Dissertation Critique: "The Spatial Inductive Bias of Deep Learning" by Benjamin R. Mitchell

Giorgio Morales GIORGIOL.MORALESLUNA@STUDENT.MONTANA.EDU
Gianforte School of Computing
Montana State University
Bozeman, MT 59715, USA

Abstract

Deep learning techniques, particularly Convolutional Neural Networks (CNN), have become the state-of-the-art approach for solving problems in a wide range of domains (e.g. vision, text, and speech). The general intuition expresses that CNNs exploit the spatial information better than unstructured approaches, although no strong evidences have previously been presented. In that sense, this thesis attempts to provide proofs for the aforementioned intuition by stating that the spatial inductive bias plays an important role for the success of deep learning. In particular, this bias is the assumption of a spatially localized structure in the data. The author presents a set of experiments to evaluate this hypothesis: First, a performance comparison is made between CNNs in the presence and absence of that structure; and second, the spatial bias is introduced in other methods (Restricted Boltzmann Machines, Deep Belief Networks, Principal Component Analysis, and Random Forests) so that an improvement on the performance can be verified. In this dissertation critique, we will summarize the concepts, techniques and results presented in the thesis, as well as discuss the validity and impact of the hypothesis.

1. Introduction

In recent years, deep learning has gained a great deal of attention due to its ability to outperform complex problems in a variety of domains. In conjunction with the availability of large-scale datasets, such as ImageNet (Deng et al. (2009)) or COCO (Lin et al. (2014)), and modern parallel hardware architectures (GPUs), convolutional neural networks (CNNs), as the most representative deep learning technique, have allowed unprecedented achievements in fields such as computer vision, speech, natural language processing, medical diagnosis and others.

While the undeniable success of deep learning have impacted real-world applications that are used in daily-basis (e.g. mobile applications, route prediction, search engines), it remains unclear what is the key for the success of such techniques. Interestingly, CNNs have usually been referred in the literature as "black boxes" (Guidotti et al. (2018); Buhrmester et al. (2019)) given the fact that the great amount of parameters and the high-complexity of the non-linear functions learned by these models turn the prediction tasks into not traceable by humans. The lack of interpretability of CNNs is also a limiting factor for use cases requiring

explanations of the features involved in modelling for sensitive applications, like financial or medical services (Modarres et al. (2018); Pasa et al. (2019)).

In this context, Explainable AI shows up as an emerging field in machine learning that attempts to "open" the black box and provide justification on issues such as how these models learn or how they take decisions (Samek et al. (2017)). In this dissertation critique, we consider that the work presented by Mitchell (2017) falls into this category. In particular, this thesis attempts to explain that the advantage of CNNs over other approaches is due to one prior assumption, which we call "inductive-bias", about the input data.

Broadly, the inductive bias is defined as an assumption or an expectation we have a priori that will help to guide the learning process and effectively identify the target concept. That is, when we assume a priori that a type of model is worth considering and another not, we are inducing a type on inductive bias in a form of a model bias. An example of this is when we have to process hyperspectral data-cubes: Since we would like to model not only spatial but also spatial-spectral relationships within the data, we prefer to use 3-D instead of 2-D convolutional neural networks in spite of the fact that it is possible to find some 2-D CNN architecture that matches the training data similarly well. In other words, the use of some form of inductive bias helps us to restrict the hypothesis space; otherwise the basic task of machine learning (i.e. the ability to generalize) is impossible (Mitchell (1980)).

The main hypothesis presented by Mitchell (2017) is that one the factors of the success of deep learning methods, such as convolutional neural networks, is an inductive-bias that assumes that a spatially localized structure exist in the data. Furthermore, it is also hypothesized that, if the previous hypothesis is true, one could apply a spatial bias to other techniques expecting an improvement on performance, even in the context of non-connectionist techniques.

Here, we list a summary of the main claimed contributions of this thesis:

- It is empirically demonstrated that the high performance of convolutional neural networks rely on the assumption that the input data presents a spatially local structure and that it will considerably drop if we remove this inductive bias.

- A spatial inductive bias was imposed in other techniques in the form of a spatially-partitioned training scheme. This adaptation was carried on both connectionist models (Restricted Boltzmann Machines and Deep Belief Network) and non-connectionist models (Principal Component Analysis and Projective Random Forests) observing an improvement in their performance with respect to the case where no spatial structure scheme is imposed.

This dissertation critique is structured as follows: In Section 2, we briefly described the machine learning methods that will be used during the experiments; in Section 3 we formalize the definition of spatially located structure and the methods used to analyze it. In Section 4, we report the experiment results that prove the importance of a spatial bias on convolutional neural networks. In Sections 5, 6, and 7 we show the impact of imposing a spatial bias on both connectionist and non-connectionist models. In Section 8 we discuss the proposed hypotheses and their relevance for future applications, and, finally, in Section 9 we present our conclusions.

2. Background

2.1 Convolutional Neural Networks

The convolutional neural network is the most recognized deep learning model and it is mainly used for analyzing grid-like data (e.g. images or videos), although it can be applied in different domains as well (e.g. natural language processing, speech recognition). It was originally proposed by LeCun et al. (1998); since then, although with many variants, the two most important components have been the convolutional and the pooling layers.

A convolutional layer basically transforms the data representation of its input by convolving it with a set of kernels or "convolutional filters", which results in a new set of feature maps (we will discuss how the convolution operation works in Section 8). After this, we may use non-linear activation functions over the resulting feature maps (e.g. sigmoid, ReLU, Leaky ReLU) or we can apply a sort of normalization that would allow for a faster learning (e.g. batch normalization). On the other hand, pooling layers are basically used to reduce the dimensionality of preceding feature maps. To do this, we mainly use a moving fixed-size filter that calculates the average or maximum value over local non-overlapping patches of the input.

The architecture of a CNN mainly consists of an alternation of convolutional and pooling layers, so that more complex feature maps are produced in the deeper layers of the network. At the same time, this arises the problem of vanishing and exploding gradients, which is drastically reduced by using short-paths that allow to share information between different parts of the network (He et al. (2016)). Depending on the application, we might add a fully connected layer of neurons at the top of the network to perform a classification task, although sometimes this is avoided because of the great number of trainable parameters required by this type of layers (Iandola et al. (2016)). Another way to reduce the number of parameters is by using other types of convolutional layers, such as atrous or depth-wise convolutions. Finally, CNNs can be trained using many forms of backpropagation error, such as stochastic gradient descent (SGD).

2.2 Restricted Boltzmann Machines and Deep Belief Networks

A Restricted Boltzmann Machine (RBM) is a connectionist model that can learn a probability distribution over its set of inputs by optimizing the reconstruction error. It was proposed by Smolensky (1986) as a modification of the Boltzmann Machine. The main goal is to learn underlying patterns of the input data by building connections that encode those patterns.

A RBM has two layers: the visible layer and the hidden layer. Here, there is no connection (weigths set to zero) between nodes within the same layer, which means it is a complete bipartite graph; that is the difference with a Boltzmann Machine. The advantage is that this model can be represented as a Boltzmann energy distribution as follows:

$$E(x, h) = -h^\top W x - b^\top x + c^\top h, \tag{1}$$

where x and h refer to the state vectors of the visible and hidden nodes, respectively; b and c correspond to the bias vector of the visible and hidden nodes, respectively; and W is the weight vector.

The set of parameters we would like to learn is determined by W; that is, the matrix of edge weights of the graph. For this, we normally use the Contrastive Divergence (CD) method proposed by Hinton (2002). This method takes a data sample and a learning rate α as a hyperparameter. Then, it updates the model parameters in two phases: First, the probability of the hidden node is calculated for all hidden nodes, given the visible vector; second, the probability of each hidden node is determined by sampling from the model. This process can be iterative, although it has been proven that one iteration is enough for many applications.

Furthermore, a Deep Belief Network (DBN) is basically a stack of RBMs, as proposed by Hinton et al. (2006). It is considered a type of deep network based on a probabilistic graphical modelling framework. DBNs are also used for classification, especially when there is noisy or missing data; besides they are considered as a way to combat the vanishing gradient problem that happens when training deep neural networks.

DBNs are trained one layer at a time using the same CD method. We train the first layer using the input data; then, we fix the weights of this layer and generate the samples for the next layer. This process is repeated for all the layers, which is why it is considered as greedy layer-wise training.

2.3 Projective Random Forests

Before explaining the concept of a Projective Random Forest, we first need to talk about Decision Tress and Random Forests. Decision Trees are a non-parametric supervised learning method used for decision analysis. In particular, it is a very suitable approach for those cases in which it is important for humans to understand and interpret how the decisions are being made by the machine learning method. A decision tree is trained in a greedy fashion so that the training algorithm recursively splits the data according to a splitting criterion. Starting from the root node, this algorithm generates a set of children and the same process is recursively repeated for each children. The base of the tree is then a set of leaf nodes that are associated with decision values.

The splitting criterion normally consists of selecting a split that reduces a sort of measure, such as entropy or impurity, as much as possible in a single step. The first criterion, called *information gain*, was proposed by Quinlan (1986) and the objective is to minimize the entropy H, which is defined as follows:

$$H(S) = -\sum_{i=1}^{k} P(s_i) log_2(P(s_i)) = \sum_{i=1}^{k} \frac{|s_i|}{|S|} log_2(\frac{|s_i|}{|S|}), \tag{2}$$

where S is the set of examples with k classes and s_i is the set of examples with class i. Besides, $P(s_i)$ is not known so it is replaced by an estimation using counts. For a classification task, the objective is to minimize the entropy aiming to get a disjoint subset of examples, where each subset corresponds to one class label. In this context, the *information gain* is defined as how much entropy is reduced by performing a split. Then, the *information gain* for a split criterion α that assigns examples to m disjoint subsets, is given by the difference between the entropy $H(S)$ and the combined entropy of those m subsets:

$$IG(\alpha) = H(S) - \sum_{j=1}^{m} \frac{|s^j|}{|S|} H(s^j), \tag{3}$$

where s^j is the set of examples that are assigned to subset j by α.

After defining the concept of a Decision Tree and the criterion that is used to split the data, we introduce the concept of a Random Forest. Random Forests are an ensemble method mainly used for classification consisting of many decision trees. Thus, decisions are taken based on an ensemble of trees, which are trained as weak classifiers. The first method that used an ensemble of intentionally-weakened decision trees, which were trained on a random subset of features, was proposed by Ho (1995). The aim is that each element of the ensemble make mistakes independently, so the ensemble as a whole has higher performance than any of the trees. The training algorithm begins with a set of orthogonal trees that start with a subset of the full set of training examples. Then, each tree is trained with the method we defined above with the difference that, at each candidate split in the learning process, a random feature (Forest-RI) or a random subset of the features (Forest-RC) is selected. In this way, the trees are less correlated.

The Projective Random Forest was proposed by Tomita et al. (2015) as a variant of the traditional Random Forest. In this approach, instead of choosing a subset of features to use as a splitting criterion, we base decisions on weighted-linear combinations of features. That is, the features are projected onto new randomly generated oriented subspaces. Thus, this approach is more flexible and general than the Forest-RI or Forest-RC approaches.

3. Spatially Local Structure

It was stated before that many deep learning techniques assume that the data contains spatially local structure. Before attempting to run any experiments, it is of vital importance to clearly define what the author means by this. In this section, we provide a formal definition of a spatially local structure, as well as a measure to analyze this locality in different image datasets.

3.1 Local Structure

The term *locality* explains the independence of a given subspace with respect to its surroundings; that is, everything that is outside that subspace. This term is usually related to short distances between data points (e.g. clustering algorithms); however, the author uses the term *spatially local* to describe short distances between feature sample locations. On the other hand, the term *structure* is used here to talk about patterns or statistical relationships between different features of the data. Therefore, a *spatial local structure*, or simply *local structure*, is basically a structure that presents a *spatially local* property.

Moreover, when we talk about locality, we do not necessarily assure that a complete independence between the information within a given subspace and information outside that subspace exists; on contrary, we would just expect to find a *high locality* behaviour, which happens when the elements within a subspace are highly correlated with each other and have low correlation with elements outside the subspace. This arises the question of how to optimally partition the data into a set of subspaces so that we achieve the highest

possible locality. This is a model selection problem and, for now, we rely only on the model bias manually imposed at the moment of design.

In this context, a *partioning scheme* is a process by which we take each full-length vector in a dataset and we decomposed it into a subset of smaller vectors, one for each subspace. The resulting dataset is called a *partitioned dataset*. This new definition allows for a more formal definition of the term *spatial locality*: "a partitioning scheme where partitions are based on spatial distance between features in our data vectors". For example, in a digital image, which can be thought of as a spatially organized grid of pixel values, the *spatial distance* is concerned with the distance between corresponding pixels in that grid; as a consequence, adjacent pixels would be highly "local," while pixels from opposite sides of the image would be "non-local". This assumption does not need to fully meet in real-world applications, as it happens with other approaches where we approximately model the data according to some assumptions that will allow to exploit certain properties (e.g. Gaussian, Markov, or Naïve-Bayes assumptions).

The author also introduces the concept of *deep locality* to explain how the ideas of locality and partitioning scheme can be extended to hierarchical models. Suppose that we have an image that has been partitioned into several patches or subspaces; we can create local patch models for each one of them, but we also could construct latent variable models, each of which covers only a subset of the local "patch" models. Each of these latent models is "less local" because it spans a larger region of the original image. Then, this process is repeated until we obtain a single latent variable model that indirectly spans the entire image. This process results in a hierarchical model where each level of the hierarchy models the image with certain level of abstraction; that is what *deep locality* refers to.

In all the experiments of this thesis, the author thoroughly uses regularly spaced and sized spatial regions for partitioning the data. For the case of images, this can be represented by square regions, also called "patches". This is the natural way that convolutional neural networks works. Although it is possible to consider overlapping between these patches, the author preferred to apply a disjoint partitioning for RBMs and DBNs in order to achieve better parallelism and speed up the training process.

3.2 Locality Analysis

One of the tools that are used in this thesis to measure spatial structure in the data is the variogram. More specifically, it measures how important spatial distance is to statistical correlation. The variogram is simply defined as the variance of the difference between sample values at two locations. Then, for image data, the variogram can be interpreted as a histogram where the 'bins' represent sampling location distances, and the height of each 'column' is the mean of the variance of the sample value differences between all feature pairs that are that distance apart.

Fig. 1 shows the variograms for MNIST and CIFAR-10 datasets before and after a random spatial permutation. MNIST consists of 28×28 - pixel images of handwritten numbers in the middle of them with a white background. As expected, its variogram shows high variance for short distances and low variance for large distances (the further two samples are, the more probable that they correspond to white pixels of the background). On the other hand, CIFAR-10 consists of 32×32 - pixel natural images with varying background,

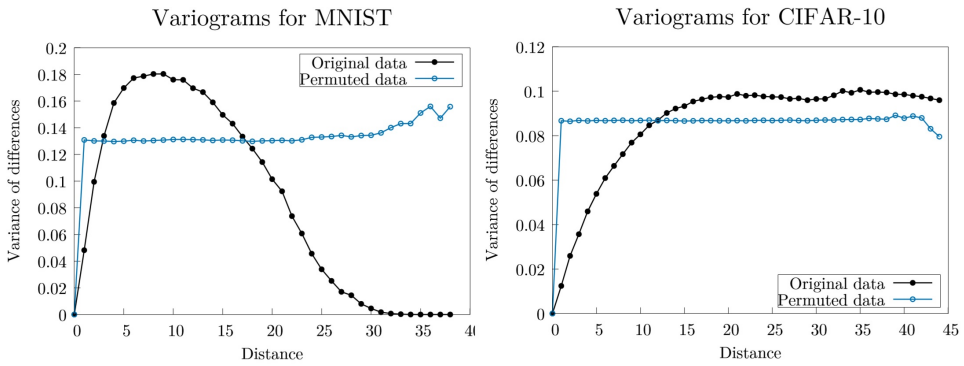

Figure 1: Variograms for MNIST and CIFAR-10 datasets for original and permuted images (Mitchell (2017)).

which explains why its variogram does not present the same fallof for large distances, as with MNIST. In both cases, the variograms for the permuted datasets are almost flat, meaning that there is no meaningful relationship between statistical information and spatial distribution. This is taken as an empirical demonstration that image data exhibits spatially localized structure and that it can be removed by applying random spatial permutation.

4. Spatial Bias in Convolutional Neural Networks

In this section, we report the results of the experiments proposed by the author in order to proof his main hypothesis. Specifically, these experiments were designed to empirically prove that the performance of CNNs heavily rely on spatially local information.

The methodology consists on training a Multilayer Perceptron (MLP), a method that makes no use of spatially local information, and a CNN in the presence and absence of spatially local structure in the data. The two datasets used for this purpose are MNIST and CIFAR-10. For the case of MNIST, the CNN consists of two 5×5 - convolutional layers of 16 and 32 filters followed by 2×2 max-pooling layers; on top of that, there are two fully connected layers of 512 and 128 units respectively. The architecture of the MLP trained on MNIST is the same as that of the last two fully connected layers of the previous CNN. For the case of CIFAR-10, the CNN was a VGG-like network (Simonyan and Zisserman (2014)) with two fully connected layers on top with 512 units each; while the MLP consists of two hidden layers with 4092 and 512 units, respectively. The comparisons of performance over the test sets are shown in Table 1, from which it can be confirmed that the hypothesis is true.

Method / Dataset	**MNIST**	**Permuted MNIST**	**CIFAR-10**	**Permuted CIFAR-10**
MLP	98.41%	98.40%	60.69%	61.27%
CNN	99.54%	97.76	92.34%	56.46%

Table 1: Classification accuracy on MNIST and CIFAR-10 test sets.

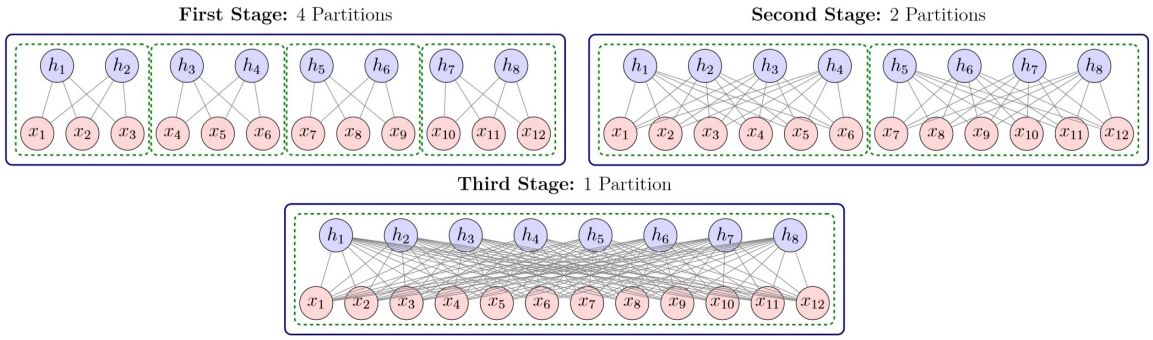

Figure 2: An example of the hierarchy partition scheme used for RBMs (Mitchell (2017)).

Configuration	Original MNIST	Permuted MNIST
Single RBM	2.55%	2.55%
RBM-28	3.32%	7.00%
RBM-20	2.20%	6.42%
RBM-15	1.87%	6.13%
RBM-10	1.64%	5.00%
RBM-5	1.49%	3.88%
RBM-2	1.44%	2.89%
RBM-1	1.42%	2.24%

Table 2: Reconstruction error of traditional and partition-trained RBMs on the MNIST dataset.

5. Partitioned Training of RBMs and DBNs

It is already proven that CNNs rely on spatially local structures in the data. Now it will be demonstrated that we could incorporate a spatial inductive bias in methods that normally do not use this kind of information. RBMs and DBNs are architecturally similar to MLPs, which leads to a behaviour of being independent of the presence of local structures in the data.

In that sense, Mitchell (2017) proposed a training scheme for RBMs that recursively partitions the network exploiting the concept of *deep locality* (described in Section 3.1), as shown in Fig. 2. Each sub-network is locally trained and their weights are re-trained in the next level of the hierarchy, so this process can be considered as a greedy pre-initilization of the weights of the model. Furthermore, in Section 2.2 we described DBNs as stacks of RBMs that are trained in a greedy layer-wise manner. Hence, the partitioned training process for DBNs just uses the partitioned training scheme for each RBM instead of the standard one.

Here, the strategy is to train RBMs and DBNs with and without a partitioning strategy so that we can compare their performances. The training process is carried on both standard and permuted data to verify that the spatially local structure in the data is being exploited. The MNIST dataset is used for this experiment and the reconstruction error is used to measure the performance of the model. Table 2 shows the comparison results for partitioned RBMs, where "Single RBM" refers to the standard RBM and "RBM-n" indicates a Parti-

Width	Validation	Classifier	Flat	Deep	Deep:Flat Wins
128	10 - fold	KNN	52.56%	53.20%	50:20
128	10 - fold	SVM	45.40%	48.09%	90:10
128	5 x 2	KNN	43.84%	44.93%	60:20
128	5 x 2	SVM	37.77%	39.63%	80:10
256	10 - fold	KNN	51.26%	52.08%	50:30
256	10 - fold	SVM	45.04%	46.71%	60:40
256	5 x 2	KNN	42.33%	43.54%	70:20
256	5 x 2	SVM	36.28%	37.83%	70:30
512	10 - fold	KNN	50.83%	52.60%	50:30
512	10 - fold	SVM	43.59%	46.57%	50:30
512	5 x 2	KNN	43.87%	45.03%	80:20
512	5 x 2	SVM	36.61%	38.47%	80:20

Table 3: Classification accuracy comparison on the "Simple Object" dataset. Last column shows the percentage of validation runs in which one technique out-performed the other.

tioned RBM with n partitions. From these results, it is concluded that Partitioned-RBMs make use of the spatially local structure of the data, which leads to better performance results. Similarly, a DBN with two RBM-(16-4-1) hidden layers (the number of partitions in each training stage is defined in parentheses) was used and it achieved an error of 1.05%. This is proof that a DBN trained using a partitioning training scheme performs better than the traditional method.

6. Deep Partitioned PCA

The previous section showed that a connectionist model can be improved by making use of spatially local structures. In this section, the author proposed a way to introduce a spatial bias into a Principal Components Analysis (PCA), which will be referred as Deep Partitioned PCA (DPPCA). This variant of the traditional PCA exploits the *deep locality* concept in a similar way that a CNN does. By choosing a linear model like this, it can be said that if there is an improvement on performance by using a "deep" version instead of a "flat" one, then it should be due to the introduction of a spatial bias and not to the mathematical power inherent in stacked non-linearities, like in the case of MLPs or CNNs.

The Deep Partitioned PCA works as follows: Given a set of images, we partition each image into a set of 4×4 - pixel patches and apply PCA on the set formed by all the patches of all the images in the dataset, selecting the top k eigenvectors to reduce its dimensionality (i.e. the reduction factor is $k/16$). Once that all the patches have been reduced, they form a new reduced dimensionality dataset so that the process can be repeated until the remaining data cannot be split. Each repetition can be considered a new level of the hierarchy. Fig. 3 shows an example where the dataset consists of only two images and only a small subset of patches at each level is being taken for the sake of visibility. Here, $v_{i_j}^l$ represents the j-th vector of image i at level l of the hierarchy, whereas $p_{i_j}^l$ is the projection of each path into the new basis.

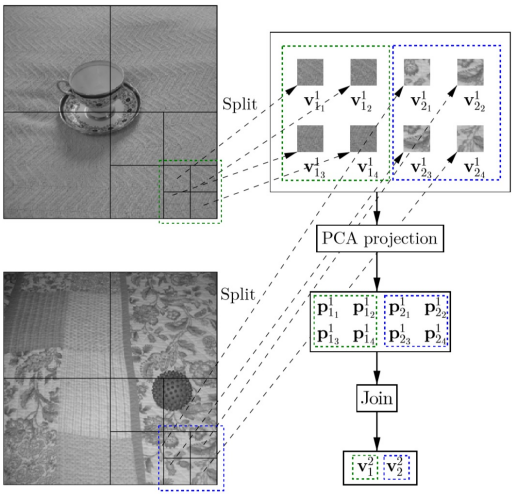

Figure 3: An example of DPPCA on two images. The v^1 are vectors in S^1, the v^2 are vectors in S^2, and the p^1 are vectors in P^1. (Mitchell (2017)).

For the experiments, the "Simple Objects" dataset proposed by Mitchell (2017) was used. Actually, this dataset consists of 3 datasets of the same images a three different resolutions (512×512, 256×256, and 128×128). A set of only 16 features was extracted from each dataset using PCA and DPPCA. Those features were trained using two classifiers: Support Vector Machine (SVM) and k - Nearest Neighbor (KNN). The validation was carried on using 10-fold and 5×2 cross-validation. The results of the 12 experiments are shown in Table 3, from which it can be seen that those classifiers that use the features extracted with DPPCA are consistently better (on average) than those extracted with PCA.

7. Spatial-biased Random Forests

In the last two sections, it was proven that it is possible to modify traditional machine learning methods so that they make use of the spatial structure in the data. However, those implementations followed a hierarchical scheme similar as that of CNNs. Therefore, in order to prove that a spatial bias can be beneficial for techniques that are completely different than that of CNNs, the author presented a spatial-biased version of the Projective Random Forests (SBPRF or Forest-RS).

Projective Random Forests (Forest-RP) create candidate projections choosing random features according to a uniform distribution. Forest-RS follows the same principle of Forest-RP with the difference that features are chosen according to a spatially-biased distribution. That means that, for spatial data, the center location of each candidate projection is sampled from a uniform distribution across each spatial dimension of the data; from that center location we form a Gaussian distribution from which the features will be sampled. The variance of that distribution is a function of the tree depth (d) of the current node and dependant on the width of the image w and the scaling factors α and β ($\sigma = (\alpha + \beta d)w$). Then, the variance will be small for the low levels of each tree, which forces to focus on

	MNIST	**SVHN**
Forest-RI	0.967	0.701
Forest-RP	0.971	0.706
Forest-RS	0.974	0.722

Table 4: Classification accuracy for random forest variants on MNIST and SVHN datasets.

local regions; on the other hand, the deeper the tree, the higher the variance, which makes the distribution be more similar to a uniform distribution, which at the same time allows to consider non-local relationships.

The experiments compared the performance of three types of random forest: Single-index Random Forests (Fores-RI), Projective Random Forests, and Spatial-biased Random Forests. For each method, c candidates splitting projections are generated (c was empirically set to 200); then, the data is sorted according to the projected value, and the optimal split is found after computing a utility estimate (in this case, the information gain criterion was chosen). For SBPRF, the parameters α and β were set to 0.05 and 0.015, respectively. The models were trained on MNIST and SVHN datasets. The SVHN dataset consists of 32×32 - pixel images of digits taken from real photographs and varying background. The results from Table 4 show that Forest-RS achieves better classification accuracy than the other two methods. Besides, it was reported that this improvement was statistically significant according to the Wilcoxon tests. The number of trees used for the MNIST and SVHN were 1008 and 384, respectively; however, it was shown that increasing the number of trees, and therefore the complexity of the model, did not affect the performance of any model.

8. Discussion

8.1 Main Hypothesis

The main hypothesis of the work presented by Mitchell (2017) is that successful deep learning techniques, such as Convolutional Neural Networks, assume the existence of spatially local structure within the data. In order to discuss this hypothesis from the perspective of this critique, first let us analyze the way a normal convolutional filter works. For the sake of simplicity, we will stick to 2-dimensional examples, though the same principle could be applicable to other kinds of data.

A 2-D convolution operation between a matrix I and a filter w is defined as follows:

$$g(x,y) = (I * w)(x,y) = \sum_{i=-\frac{m-1}{2}}^{\frac{m-1}{2}} \sum_{j=-\frac{n-1}{2}}^{\frac{n-1}{2}} w(i,j) I(x-i, y-j), \tag{4}$$

where (x, y) is concerned with the image coordinates, while (i, j) deals with those of the kernel, whose size is of (m, n). Visually, this can be interpreted as if we were moving a sliding window w through the image performing a point-wise multiplication between w and the current local patch of I so that we store the sum of those products in the corresponding position of the new matrix g.

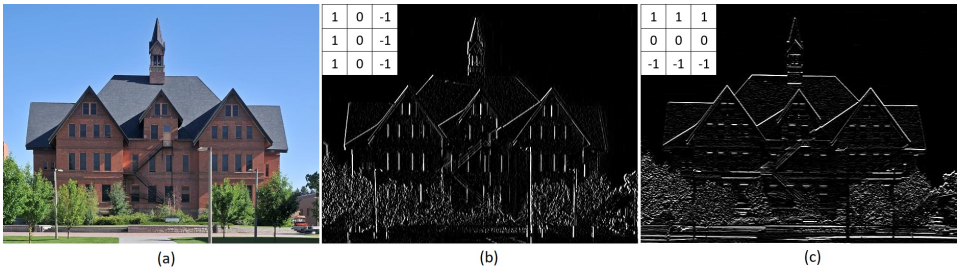

Figure 4: A 2-D convolution example. **(a)** Original image. **(b)** Vertical edges extracted. (c) Horizontal edges extracted.

Depending on the values or weights of w, the matrix g will represent different features extracted from I. For example, Fig. 4 shows the result of two convolutional operations using 3×3 - Prewitt filters with different orientations. Both results extracted vertical and horizontal lines, respectively, from the original image by analyzing the spatial structure of 3×3 local neighborhoods. This is true because the value of g at position (x, y) is affected only by the surrounding pixels of $I(x, y)$ within a window size (m, n), as shown in Eq. 4; this means that a convolutional filter specifically analyzes the spatial structure of local neighborhoods and treats them as if they were completely independent of the rest of the image, like a Markov Random Field (MRF). The most important aspect is that **the resulting matrix g at position (x, y) is a sort of measure of how similar the structure of the filter is to the structure of the local neighborhood** (Shapiro and Stockman (2001)), as seen in Fig. 4.b and Fig. 4.c. Therefore, it is logical to convey that if this inherent spatial structure is removed (i.e. if we apply a spatial permutation on I), then no meaningful information can be extracted from I using w.

Fig. 4.b and Fig. 4.c can be thought of as feature maps. In a CNN, a single layer may contain many convolutional filters and each one of them produces a feature map. Since a convolutional neural network is a hierarchical model, feature maps serve as inputs for the following layers so that more complex information can be extracted. As stated before, if we remove the spatial structure from the input data, we will produce meaningless feature maps, which will be propagated towards deeper layers of the network generating meaningless information.

The previous explanation in not only intuitive but also based on the explicit design of convolution operations. For this reason, we do not fully agree with the author when he says: *"This basic hypothesis may be fairly intuitive for many deep learning researchers. (...) However, it is a claim made without any evidence or citations to back it up"*. Nevertheless, the author presents a set of valid empirical proofs (Section 4) that confirms that the inductive bias that assumes the presence of spatially local structure in the data is of great importance for the superior performance of CNNs over other techniques.

8.2 Secondary Hypothesis

Acknowledging that the ability of using the spatially local structure of the data is important to get good results, we are now capable of modifying traditional machine learning methods

to incorporate a spatial bias expecting an improvement on their performance. The first two methods used to test this hypothesis were RBMs and DBNs. For the case of RBMs, the results consistently prove that the partitioned version of RBMs produces better results than the traditional one; however, the performance can be significantly hurt when we use that approach for permuted data. Partitioned RBMs split the model in sub-networks and train each one of them using small partitions of the data; if the data is permuted, then each partition will contain arbitrary information, so the sub-networks will not learn useful representations at lower levels of the hierarchy. This can be considered as a bad pre-initialization, which leads to bad final results. DBNs are stacks of RBMs that are trained one layer at a time, so the same logic applies for them. It would be also interesting to try this approach for other types of connectionist models, such as autoencoders with MLPs.

Regarding Deep Partitioned PCA, although the logic behind it seems to make it very suitable for grid-like data (e.g. images, datacubes), the results presented in Section 6 appear to be not entirely conclusive, especially when we look at Table 3. The "Simple Object" dataset consists of 600 images, which means each validation set has 60 images and one image represents 1.66%. In Table 3, the average difference between the accuracy obtained with the DPPCA and that of traditional PCA is just 1.60%. In fact, the Wilcoxon test concludes the difference between those results is not statistically significant. Furthermore, even when a CNN is used as a classifier, the classification accuracy is very low (48%). We believe it would be more productive if another dataset was used so that we can assure that it is possible for a classifier to achieve a high classification accuracy and reach the state of convergence. Even if we are not interested on achieving high classification metrics, it is an indicator that the model is a good description of the concept that we want to learn; from there, any improvement should be considered as valuable, whereas an increase of 1% from 37% does not necessarily mean that the model has learned something useful about the target concept (e.g. perhaps the model is getting closer to a local minimum).

The proposed adaptations for RBMs, BNs, and PCA used the concept of *deep locality*, which is similar to the way that CNNs work. For this reason, the author proposed the Spatial-biased Random Forest (Forest-RS) as a way to prove that it is not mandatory to use a hierarchical training scheme to be able to take advantage of spatially local structures in the data. In fact, the classification results showed an statistically significant improvement for the Forest-RS model over the Forest-RI and Forest-RP models in both tested datasets. Nevertheless, it is noticeable that the performance obtained in MNIST is much greater than the performance obtained in SVHN. In particular, it has been shown in Fig. 1 that MNIST presents high variance only in short distances, which does not happen with SVHN, as shown in the original thesis (in fact, its variogram is similar to that of CIFAR-10). Moreover, due to the selection of the α and β parameters, the low depth nodes of the trees are restricted to sample features from small neighborhoods; as depth increases, so does the neighborhood and the probability of sampling a white pixel in the case of MNIST. This arises the question whether Forest-RS works better for data with small local structures and uniform backgrounds rather than natural images where it is possible to sample arbitrary and not correlated features as long as the depth of the tree increases. We acknowledge that the SVHN dataset implies a more difficult classification problem in comparison with the MNIST dataset; however, it is not clear how much of the poor performance of the model is

due to the complexity of the problem per se and how much to the lack of ability of extracting information from bigger neighborhoods.

9. Conclusions

In this dissertation critique, we have summarized and discussed the concepts, methods and results presented by Mitchell (2017) in his original thesis. The main hypothesis of this work was that the success of Convolutional Neural Networks heavily relies on the inductive bias that assumes the existence of spatially local structures in the data. This hypothesis was empirically proven true by comparing the performance of CNN models trained on both normal and spatially permuted datasets. Although we consider these experiments as valid, we also discussed that this was the expected behaviour due to the way that a convolution operation works; that is, it is a sort of measure of similarity between the structure of the convolutional filter and the local structure of the data. Then, when we learn different weights for multiple convolutional filters within a CNN, what we are really doing is learning to extract different kinds of spatial structures from the data. If any spatial structure within the data is lost, then a CNN is not expected to perform well.

The secondary hypothesis is more attractive in terms of applications. It states that it is possible to impose a spatial bias on methods that normally do not use any local structure of the data and, by doing so, we could expect an improvement in their performance. Some connectionist models, such as MLPs, RBMs, and DBNs, consider the data simply as a set of features without taking into account their order or structure. For the case of images, for example, this assumption goes against the logic: an object in an image is not just a set of pixel values, that object will be defined also by the way that those pixel values are spatially arranged. In that sense, it was shown that RBMs and DBNs can benefit from the introduction of a spatial bias. However, the proposed modifications for the selected non-connectionist models, namely Deep Partitioned PCA and Spatial-biased Random Forests, did not show conclusive experimental results to validate the hypothesis in spite of the fact that their theoretical formulations are well structured. We believe that a better experimental design could help to reach to a better conclusion regarding the usefulness of the introduction of a spatial inductive bias in non-connectionist models.

References

V. Buhrmester, D. Münch, and M. Arens. Analysis of explainers of black box deep neural networks for computer vision: A survey. *ArXiv e-prints*, arXiv:1911.12116, 2019.

J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. ImageNet: A Large-Scale Hierarchical Image Database. In *CVPR09*, 2009.

R. Guidotti, A. Monreale, S. Ruggieri, F. Turini, D. Pedreschi, and F. Giannotti. A survey of methods for explaining black box models. *ArXiv e-prints*, arXiv:arXiv:1802.01933, 2018.

K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, June 2016.

G.E. Hinton. Training products of experts by minimizing contrastive divergence. *Neural Comput.*, 14:1771–1800, August 2002.

G.E. Hinton, S. Osindero, and Y.W. Teh. A fast learning algorithm for deep belief nets. *Neural Comput.*, 18(7):1527–1554, July 2006.

T. Ho. Random decision forests. In *Proceedings of International Conference on Document Analysis and Recognition (ICDAR)*, pages 278–282, 1995.

F.N. Iandola, S. Han, M.W. Moskewicz, K. Ashraf, W.J. Dally, and K. Keutzer. Squeezenet: Alexnet-level accuracy with 50x fewer parameters and 0.5mb model size. *ArXiv e-prints*, :1602.07360, 2016.

Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. In *Proceedings of the IEEE*, volume 86, pages 2278–2324, 1998.

T.Y. Lin, M. Maire, S. Belongie, L. Bourdev, R. Girshick, J. Hays, P. Perona, D. Ramanan, C. L. Zitnick, and P. Dollár. Microsoft coco: Common objects in context. *ArXiv e-prints*, arXiv:1405.0312, 2014.

B. R. Mitchell. *The Spatial Inductive Bias of Deep Learning*. PhD thesis, The Johns Hopkins University, 2017.

T.M. Mitchell. The need for biases in learning generalizations. Technical report, Rutgers University, 1980.

C. Modarres, M. Ibrahim, M. Louie, and J. Paisley. Towards explainable deep learning for credit lending: A case study. *ArXiv e-prints*, arXiv:1811.06471, 2018.

F. Pasa, V. Golkov, F. Pfeiffer, D. Cremers, and D. Pfeiffer. Efficient Deep Network Architectures for Fast Chest X-Ray Tuberculosis Screening and Visualization. *Scientific Reports and Visualization*, 9(1), 2019.

J. Quinlan. Induction of decision trees. *Machine Learning*, 1:81–106, 1986.

W. Samek, T. Wiegand, and K.R. Müller. Explainable artificial intelligence: Understanding, visualizing and interpreting deep learning models. *ArXiv e-prints*, arXiv:1708.08296, 2017.

L.G. Shapiro and G.C. Stockman. *Computer Vision*. Prentice-Hall, Upper Saddle River, NJ, 2001.

K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *ArXiv e-prints*, arXiv:1409.1556, 2014.

P. Smolensky. *Information Processing in Dynamical Systems: Foundations of Harmony Theory*, page 194–281. MIT Press, Cambridge, MA, USA, 1986.

T.M. Tomita, M. Maggioni, and J.T. Vogelstein. Randomer forests. *ArXiv e-prints*, arXiv:1506.03410, 2015.