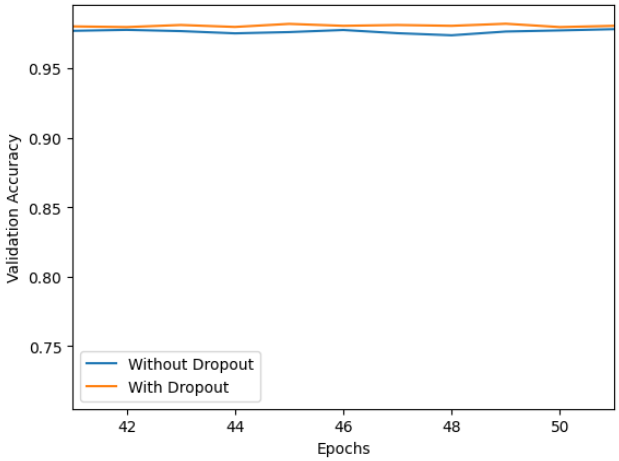

Figure 4: Zoomed on the 40 to 50 epochs

- **4.2 Experimental Results on SVHN**

We implemented a convolutional neural network with three convolutional layers, two fully connected hidden layers, and ReLU activation functions, each followed by a max-pooling layer. Our results (Figure 5) show that the dropout model achieved **93.10%** accuracy, slightly higher than the **92.15%** of the non-dropout model. A detailed view of the last 10 epochs is in Figure 6. The minimal accuracy difference is likely due to the complexity of the SVHN model, which may require more epochs or a complex design to significantly demonstrate overfitting. Test set accuracy details are in Table 3.

Method	Test Accuracy
CNN without dropout	92.15 %
CNN with dropout	**93.10 %**

Table 3: Accuracy of Different Models on SVHN

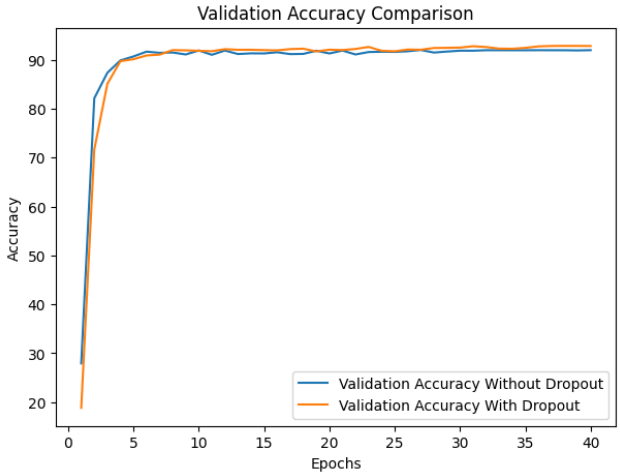

Figure 5: SVHN val set accuracy

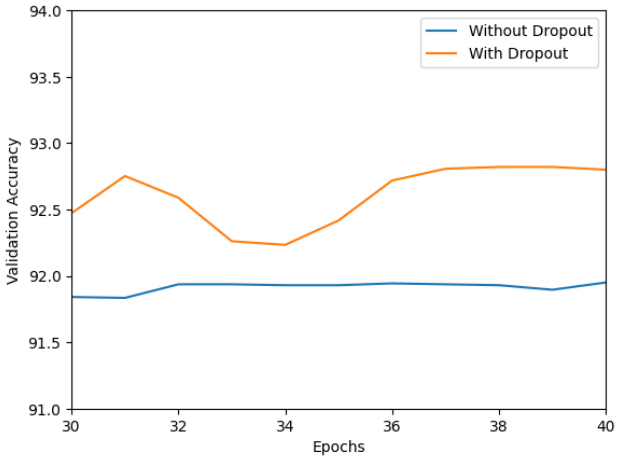

Figure 6: Zoomed on the 40 to 50 epochs

- **4.3 Experimental Results on CIFAR-10**

For the CIFAR-10 dataset, we employed a CNN network with a custom structure shown in Figure 7. The validation set accuracy of the model with and without dropout is as shown in Figure 8. As can be seen, after certain amount of epochs has been reached, the models starts to show overfitting behavior. In this region, the model with dropout achieved a higher accuracy than the original model. Furthermore, the accuracy on test set is as shown in Table 4.

Method	Test Accuracy
CNN without dropout	80.82 %
CNN with dropout	**81.23 %**

Table 4: Results of Dropout on CIFAR-10 Dataset

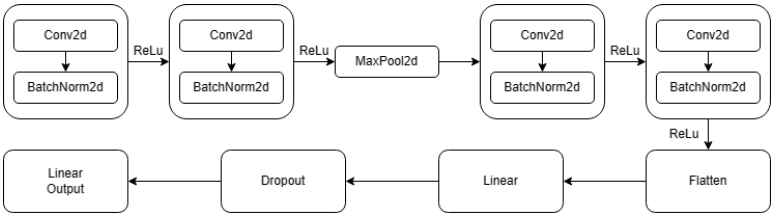

Figure 7: Used Model Structure on CIFAR-10

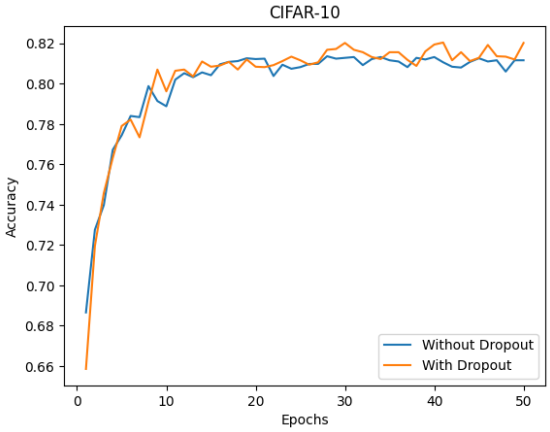

Figure 8: Val Set Accuracy CIFAR-10

- **4.4 Experimental Results on CIFAR-100**

  For the CIFAR-100 dataset, we implemented a CNN network according to the structure [5] provided by the author of the dropout paper Figure 9. The validation set accuracy is as shown in Figure 10

  Once again, we can see in Figure 10 that the model incorporating dropout method outperforms the CNN model without dropout. When the accuracy starts to become constant and overfitting begins, the model with dropout maintains a better accuracy. The test set accuracy as shown in Table 5

Method	Test Accuracy
CNN without dropout	41.41 %
CNN with dropout	**43.13 %**

Table 5: Effect of Adding Regularization on Test Accuracy

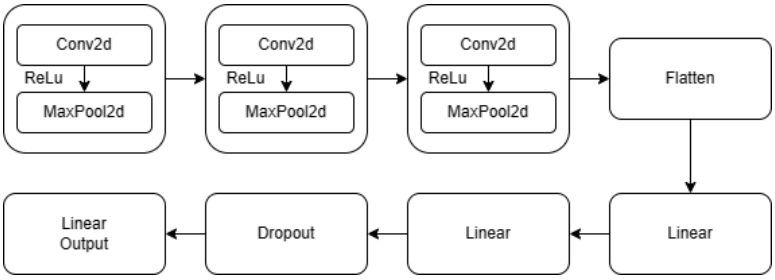

Figure 9: Used Model Structure on CIFAR-100

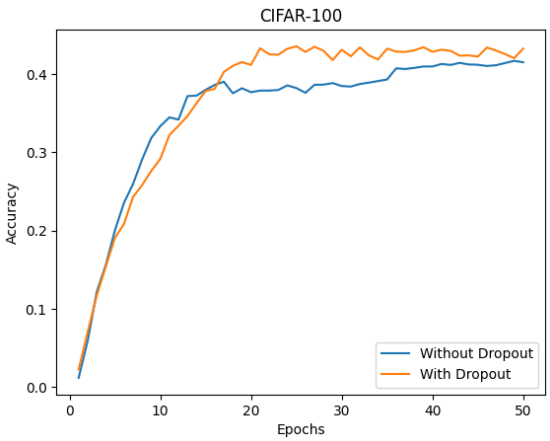

Figure 10: Val Set Accuracy CIFAR-100

5 Hyperparameters

In this section, we conducted experiments on MNIST and SVHN datasets by varying the dropout rate to study their performance.

5.1 Changing of dropout rates for MNIST dataset

In this section, we conducted experiments on MNIST and SVHN datasets by varying the dropout rate to study their performance. Our experiment on the MNIST dataset shows that a dropout rate of 0.5 achieves the highest accuracy of 98.26 %. A dropout rate of 0.9 that is too high performs very poorly on the dataset and reduces the test accuracy to 11.35 %. The accuracy on test set is shown in Table 6. The overall result is shown in Figure 11 and Figure 12 shows the result zoomed in on the last 10 epochs.

Dropout Rate	Test Accuracy
0.1	97.94 %
0.3	98.09 %
0.5	**98.26 %**
0.7	97.09 %
0.9	11.35 %

Table 6: Effect of Varying Dropout Rates on MNIST Dataset

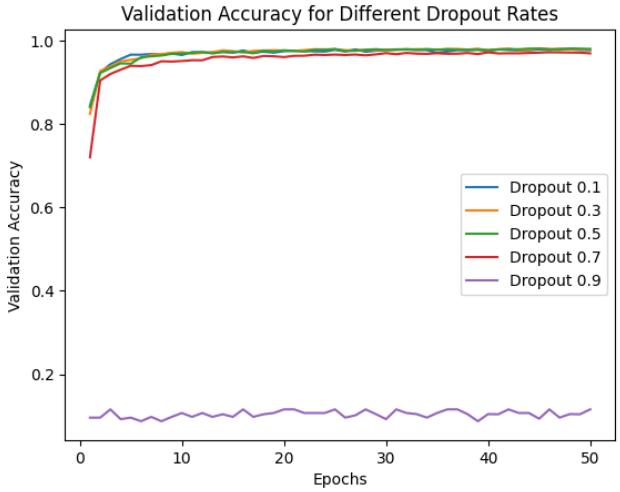

Figure 11: Varying Dropout Rates on MNIST

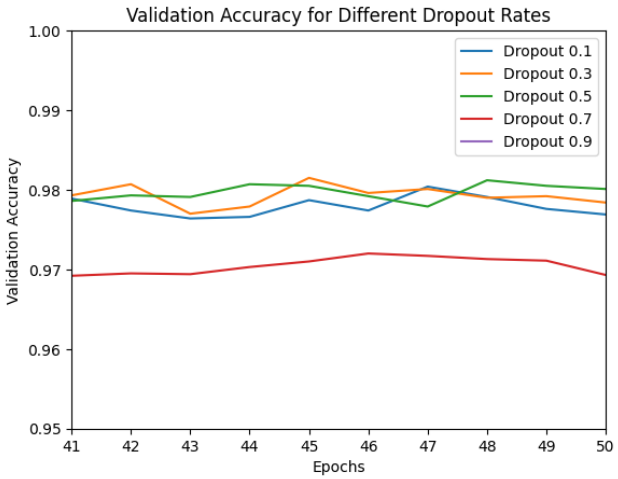

Figure 12: Varying Dropout on MNIST-Zoomed

5.2 Changing of dropout rates for SVHN dataset

Our experiment conducted on the SVHN dataset revealed that a dropout rate of 0.3 yields the highest accuracy, registering at 93.26%, which marginally surpasses the 93.04% accuracy achieved at a dropout rate of 0.5. This finding is noteworthy as the latter rate was identified as the optimal rate in the referenced paper. The divergence in results could be attributed to the complexity of the model and the variations in experimental implementations, particularly since the authors of the paper did not provide their code. Conversely, a dropout rate of 0.9 resulted in the lowest accuracy, achieving only 81.66%. The accuracy on test set is shown in Table 7. The overall result is shown in Figure 13 and Figure 14 shows the result zoomed in on the last 10 epochs.

Dropout Rate	Test Accuracy
0.1	93.10 %
0.3	**93.26 %**
0.5	93.04 %
0.7	92.86 %
0.9	81.66 %

Table 7: Effect of Varying Dropout Rates on SVHN Dataset

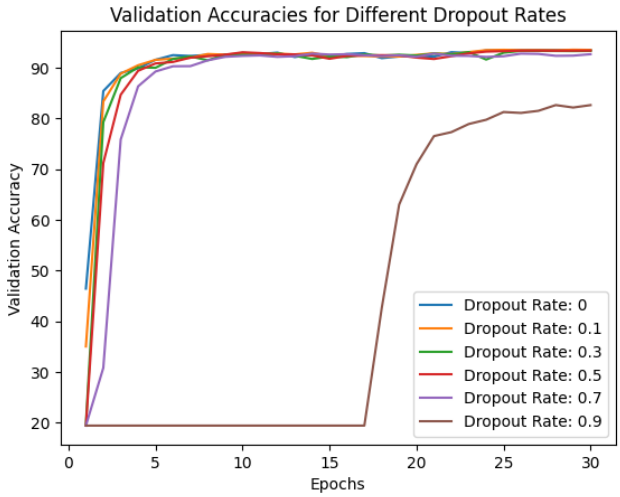

Figure 13: Varying Dropout Rates on SVHN

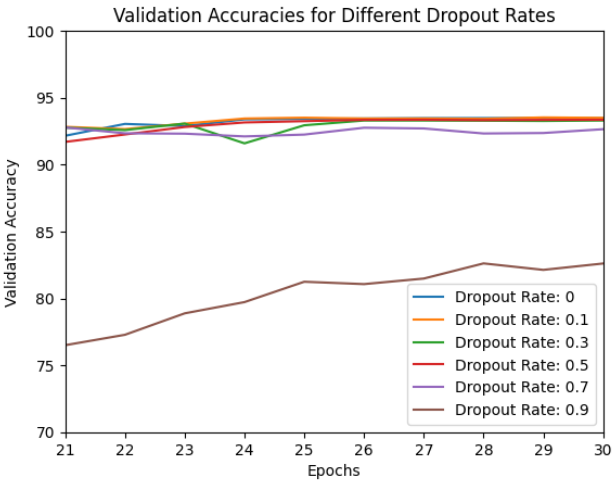

Figure 14: Varying Dropout on SVHN-Zoomed

6 Results compared with paper

In summary, our observations revealed an enhancement in accuracy across all four datasets upon the implementation of dropout techniques. These findings are consistent with those reported in the referenced paper. We noted a marginal variance in accuracy between the models employing dropout and those not utilizing it. Intriguingly, the paper indicated a generally narrower disparity between dropout and non-dropout models. This variation could potentially be attributed to differences in the models and hyperparameters used. However, it is important to note that the original code from the paper was not provided, which could be a contributing factor to these differences.

7 Challenges

The main challenge in replicating this paper was the the original code was not provided by the author. While the architecture of different models is provided, information such as data preprocessing techniques and dimensions of the datasets were missing. Minor variations in the implementation, and initialization can lead to different outcomes. This can explain why we noticed some variations, as mentioned in the previous section. Furthermore, due to the stochastic nature of dropout (randomly dropping out neurons during training), reproducing the exact results reported in the paper can be difficult. Another challenge for this project was data acquisition. Some of the datasets that the author worked with were not available or were too large to work with such as the ImageNet dataset. Hence, our experimentation focused on a subset of the datasets used in the paper. Finally, the dropout rate is a critical hyperparameter in dropout-based neural networks and it varies from model to model. Finding the optimal dropout rate for different models was a difficult task because of long training time required.

8 Discussion and Conclusion

In summary, our study extensively evaluated dropout methods in machine learning, confirming its effectiveness against overfitting in large neural networks. Building on Srivastava et al.'s foundational research, we overcame challenges in code accessibility to replicate and adjust the model, achieving similar results. Our tests covered MNIST, SVHN, CIFAR-10, and CIFAR-100 datasets, demonstrating dropout's broad applicability. The findings validate dropout's potential to improve performance across various scenarios. Future research could focus on hyperparameter optimization and applying these insights to other domains like text classification.

9 Statement of Contributions

Each team member significantly contributed across all phases of the project, ensuring a holistic involvement in preprocessing, modeling, and documentation.

10 Appendix

References

[1] Li Deng. The mnist database of handwritten digit images for machine learning research. *IEEE Signal Processing Magazine*, 29(6):141–142, 2012.

[2] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[3] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International conference on machine learning*, pages 448–456. pmlr, 2015.

[4] Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y. Ng. Reading digits in natural images with unsupervised feature learning. In *NIPS Workshop on Deep Learning and Unsupervised Feature Learning 2011*, 2011.

[5] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1):1929–1958, 2014.