

TDK-dolgozat

Mohamad Nashouqu, Tamás Traub

**Accuracy and Robustness of Sparse Neural Networks from
Watts-Strogatz Network Priors**

Mohamad Nashouqu, Tamás Traub
MSC Second-year students

Supervisor: László Gulyás, Associate Professor, ELTE Faculty of Informatics

Artificial Intelligence Department
Faculty of Informatics
Eötvös Loránd University
December - 2022

Abstract. Sparse Neural Networks (SNN) have many potential advantages, for example, lower memory requirements and potentially faster computation. In addition, they can also help to avoid overfitting. Sparsity can be achieved in several ways, among which one of the most frequently used methods is pruning. Recently, Stier et al. proposed another approach, using specific classes of sparse networks known from Network Science and using them to 'prior' the artificial neural network. In this paper, we study the use of Watts-Strogatz (WS) network priors to generate SNNs through a series of computational experiments. In particular, we train the generated SNNs on the MNIST dataset and investigate the sparse network's accuracy and robustness (against adversarial attacks) as a function of network structure (network statistics) and on the generating parameters of the WS prior. In addition, we also generate sparse Recurrent Neural Networks (RNNs) using WS priors. We identify WS parameter ranges that yield the best accuracies on the MNIST dataset. Comparing these sparse RNNs to sparse networks gained using the gradual pruning of a full network, we find that sparse RNNs from WS priors can match the performance of full networks and can outperform pruned networks even at higher levels of sparsity.

1 Introduction

Artificial neural networks are popular and successful machine learning models. They can solve very complicated problems, but usually, the harder the task, the bigger computational power is necessary. Therefore, creating a lighter model with lower computational needs, achieving the results comparable to more complex models, is desirable. Most neural networks have layers where every node is connected to every other node in the next layer and the layer before it. Leaving out some of these connections can lead to a lighter model. Actually, the human brain, what these networks try to replicate has these kind of sparse connections, too [P+08].

One method to create sparse networks from densely connected ones is pruning. The goal of pruning is to delete some of the connections in an already trained neural network. During pruning, some of the neural network's weights are masked, meaning they are turned to zero. As a result, the network can become faster and more robust. In certain cases they can even achieve better accuracy. The first pruning algorithms appeared in 1988. Optimal Brain Damage [CDS90] and Optimal Brain Surgeon [HSa93] both use the Hessian off the loss function to reduce the connections. Skeletonization [MS89] tries to pinpoint and eliminate irrelevant units of the network. Thodberg [Tho91] proposed pruning to improve generalization. Some of the most popular pruning methods are magnitude class blinded and magnitude layer wise pruning. Magnitude class blinded pruning prunes the given percentage of weights which have the smallest magnitude in the network. Magnitude layer wise pruning prunes the smallest given percentage of weights in the given layer, so the pruning is not affected by other layers in this case. Random pruning is only pruning the weights completely randomly, it serves as a benchmark to other methods.

In the last few years some authors proposed pruning the network before training, like SNIP [LAT19] and GraSP [WZG20], but they could not outperform magnitude-based pruning. Stier et al. proposed another approach. They looked at neural network structures from a network science view [SG19] and tried to build networks by a predetermined structure, namely Watts-Strogatz and Barabási-Albert graphs. They trained the networks and looked for correlation between graph properties and accuracy [SDG22].

This paper starts by replicating the various experiments of Stier et al. concerning the use of Watts-Strogatz priors. It performs an extended analysis of accuracy and robustness in case of sparse neural networks trained on the MNIST dataset. It investigates the dependence on network structure, both from the point of view of WS model parameters and the network statistics of the resulting sparse neural network. In addition, following [SDG22], the studies are extended to sparse LSTM networks also applied on the MNIST dataset, using both pruned and WS prior generated networks. After an extended analysis of the correlations between their validation accuracy and their various structural properties, a class of networks is identified that guarantees high quality results. Based on this insight, the accuracy of WS prior based networks is compared to the quality of results produced by networks yielded by various levels of pruning.

2 Background

2.1 Sparse Neural Networks

"Sparse Neural Networks are artificial neural information processing systems restricted in their structure and dynamics to conserve resources. The concept of sparseness refers to the network's connectivity structure, such that each neuron receives inputs from only a limited number of other neurons." [Thoty] Figure 1 illustrates the two kinds of connectives the dense connectivity on the left side and the sparse connectives on the right side.

Sparse networks can be generated using several approaches but in this paper, we will focus on two of them. The first one is the classic approach using pruning which starts from the standard fully connected network and then removes some nodes or edges, either during the training or after it. The second approach is that of the network prior by Mehdi Ben Amor et al [Sti21]. Network priors start by generating a sparse network (graph) according to network classes introduced in Network Science. These networks are then transformed into artificial neural networks with corresponding sparse structures.

2.2 Watts-Strogatz Network

The Watts-Strogatz model is a random undirected graph generation model that produces graphs with small-world properties, i.e. short average path lengths $L(p)$ and high clustering $C(p)$ [WS98]. Many real life biological and social networks have these same properties. Average path length measures the typical

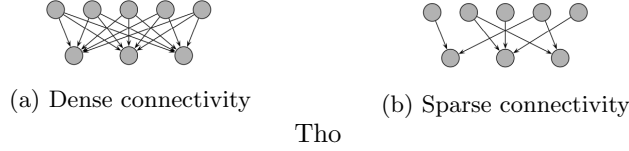


Figure 1: Principle of dense connectivity versus sparse connectivity. The arrows indicate which neurons from the bottom row, that is from the output layer, receive input from the neurons from the upper row, which forms the input layer. With sparse connectivity, the state of the output layer can be inferred more efficiently than with dense connectivity, as less information has to be processed [Tho].

separation between two nodes of the network, clustering coefficient measures the cliquishness of a typical neighbourhood. Figure 4 shows an example of a WS graph.

The Watts-Strogatz graph (WS) has three parameters. n is the number of vertices, k is the number of edges per vertex (each node is connected to their k closest neighbours) and p is the rewiring probability. First, the graph is initialized as a ring lattice, then every edge is rewired with probability p . Consequently, with $p = 0$ we acquire the initial ring lattice, but with $p = 1$ the graph becomes completely random, as visible in Figure 2. For small p values (i. e. p between 0.001 and 0.1) the $L(p)$ is low and the clustering coefficient $C(p)$ is high, as it can be seen in Figure 3. Therefore it is desired to generate WS graphs with p values in this range.

2.3 Transformation To Directed Acyclic Graph

In order to use WS networks as artificial neural networks, we transfer the WS-graph to a directed acyclic graph as Mehdi Ben Amor et al do [Sti21] with some modification, by using the lower triangular adjacency matrix of the graph and setting it as the new adjacency matrix of the graph. Diagonal entries and the upper triangular matrix are set to zero. Figure 5 shows a WS-graph on the left side and the transferred version using the described method on the right side.

In Artificial Neural Networks, nodes are organised in layers, so that is what we do too. In the first layer of the network, there are the nodes of the graph that do not have any incoming edges. The ones that do not own any outgoing edges are in the output layer. The other nodes are divided into layers in a way that every outgoing edge is going into one of the next layers. The output layer of the artificial network is not the final layer, it is followed by a layer of perceptrons.

2.4 Generating an RNN from Prior

After obtaining the layered graph, we have to turn the graph into an RNN. We turn every layer into a separate RNN. These RNNs have only one layer, and their hidden size is the number of nodes in the corresponding layer. The

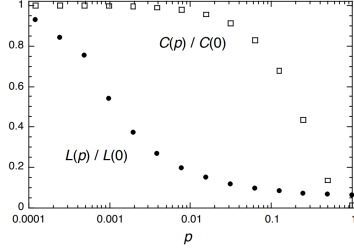


Figure 2: Characteristic path length $L(p)$ and clustering coefficient $C(p)$ normalized by the values $L(0)$, $C(0)$ for a regular lattice. The horizontal scale is logarithmic because of the rapid drop in $L(p)$. In the case of $0.001p0.1$ the $C(p)$ is still relatively high while the $L(p)$ is relatively low. The image is from the paper [WS98].

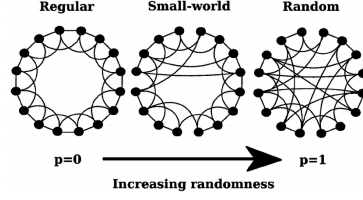


Figure 3: When the parameter p is 0, the generated graph is a ring lattice where every node is connected to its k closest neighbour. By increasing the value of p the graph becomes more random since every edge is relocated with probability p . Between 0.001 and 0.1 we can obtain a graph with the small-world properties (high $C(p)$ but low $L(p)$). At $p = 1$ the graph becomes completely random because every edge is relocated from its original place. The image is from the paper [WS98].

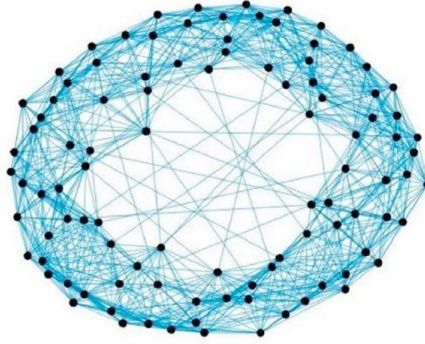


Figure 4: Illustration of a Watts-Strogatz graph with moderate rewiring [Bre]



Figure 5: Illustration of the conversion of a WS network to a directed acyclic graph.

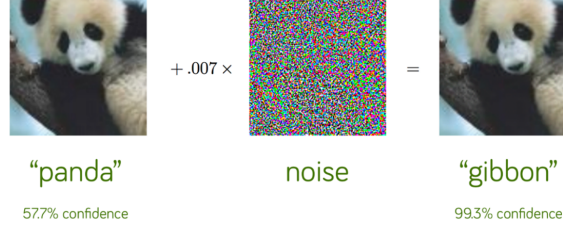


Figure 6: The addition of inconspicuous noise causes NN to classify the panda as a gibbon [JSS15]

output of these RNNs has the same size as their hidden size. But the nodes are connected to each other, not the layers. Therefore, for every node in the layer, except the input layer, we check from which layer its input is coming from. Then we give the whole output of that layer to the node and zero out everything except the output of the node where its edge is coming from. So the input size of a layer RNN is the sum of those layer’s sizes which have an outgoing edge into the given layer. If a layer A has several outgoing edges into the layer B then those are counted separately.

2.5 Adversarial Attack

An adversarial attack is a method of making small modifications to the input in such a way that the machine learning model begins to misclassify them. Neural networks (NN) are known to be vulnerable to such attacks. Research on adversarial methods historically started in the sphere of image recognition. It has been shown that minor changes in pictures, such as the addition of insignificant noise can cause remarkable changes in the predictions of the classifiers and even completely confuse ML models [JSS15]

2.5.1 The Fast Gradient Sign Method (FGSM)

The Fast Gradient Sign Method (FGSM) works by using the gradients of the neural network to create an adversarial example. For an input image, the method uses the gradients of the loss with respect to the input image to create a new image that maximizes the loss. This new image is called the adversarial image. This can be summarized using the following expression:

$$adv_x = x + \epsilon * sign(\Delta_x J(\theta, x, y))$$

where: adv_x : Adversarial image, x : Original input image, y : Original input label, ϵ : Multiplier to ensure the perturbations are small, θ : Model parameters, J : Loss.

n

2.6 Network Statistics

To study if the accuracy and robustness of the sparse neural network depend on the network structure, we rely on the following network measures (network statistics) developed by network science [BA08]

2.6.1 Density

The density of the network measures the ratio of existing links over the number of all possible links in the network. Therefore, its interpretation depends on the type of network (i.e., directed or undirected and whether self-links or loops are allowed).

2.6.2 Average path length

The term average path length is used to denote the statistics that measure the expected distances between node pairs in the network. Distance is understood as the length of the shortest path between the given pair of nodes. This interpretation is straightforward in connected networks (and, as connectedness comes easier for them, in undirected networks).

2.6.3 Eccentricity

The eccentricity of a node is its maximum distance to any other node in the network, where distance is understood as the length of the shortest path between the two nodes.

2.6.4 Degree

The degree of a node measures the number of links the node has. In a directed network, we can talk about in- and out-degree and total degree, which is the sum of the other two.

2.6.5 Closeness Centrality

Indicates how close a node is to all other nodes in the network. It is calculated as the average of the shortest path length from the node to every other node in the network. [Gol13]

Closeness Centrality formula:

$$C(x) = \frac{N - 1}{\sum d(y, x)}$$

where:

$d(y, x)$ is the distance (length of the shortest path) between vertices x and y , and N is the number of nodes in the graph.

2.6.6 Node Betweenness Centrality (NBC)

Node Betweenness centrality (NBC) measures the importance of nodes with regard to their contributions in maintaining the connectivity of the network. More formally, it considers all node pairs in the network (not including the node in question) and sums the ratios of the studied node being on the shortest path between the two nodes, that is,

$$BC(v) = \sum_{s \neq t \neq v \in V} \frac{\sigma(s, t|v)}{\sigma(s, t)}$$

where v is the set of nodes, $\sigma(s, t)$ is the number of shortest paths between nodes s and t , and $\sigma(s, t|v)$ is the number of these shortest paths going through v .

2.6.7 Edge Betweenness Centrality (EBC)

Edge Betweenness Centrality (EBC) is a measure describing the frequency at which an edge lies on the shortest path between pairs of nodes in a network. [PZM22]

Edge Betweenness Centrality(EBC) formula is:

$$C(h) = \frac{2}{N(N-1)} \sum_{a \neq b} \frac{\sigma_{ab}(h)}{\sigma_{ab}}$$

where: σ_{ab} is the number of all shortest network paths connecting nodes a and b , and $\sigma_{ab}(h)$ is the number of all of these paths that pass through edge h .

3 Accuracy and Robustness of Sparse MLPs

To study the correlation between SNNs and their graph structures from the network science perspective, first, we consider the fully connected network with one hidden layer that has 500 neurons in it trained on MNITS dataset as a baseline. The baseline model trained with hyperparameters (Alpha = 1e-03, beta1 = 0.9, beta2 = 0.999, eps = 1e-08, activation function = "RELU", epochs = 30, batch size = 100), we got 97% accuracy and 94% robustness when ϵ (of FGSM) is 0.1. Next, we investigate which Watts-Strogatz parameters (n, k, p) that produce a realistic natural network structure. Figure 7 shows the average number of nodes per layer and the transitivity according to the different rewiring probability values. According to Figure 7 to get a realistic network based on WS graphs the rewiring probability should be between 0.4 and 0.8. We continue by generating 100 Watts-Strogatz networks induced as a structural prior to Sparse Neural Networks with n, k , and p parameters selected randomly from the $n=250, 300, 350, 400, 500, k=2, 4, 6, p=0.4, 0.5, 0.6, 0.7, 0.8$ respectively. We then transfer the WS graph to a directed acyclic graph using the same method Stier et al do [Sti21] with some manipulation which is explained in 2.3 section. The generated Watts-Strogatz graph gets added to our graph set if it at least

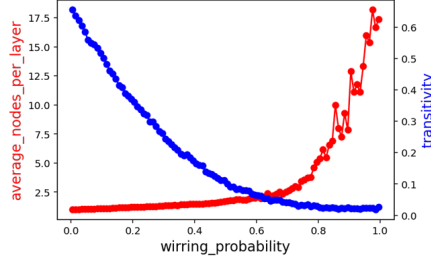


Figure 7: Transitivity/density and average nodes per layer values based on the different rewiring probabilities.

has 30 nodes in the first layer and at least 3 nodes in the other layers. After creating a sparse neural network from the directed acyclic graph (DAG), we trained the 100 models on the MNIST dataset using the same hyperparameters to exclude any possibilities of these parameters having an effect on accuracy and robustness measurement. We use Adam optimizer with (Alpha = 1e-03, beta1 = 0.9, beta2 = 0.999, eps = 1e-08, activation function = "RELU", epochs = 30, batch size = 100). After training, we apply the FGSM adversarial attack with $\epsilon = 0.1$ to the whole test set of 10,000 samples. the manipulated images were fed to the 100 models.

Figure 8 demonstrates the accuracy vs the number of parameters of the 100 SNN that were generated from WS prior. The blue dots are samples of 100 artificial sparse neural networks trained on 30 epochs on the MNIST dataset. The network structures were obtained using the network prior approach of Watts-Strogatz graphs. The WS graphs were generated randomly using the following parameters $n = 250, 300, 350, 400, 500$, $k = 2, 4, 6$, $p = 0.4, 0.5, 0.6, 0.7, 0.8$. The generated artificial networks have number of parameters between 20,000 and 150,000 parameters and the majority of the accuracies are above 95% except for two outliers that have 90%, and 92% of accuracy. Figure 8 reflects the capability of the network prior approach to produce acceptable accuracy compared to the fully connected layer network.

In order to check that our experiment is not biased to any of WS parameters (n, k, p) we do box plots: Figure 9 shows the number of parameters as a function of different values of the WS parameters. The boxes extend from the Q1 to Q3 quartile values of the data, with a line at the median (Q2). The whiskers extend from the edges of the boxes to show the range of the data. Outliers are plotted as separate dots. We plot the histogram of the number of parameters to check if it is normally distributed. Note that Figure 9 also shows the dependence of the number of parameters on the various WS parameters. In the case of number of neighbours (k), the more neighbours you have, the fewer parameters you end up with, but the variance also decreases. In contrast with the number of nodes (n) the higher the number, the more parameters you will have, but the variance also increases. The number of parameters seems independent of the ratio of rewiring (p), but its variance increases with it.

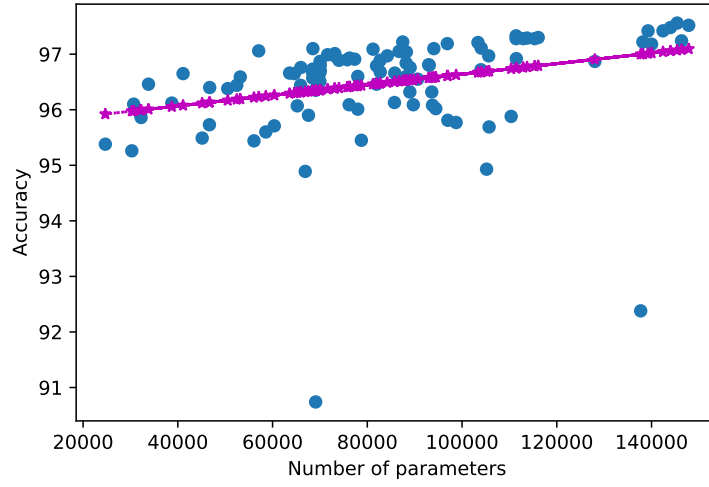


Figure 8: Scatter plot of accuracy vs number of parameters, in case of SNNs generated from WS-priors trained on the MNIST dataset, demonstrating the capability of the approach to produce acceptable accuracy. The horizontal axis shows the number of parameters, while the vertical axis plots accuracy. The blue dots are samples of 100 independent network priors generated with parameters $n = 250, 300, 350, 400, 500$, $k = 2, 4, 6$, $p = 0.4, 0.5, 0.6, 0.7, 0.8$. The purple marks and the purple line show the trend of accuracy when the number of parameters increases from 20000 to 150000.

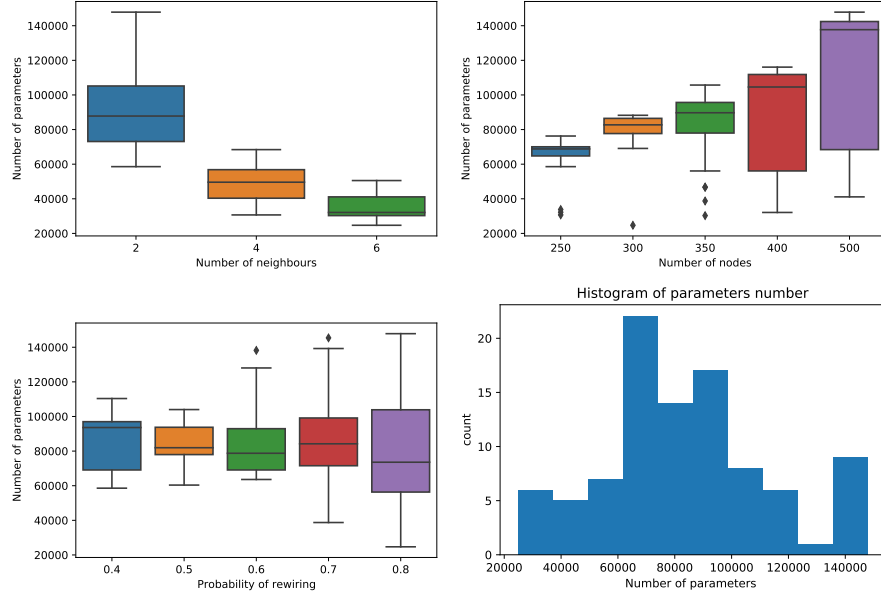


Figure 9: Dependence of the number of parameters on the various WS parameters (number of nodes, number of neighbours, probability of rewiring) of 100 Watts-Strogatz generated graphs with parameters $n = 250, 300, 350, 400, 500$, $k = 2, 4, 6$, $p = 0.4, 0.5, 0.6, 0.7, 0.8$. The boxes extend from the Q1 to Q3 quartile values of the data, with a line at the median (Q2). The whiskers extend from the edges of the boxes to show the range of the data. Outliers are plotted as separate dots. Note that in the case of number of neighbours, the more neighbours you have, the fewer parameters you end up with, but the variance also decreases, in contrast with the number of nodes the higher the number, the more parameters you will have, but the variance also increases. The number of parameters seems independent of the ratio of rewiring, but its variance increases with it.

After establishing the connection between accuracy and WS parameters, we turn our attention to the structure of the WS priors.

Figure 10 shows the dependence of validation accuracy (left column) and robustness (right column) on network structure (density, average path length, and number of layers) in the case of the 100 SNNs generated from Watts-Strogatz priors. The structural statistics are shown on the horizontal axis, while the performance measures are on the vertical axis. The density values are [0.002, 0.003, 0.004, 0.005, 0.006, 0.007, 0.008, 0.009, 0.01]. The average path lengths are [1.5, 2, 2.5, 3, 3.5, 4, 4.5, 5.5]. The number of layers is between 5 and 30. The purple line and marks show the performance trend according to the different structural measures. Note that the accuracy of the networks is above 90% and most of them are above 95%. This confirms that the WS prior-based SNNs are capable of accuracies comparable to fully connected ANNs. In addition, these results generally confirm the findings of Stier et al.: a weak negative correlation between (density, average path length, number of layers) and (accuracy, and robustness) is apparent. However, in case of robustness, the plots of Figure 10 suggest that this effect is rather an computational artefact than a real trend. Therefore, we decided not to report numerical correlation indices here.

In Figure 11 we continue the investigation of structural properties by extending our studies to the dependencies between accuracy (left column) and robustness (right column) versus maximum degree, maximum eccentricity, and maximum of betweenness by plotting the scatter (blue dots) of the 100 SNNs generated from Watts-Strogatz priors mentioned earlier. In addition, we plot the trend line of the scatter plots (purple marks and line). In the case of these network statistics, we observe no clear trends. However, in most networks, the maximum betweenness is apparently very low.

Finally, in Figure 12 we complete our study of the possible dependence on network structure by investigating the variance of degree, closeness, and betweenness centralities. As above, the dependence on accuracy is shown on the left column, while robustness is depicted on the right. The blue dots show the values for the 100 SNNs generated from Watts-Strogatz priors. In addition, we plot the trend line of the scatter plots (purple marks and line). Again, these plots do not show clear dependences between these network statistics and accuracy or robustness.

Given that we found no strong correlations between the structure of the sparse network and its accuracy or robustness when trained on the MINST dataset, we turn our attention to the effect of the generating WS parameters.

Figure 13 summarizes the dependence of accuracy and the robustness as a function of the different WS-graph parameters (n , k , and p) of the 100 trained models. The boxes extend from the Q1 to Q3 quartile values of the data, with a line at the median (Q2). The whiskers extend from the edges of the boxes to show the range of the data. Outliers are plotted as separate dots, note that the median(Q2) of all parameters has the trend for accuracy (left column) and robustness (right column). Note that the higher k is, the less accuracy we end with, while the variance is almost the same. We observe the same behavior for the robustness except that the variance gets smaller when k increases. On

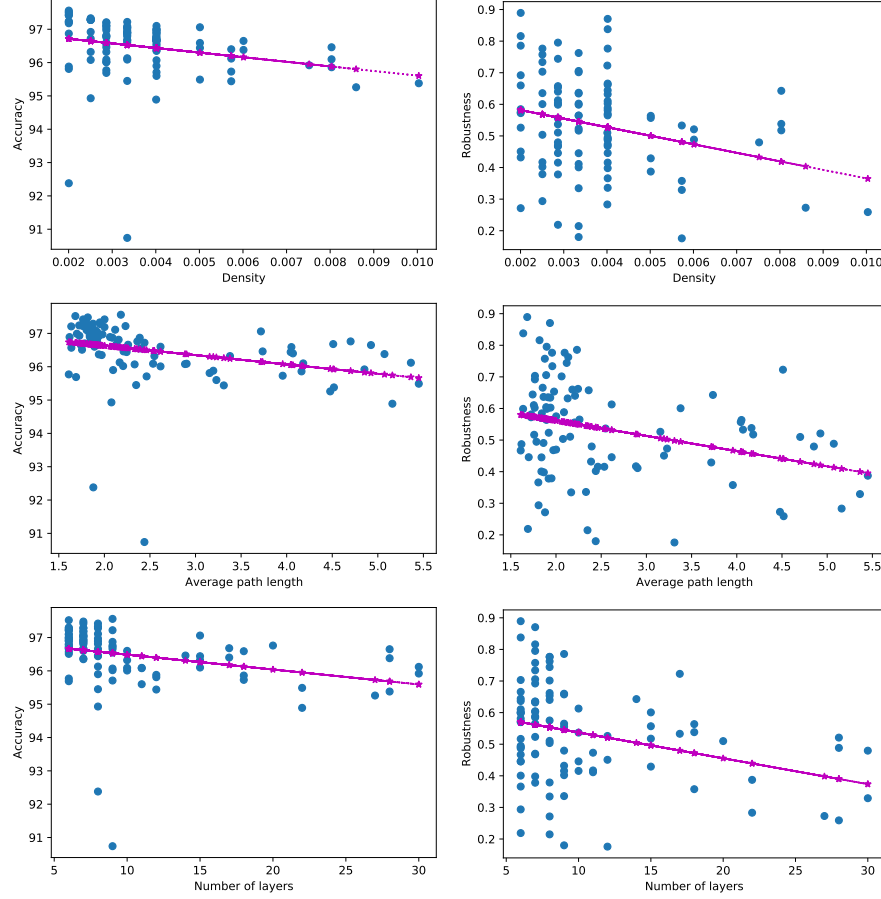


Figure 10: Dependence of validation accuracy (left column) and robustness (right column) on network structure (density, average path length, and number of layers) in case of 100 SNNs generated from Watts-Strogatz priors. The structural statistics are shown on the horizontal axis, while the performance measures are on the vertical axis. The density values are [0.002, 0.003, 0.004, 0.005, 0.006, 0.007, 0.008, 0.009, 0.01]. The average path lengths are [1.5, 2, 2.5, 3, 3.5, 4, 4.5, 5.5]. The number of layers is between 5 and 30. The purple line and marks show the performance trend according to the different structural measures. Note that the accuracy of the networks is above 90% and most of them are above 95%. that means the WS prior-based SNNs are capable of accuracies comparable to fully connected ANNs, also it seems there is a weak negative correlation between these 3 structural measures and the accuracy

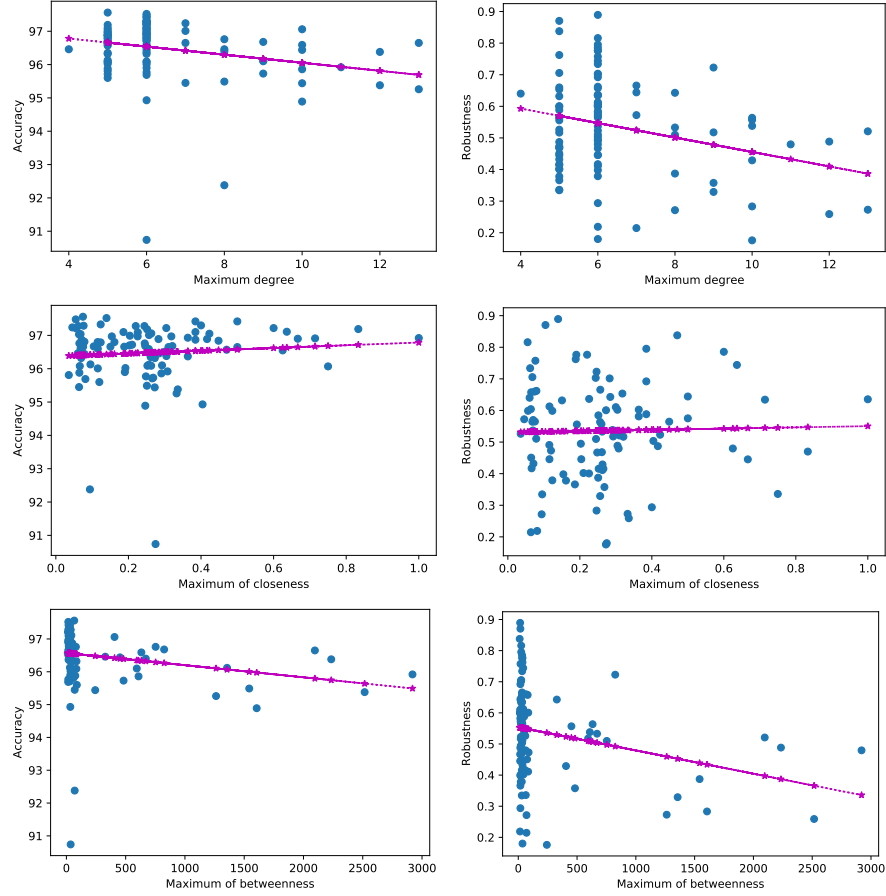


Figure 11: Dependence of validation accuracy (left column) and robustness (right column) on network structure (maximum degree, maximum of eccentricity, and maximum of betweenness) in case of 100 SNNs generated from Watts-Strogatz priors. The structural statistics are shown on the horizontal axis, while the performance measures are on the vertical axis. The purple line and marks show the performance trend according to the different structural measures. Note that based on these plots, there is no clear trend.

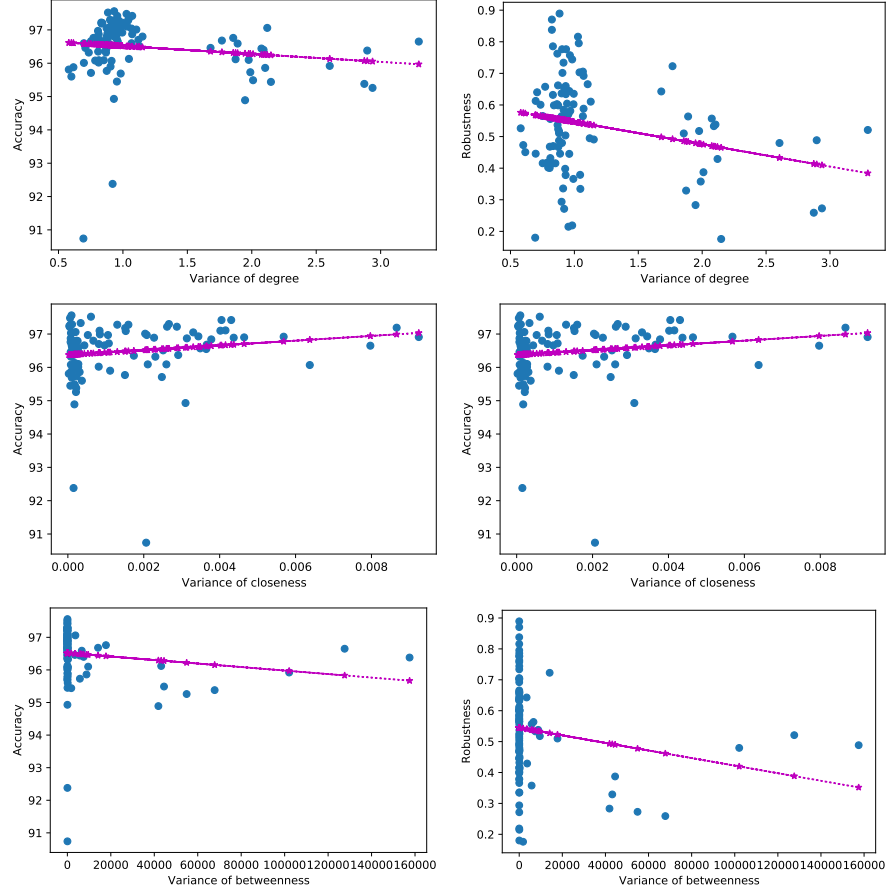


Figure 12: Dependence of validation accuracy (left column) and robustness (right column) on network structure (Variance of degree, Variance of eccentricity, and Variance of betweenness) in case of 100 SNNs generated from Watts-Strogatz priors. The structural statistics are shown on the horizontal axis, while the performance measures are on the vertical axis. The purple line and marks show the performance trend according to the different structural measures. Note that based on these plots, there is no clear trend.

the other hand, we get the opposite for p , the accuracy increases when the p increases, while the robustness remains almost the same, but with increasing variance.

4 Experiments with Sparse LSTM Networks

After studying sparse MLPs, we turn our attention toward sparse recurrent networks (RNNs). In particular, we study sparse LSTM networks. We generate three WS prior datasets. Two to search for the best structure, and one to compare with pruning. To make the comparisons fair, we create networks with the same number of nodes in them. Namely, we create the full LSTM networks with 3 layers and 16 hidden dimensions, and use $n = 48$ as a WS parameter. After both RNN types, we add a layer of perceptrons, too.

4.1 Structure of a Pruned Network

First, we study the structural properties of pruned networks. We create a full LSTM network with 3 layers and a hidden size of 16. We add one layer of perceptrons at the end of this network and train it on the MNIST dataset for 50 epochs with a learning rate of 0.01 and a batch size of 100. After the training we take the version of the model which had the best validation accuracy and prune it. By increasing the pruning percentage the accuracy starts to decrease. Our results suggest that, except for eccentricity, most network properties show a clear dependency on the pruning percentage. In many cases, the dependency is linear, but we also observe nonlinearities. In particular, both the average and variance of node betweenness are stable up until a threshold, but then collapses. The average of edge betweenness first shows a non-linear increase and then collapses.

We conduct four prunings with every percentage from 0 to 100 with a step size of 10. All these prunings are magnitude based, only differing in the type of the pruned edges. We prune the input-to-hidden (IH) edges, the hidden-to-hidden (HH) edges and both of them at the same time (IH HH), layerwise in every layer. Finally, we conduct a blinded pruning (FULL) where the edges with the smallest magnitude are pruned regardless of their layer. The network properties regarding the pruning percentages are shown in Figure 14. The figure shows the average (mean), the variance and the maximum value of the studied statistics. (The minimum and standard deviation of each measure are also collected, but omitted from the present discussion for the sake of clarity.)

4.2 Analyzing the RNNs from Network Priors

Next, we study how the structural properties of the underlying sparse network affect the accuracy of sparse LSTM networks generated from WS priors. To this end we create a sample of 64 WS networks with various parameters. For networks with 20 and 25 nodes ($n = 20, 25$) we use $k = 2, 4, 6, 8$ and $p =$

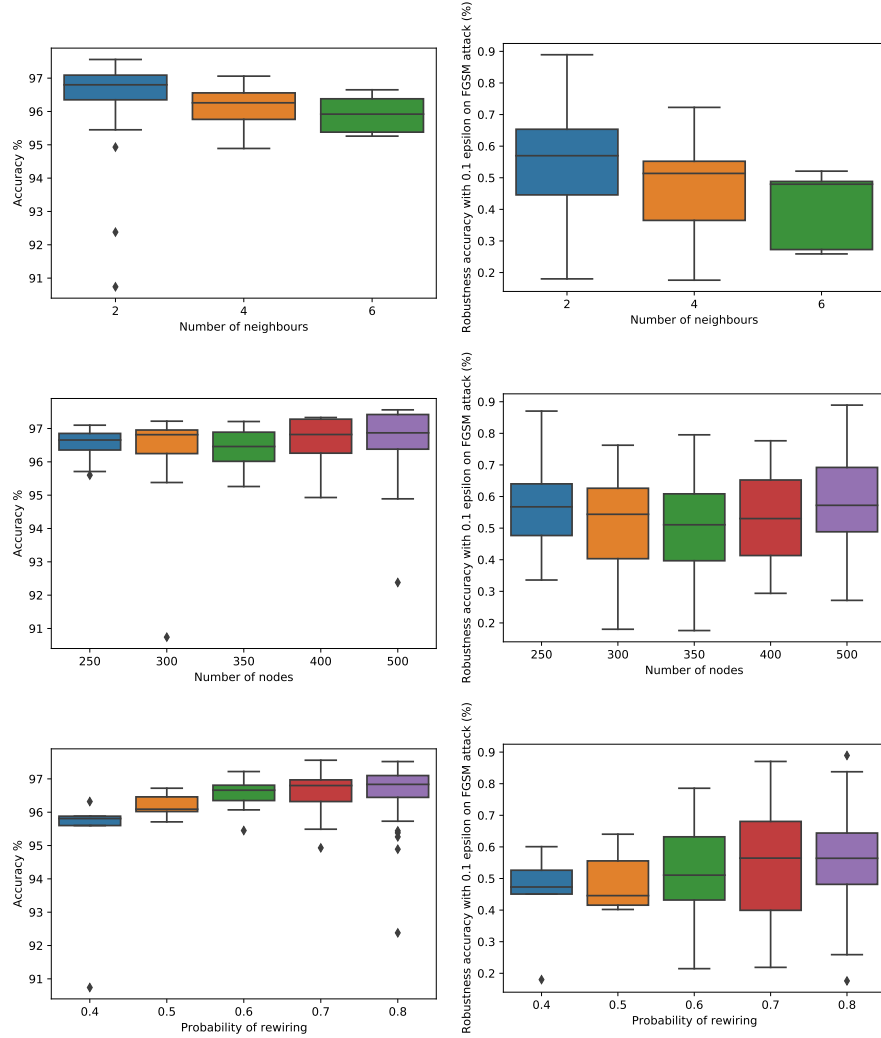


Figure 13: Boxplots summarize the dependence of accuracy and the robustness as a function of the different WS-graph parameters (n , k , and p) of 100 trained model. The boxes extend from the Q1 to Q3 quartile values of the data, with a line at the median (Q2). The whiskers extend from the edges of the boxes to show the range of the data. Outliers are plotted as separate dots, note that the median(Q2) of all parameters has the trend for accuracy (left column) and robustness(right column). Note that the more the number of neighbours, the less accuracy we end with while the variance is almost the same, and the same behavior for the robustness except that the variance gets smaller when the number of neighbours increases, but we get the opposite with the probability of rewiring, the accuracy increases when the rewiring probability increases, and the robustness almost the same but the variance in increases.

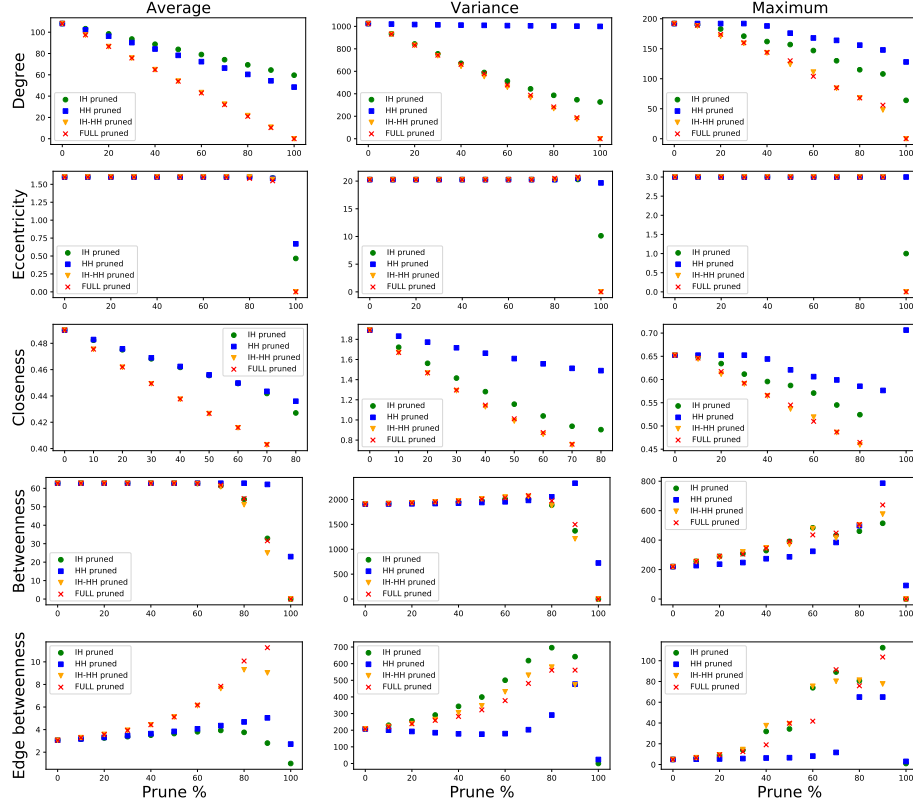


Figure 14: Network properties compared to the pruning percentage of an LSTM network. All of the plots, except the edge betweenness ones, contain the original graph and its different pruned versions from 10 to 100 percent pruning with a step size of 10 percent. Four pruning methods are shown in the figure: layerwise input-to-hidden (IH), layerwise hidden-to-hidden (HH), both IH and HH at the same time (IH HH) and a layer and edge type blinded pruning (FULL). For the edge betweenness IH HH and FULL prunings the 100% pruned versions are not shown because they result in an empty network. By increasing the pruning percentage the accuracy starts to decrease. Our results suggest that, except for eccentricity, most network properties show a clear dependency on the pruning percentage. In many cases, the dependency is linear, but we also observe non-linearities. In particular, both the average and variance of node betweenness are stable up until a threshold, but then collapses. The average of edge betweenness first shows a non-linear increase and then collapses.

0.2, 0.5, 0.7 and 1.0. For $n = 30, 40$ and 50 , the settings are $k = 2, 4, 6, 8$ and $p = 0.2$ and $p = 1.0$. For networks of size $n = 200$ and 300 we use $k = 20$ and 40 with $p = 0.9$ and 1.0 . We use these networks as network priors to generate sparse LSTM networks and train them on the MNIST dataset for 15 epochs.

In Figure 15 we show the average, variance and maximum values of eccentricity, closeness, node betweenness, edge betweenness and degree. In case of eccentricity, most networks fall in the low range, but those above a certain threshold appear to show better accuracies for higher eccentricity values. Furthermore, we observe that high average closeness increases the likelihood of high accuracy. We cannot state anything about the other attributes, so we check the parameters of the WS priors. Our findings are shown in Figure 16.

For every n and p parameter pair the best results were obtained with $p = 1.0$. We discover that with lower p values our generated RNNs have too few nodes in their layers. Since they have too few nodes, their hidden dimension is also too low and they do not have the capacity to extract important features. This is because our algorithm first transforms the initial WS graph into a directed graph where edges go from the node with the smaller index into the node with the larger index. Then it generates the layers. In Artificial Neural Networks, nodes in the same layer cannot be connected. The transformation of WS priors to neural networks must respect this. With $p = 0$ we obtain the initial ring lattice (Figure 3), in which every node is connected to its k closest neighbours. In this ring lattice every node has an edge to the node directly before and after it (the first node is connected to the last). Therefore, the generated RNN has only one node in every layer. By increasing the p value this chain can be broken and the layers can contain more nodes. In our further experiments, we used $p = 1.0$ to obtain good results. However, this means that, technically, we used random graphs instead of WS networks as priors.

Bigger k parameter means the nodes in the ring lattice are connected to more of their closest neighbours. Even if we use high p values to make our graph less like a ring lattice the number of the edges stays the same. Consequently, the layers will have less nodes, since nodes that are connected to each other cannot be in the same layer.

We want to have a better understanding of the structure of WS graphs generated with the lowest possible k (2) and highest possible p (1.0) parameters. Therefore, we create 10 WS graphs for both an n of 25, and another 10 with an n of 30, keeping $k = 2$, $p = 1.0$. The WS graphs that have a p value of 1.0 are technically random graphs, so we decide to add some random graphs to the comparison, too. We add random graphs with the same node and edge parameters, and generate 10 with $n = 25$ and $n = 30$ just like with the WS graphs. All of these networks are trained for 15 epochs with a learning rate of 0.01. The statistics can be seen in Figure 17. Nevertheless, the random graphs can achieve slightly worse accuracies than the WS priors with $p = 1.0$ on average.

The dependence on the number of edges is clear and well-understood. However, the higher the number of first layer IH edges, the better the accuracy is. Since first layer IH edges account for about half of the SNN edges, this depen-

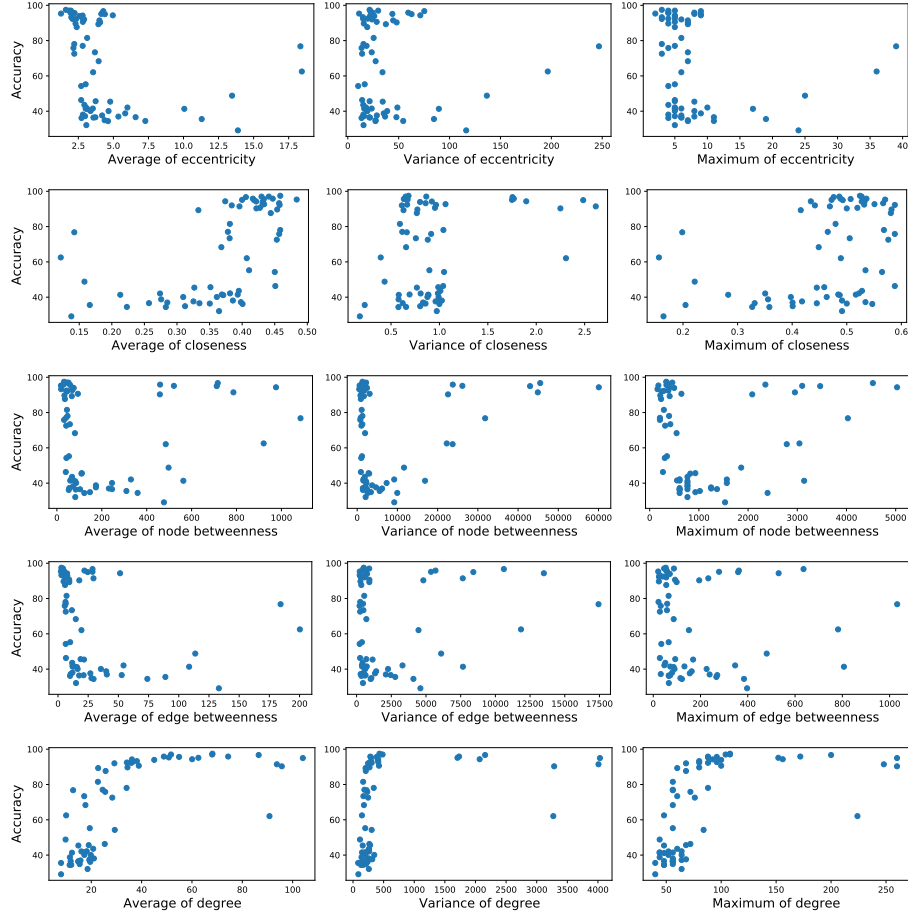


Figure 15: Dependence of validation accuracies on the various network statistics of WS prior networks. We can see that the networks with bigger average and maximum degree could achieve better results. For the bigger networks we can notice a linearly increasing accuracy with bigger average, variance and maximum eccentricity.

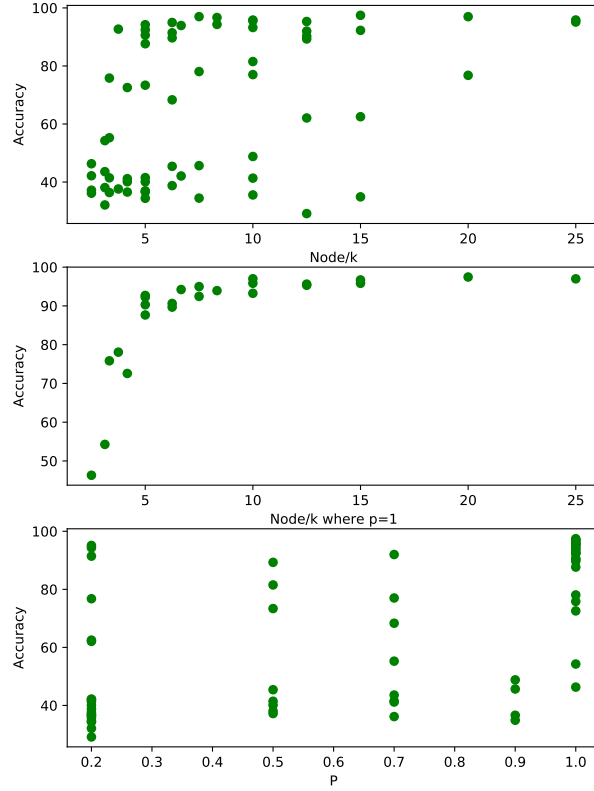


Figure 16: Validation accuracy vs structural statistics of SNNs from WS and random network priors ($n = 25$ and 30). WS prior generated LSTM RNNs vs their generating WS network parameters (p, k and n). These graphs are generated with different parameters (one graph per parameter combination). By comparing the two top plots we can discover a clear trend. Accuracy grows with increasing p and higher n/k ratio. Comparing the two top panels shows that best accuracy is achieved for $p = 1$ and higher n/k ratios. The bottom panel confirms this by its increased density of good accuracies at $p = 1.0$.

dence also explains the dependence on the variance of the number of nodes per layer. Furthermore, the number of the first layer’s IH edges is enormous compared to the other IH edges which are indeed the edges of the WS prior. Based on this observation, we fix the size of the first layer at a properly chosen value in further experiments.

4.3 Comparing Pruned and Prior Generated Networks

Finally, we compare the full LSTM networks and their pruned versions to those generated by WS priors. First, we study their accuracies vs their number of edges, then the accuracies vs the same network properties as before. In our comparison, all the initial graphs, both WS generated and pruned, have 48 nodes and 16 of those nodes were in the first layer. We include 8 networks with WS priors and 3 full LSTM networks. The WS priors are generated with the same parameters, but their seeds are different. The shared parameters are: $n = 48$, $k = 2$ and $p = 1.0$. Since the priors have a common k parameter, the generated sparse RNNs have the same number of IH edges. Their overall edge number differs because their layer sizes are different, thus some have more HH edges others have less. The full LSTM networks were created with 16 hidden dimensions and 3 layers. Every network was trained for 50 epochs with a learning rate of 0.01 on the MNIST dataset and then their version with the best accuracy on the validation set was evaluated. Several schedulers and learning rate adjustments were utilized to improve the result, but they did not make a difference. So, in the end, the learning rate stayed 0.01 throughout all the trainings. After the pruning we did not retrain the pruned model, so we could compare them with the same number of training epochs. We present our results in Figure 18 and 19.

In Figure 18 we show the accuracies vs the number of all, and IH edges. For the same pruning percentages we took the averages of the different instances of the full and pruned LSTM networks. The networks based on WS priors can generate almost as good results as the full networks, with significantly lower number of edges. As the pruning percentage grows and the edge number shrinks, the accuracy of the networks starts to drop. On the other hand, the WS prior based networks’ accuracies stayed consistent.

The accuracies vs the network properties are shown in Figure 19. In most of the cases the WS prior networks are clearly separable from the others. As the structure of the initial full LSTM network changed due to pruning, its accuracy dropped. In contrast, the accuracy of SNNs based on WS priors shows stability.

5 Conclusion and Future Work

In this paper, we analyzed sparse neural networks generated by network priors and pruning methods. We studied the correlation between the prediction accuracy of MLPs and RNNs as a function of their network structure. Special attention was devoted to the approach of using Watts-Strogatz network priors

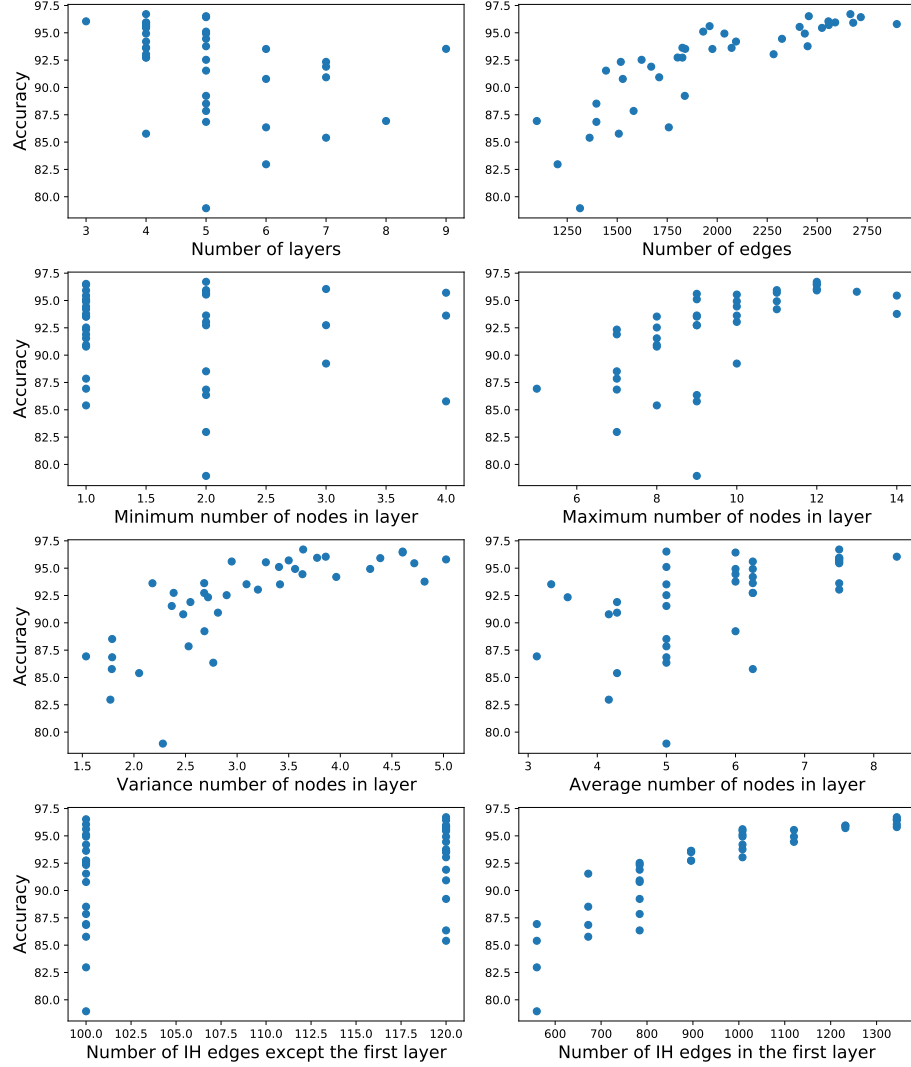


Figure 17: Validation accuracy vs structural statistics of WS and random graphs. There are clear trends regarding the number of edges, the variance in the number of nodes per layer and the number of IH edges in the first layer. In all of these cases more edges result in better results. This is consistent with the fact that more edges mean more weights. Networks with less number of layers and bigger maximum number of nodes in layer tend to have better accuracies, too.

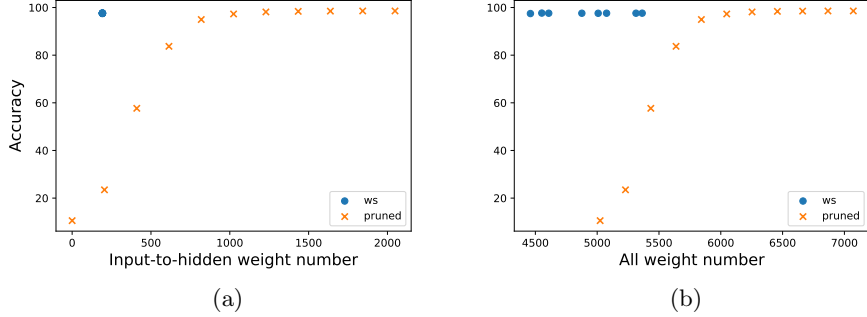


Figure 18: (a) Validation accuracy vs input-to-hidden edges (weights), (b) validation accuracy vs total number of edges (weights) in case of SNNs (generated both by pruning and from WS priors) trained for 50 epochs on the MNIST dataset. The figure shows the results of experiments with 8 SNNs from WS priors with blue circles. The orange x-es are the average accuracies of 3 full LSTM networks and the averages of their pruned versions by separate percentages. Every LSTM network was pruned with various percentages of the second and third layers’ HH edges, from 0 to 100% in 10% increments. Clearly, SNNs generated from WS priors can consistently achieve near optimal accuracies with significantly lower number of edges (weights) than their pruned counterparts.

and to the assessment of the conditions for it to yield well-performing sparse neural networks. We found that high rewiring probability is required, when WS graphs become essentially random.

We also studied the correlations between the various structural measures of the sparse networks’ and their accuracy and robustness. We generally confirm the findings of Stier et al.: there is a weak negative correlation between (density, average path length, number of layers) and (accuracy, and robustness). However, our results also suggest that this observation is rather an artefact than a true trend.

Then we turned our attention to studying the dependence between WS parameters and the accuracy. Surprisingly, we find that p has no effect on the number of parameters, while it has a positive effect on the accuracy.

Next, we aimed to make our findings more general by applying the network prior method on a different kind of network. We choose RNNs, and then we found that the best accuracies are produced by RNNs generated from WS priors with as low k and as high p as possible, just like in the case of MLPs. We also analyzed the effect of pruning on the network properties of an LSTM and compared them to prior generated networks. We found that SNNs generated from WS priors (with the optimal parameters) can consistently achieve near-optimal accuracies with a significantly lower number of edges than their pruned counterparts. We also measured the performance of the prior networks, but since they are not optimized the results could be misleading. Therefore, we did

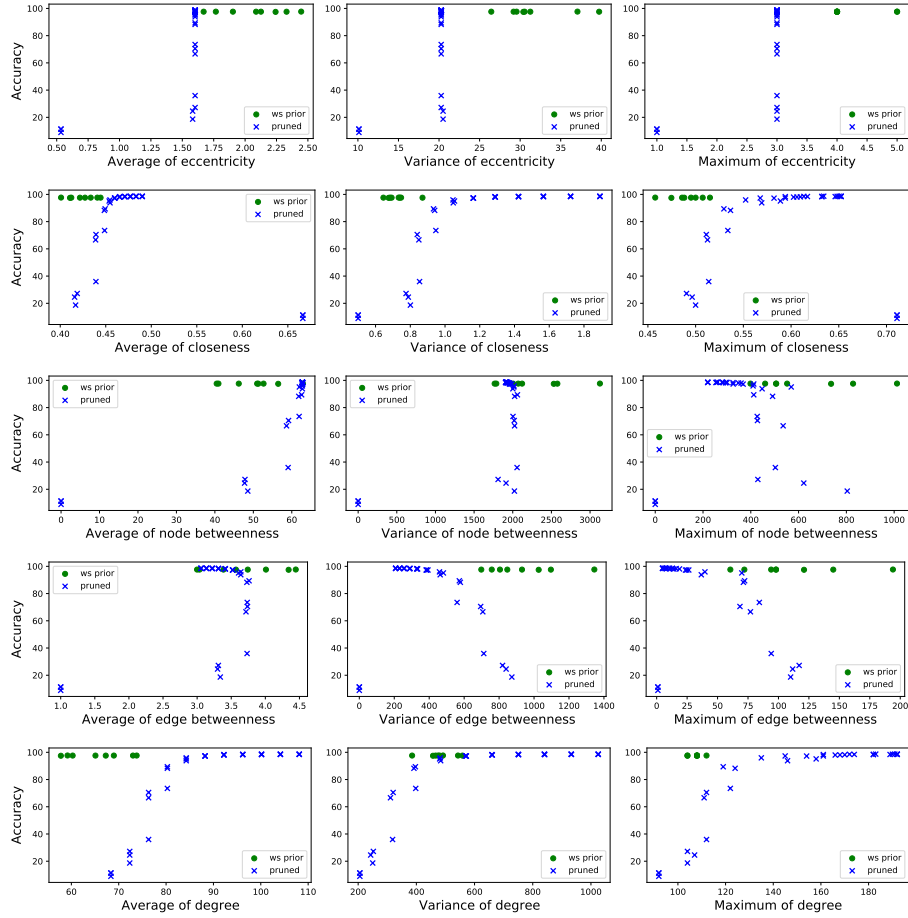


Figure 19: Dependence of validation accuracies on various network statistics. The networks shown in this figure are the same ones as in Figure 18. We can see clear differences between the two methods. The average, variance, and maximum of eccentricity are all bigger for the priors. For all other attributes, except the eccentricity, the accuracy of pruned SNNs shows a clear dependence on the changing structure of their sparsified networks. In contrast, the accuracy of SNNs based on WS priors shows stability.

not include them.

In our future work, we plan to conduct studies on other datasets and applications of neural networks. Robustness was only studied in the case of MLPs, this will be extended to RNNs too. We will also check different RNN architectures and priors based on other network models (e.g., Barabási-Albert or Jin-Girwain-Newman).

References

- [MS89] Michael C. Mozer and Paul Smolensky. “SKELETONIZATION: A TECHNIQUE FOR TRIMMING THE FAT FROM A NETWORK VIA RELEVANCE ASSESSMENT”. In: *Advances in neural information processing systems*. pp. 107–115 (1989).
- [CDS90] Yann Le Cun, John S. Denker, and Sara A. Solla. “Optimal Brain Damage”. In: *Advances in neural information processing systems*. pp. 598–605 (1990).
- [Tho91] Hans Henrik Thodberg. “IMPROVING GENERALIZATION OF NEURAL NETWORKS THROUGH PRUNING”. In: *International Journal of Neural Systems, Vol. 01, No. 04*, pp. 317–326 (1991).
- [HSa93] Babak Hassibi, David G Stork, and et al. “Second order derivatives for network pruning: Optimal brain surgeon”. In: *Neural Networks, 1993., IEEE International Conference on. IEEE*. pp. 293–299 (1993).
- [WS98] Duncan J. Watts and Steven H. Strogatz. “Collective dynamics of ‘small-world’ networks”. In: *Nature* 393.6684 (June 1998), pp. 440–442. ISSN: 1476-4687. DOI: 10.1038/30918. URL: <https://doi.org/10.1038/30918>.
- [BA08] Albert-László Barabási and Réka Albert. “Emergence of scaling in random networks”. In: *science* 286.5439, pp. 509–512 (2008).
- [P+08] Hagmann P et al. “Mapping the Structural Core of Human Cerebral Cortex”. In: *PLoS Biol* 6(7): e159 (2008).
- [LAT19] Namhoon Lee, Thalaiyasingam Ajanthan, and Philip H. S. Torr. “SNIP: Single-shot Network Pruning based on Connection Sensitivity”. In: *International Conference on Learning Representations* (2019).
- [SG19] Julian Stier and Michael Granitzer. “Structural Analysis of Sparse Neural Networks”. In: *Procedia Computer Science* 159, pp. 107–116 (2019).
- [WZG20] Chaoqi Wang, Guodong Zhang, and Roger Grosse. “Picking Winning Tickets Before Training by Preserving Gradient Flow”. In: *International Conference on Learning Representations* (2020).
- [Sti21] Julian Stier. “Correlation Analysis between the Robustness of Sparse Neural Networks and their Random Hidden Structural Priors. [On the electrodynamics of moving bodies]”. In: *Elsevier B.V* 322.10 (2021), pp. 891–921. DOI: <https://arxiv.org/abs/2107.06158>.
- [PZM22] Mahshid Pournajar, Michael Zaiser, and Paolo Moretti. “Edge betweenness centrality as a failure predictor in network models of structurally disordered materials”. In: *Scientific Reports* 12.1 (July 2022), p. 11814. ISSN: 2045-2322. DOI: 10.1038/s41598-022-15842-y. URL: <https://doi.org/10.1038/s41598-022-15842-y>.

- [SDG22] Julian Stier, Harshil Darji, and Michael Granitzer. “Experiments on Properties of Hidden Structures of Sparse Neural Networks”. In: *Machine Learning, Optimization, and Data Science*. Ed. by Giuseppe Nicosia et al. Cham: Springer International Publishing, 2022, pp. 380–394. ISBN: 978-3-030-95470-3.
- [Bre] Jan-Tino Brethouwer. *Stay nearby or get checked”: A Covid-19 control strategy*. URL: <https://doi.org/10.1016/j.idm.2020.10.013>.
- [Tho] M. Thom. *Sparse neural networks* — *Semantic Scholar*. URL: <https://www.semanticscholar.org/paper/Sparse-neural-networks-Thom/f162dcbe92e5da23fdb5b2a1f8789e1424d874fc>. (accessed: 01.09.2015).
- [Gol13] Jennifer Golbeck. “Network Structure and Measures”. In: (ScienceDirect 2013). DOI: <https://www.sciencedirect.com/topics/computer-science/closeness-centrality>.
- [JSS15] Ian J., Jonathon Shlens, and Christian Szegedy. “Explaining and harnessing adversarial examples.” In: (Published as a conference paper at ICLR 2015). DOI: <https://arxiv.org/pdf/1412.6572.pdf>.
- [Thoty] Markus Thomas. “Sparse Neural Networks”. In: 2014 UML university. Chap. 1. URL: <https://d-nb.info/1067496319/34>.