Using Clustered Random Samples to Find Spurious Correlations in Neural Networks

Henry Tischler, Jenifer Hooten (Sponsoring Teacher)

The Academy for Technology and the Classics

April 10, 2024

Abstract

Neural networks are known to frequently learn from spurious correlations within their training dataset, a property known as shortcut learning. I propose a novel algorithm that tests a neural network-based classifier on a large number of samples seemingly unrelated to the data on which it was originally trained and then attempts to find groupings in the latent space amongst the samples classified into any given group. Through this, the network's spurious correlations could, in theory, be quantified as regions within this latent space, and ultimately become known.

On an experimental dataset of EMNIST digits colored by class used to train a neural network classifier, and a dataset of solid colored images to find this spurious correlation, the proposed algorithm was able to achieve an average 68.8% probability of belonging for each sample in the clusters it creates (as determined by the classifier network). Although this average probability is likely not significant enough for the proposed algorithm to be practically useful in its present form, the significance of its clustering suggests that the proposed algorithm and its potential variations have strong potential as tools for finding spurious correlations in neural networks.

1 Introduction

A neural network can be conceptualized as a "black box". While the neural network can learn complex and cognitively intense tasks, their complex architecture obscures the exact nature of how they make their decisions [5].

Additionally, those training neural networks are often faced with a problem known as "shortcut learning". Essentially, if your training dataset has a hidden pattern within it - which correlates given input samples with given outputs - the neural network may learn this pattern, which is often a "shortcut" towards the true reasoning one wants a neural network to learn [2] [6].

Shortcut learning can often be difficult to identify because of the impracticality of ascertaining the characteristics of the input data your model is making decisions based on. In this paper, I propose a novel approach to identifying and classifying the criteria that a neural network uses to make its decisions, based on feeding the neural network large amounts of theoretically uncorrelated data and then analyzing the similarities between each individual item predicted - theoretically arbitrarily - into any given group. Additionally, this novel algorithm uses a VAE (Variational Autoencoder), to reduce each item to a lower dimensional space, from which different high-dimensional items can be compared - and thus clustered - geometrically.

1.1 Past Approaches to Understanding Neural Network Behavior

Of course, many techniques have been used in the past to identify how neural networks make decisions.

Adversarial learning techniques, which aim to learn a pattern within a neural network that can then be used to manipulate the neural network into making false decisions, have been used in the past to find spurious correlations in neural networks [7]. However, because these networks are based on finding highly complex correlations that can be abused, these approaches often struggle when being tasked to find more fundamental correlations.

Additionally, for spatial data, activation heatmaps can indicate where in an image a network most strongly weights when making its decisions [3]. However, for correlations that aren't constrained to a single spatial region, activation heatmaps can often struggle to generalize the correlation.

2 Novel Algorithm Architecture

The novel algorithm described in this paper consists of the following "steps":

1. Find a "test dataset" of items which - theoretically - aren't related to your training data, though still contain the hidden pattern within your training data.
2. Train a VAE (Variational Autoencoder) on this test dataset.
3. Run each item in the test dataset through the original neural network.
4. Cluster the test dataset - after being reduced by the VAE - though weight these clusters towards each group that needs to be classified. These clusters represent the correlations of the network.

2.1 The Test Dataset

The first part of this algorithm - finding a "test dataset" - is a highly subjective task and can change depending on the type of dataset being analyzed, and the suspected correlations to inspect.

The fundamental requirement of the test dataset is that it - at some level - contains the correlations that you hope to capture.

2.2 The VAE

The VAE [4] is a deep learning model that uses two neural networks trained together - an encoder and a decoder - to reduce the dimensionality of data given to it while still preserving the meaning of these images within the lower dimensional space.

Because of this property, the algorithm is particularly useful for this project. For items as high-dimensional as an image, it would be very difficult to try to find similarities between different groups of these images.

However, after reducing these images to a lower-dimensional latent space, valuable conclusions about the similarities of images can be drawn simply from their geometric distance.

Additionally, while not used directly in this project, the VAE is also a generative tool - from these lower dimensionality embeddings, you can generate accurate reconstructions of the item itself - which makes the VAE a particularly useful tool if trying to identify what exactly the spurious correlations identified by this algorithm qualitatively mean.

2.3 Applying the test dataset to the neural network

To find patterns between the samples of the test dataset classified into their different groups I, of course, need to run each item through the neural network, and classify each image.

The main consideration with this step is that by using the sigmoid activation function, I am able to get the results of the neural network as probabilities - which allows the importance of each sample to be attenuated by the significance the model attaches to it's own predictions.

2.4 Clustering

The final output of this algorithm is several clusters across the latent space. These clusters are - ideally - representations of the latent criteria that a neural network uses to make its decisions.

To form these clusters, I took a simple K-means cluster across the whole dataset for each class but weighted it to the square of each probability that an item belongs to that given class, according to the neural network.

3 Novel Algorithm Testing

3.1 Simulated Spurious Correlation

To test this novel algorithm, I, of course, need to simulate some kind of spurious correlation within our neural network.

To achieve this, I trained a neural network to classify images from the EMNINST dataset [1] as either 0 or 1, though colored each 0 green and each 1 blue. An example of two images from this dataset can be seen in Figure 1. Because of this correlation between color and the class of the digit, the neural network only learned how to classify these digits by analyzing their color.

The spurious correlation of the model was validated experimentally, with a neural network trained with this dataset achieving a 100% rate of accuracy on the colored dataset after only one epoch, though only a 53% rate of accuracy between the digits in a monochrome dataset. [1]

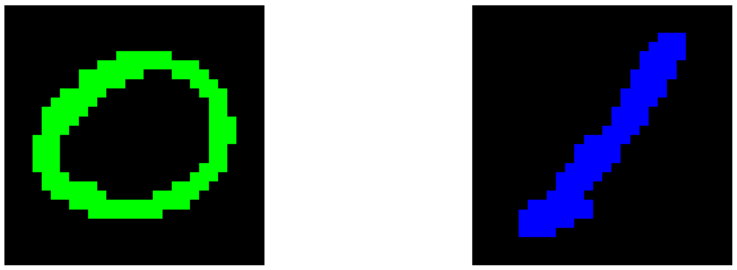

Figure 1: Two examples from this dataset. Each zero was colored green, and each one was colored blue.

[1]Both accuracy metrics were taken from a validation dataset.

Figure 2: Two randomly colored images, like the ones put in the actual test dataset used during experimentation.

3.2 The Test Dataset

For the simulated correlation to be identified, I needed a test dataset that contained the spurious correlation I hoped to identify. To achieve this - and as a minimum viable example to test this model architecture on - I used a dataset of solid-colored images. Examples of these can be seen in Figure 2.

3.3 Average Cluster Probability

The final output of this algorithm is several clusters, each of which is supposed to represent an area within the latent space that represents the characteristics that will result in an image being classified into a certain class by the neural network.

The success of the algorithm proposed in this paper is ultimately a result of how homogeneous the clusters within the latent space are, regarding their belonging to a certain class, according to the neural network.

For instance, if you imagine the algorithm working perfectly on the neural network used for testing, you would have two clusters within the latent space, each of which only contains samples that the neural network strongly believes belong to a certain class.

Conversely, if the algorithm didn't work at all, you would have two clusters within the latent space that have no homogeneity between the neural network's classifications, with the assignments being entirely random.

To measure this effect quantitatively, the final success of the algorithm proposed in this paper is ”average cluster probability”, which is the average probability, across both clusters, that the neural network gives each item belonging to the cluster it lies within.

3.4 Plots Within Latent Space

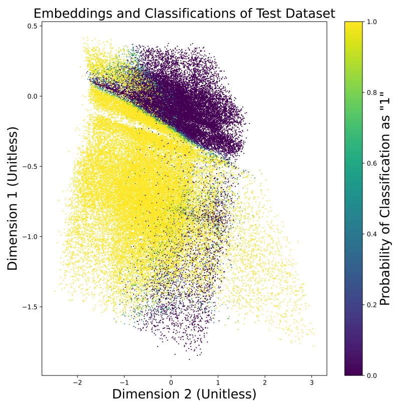

Figure 3: A scatterplot of the embeddings given to the test dataset of colored squares, and their classification probabilities.

From this experimental task, I was able to create the following scatterplots, visually representing the test dataset of solid-colored images mapped onto a 2d latent space.

In both Figures 3 and 4, each point represents an image from the test dataset mapped onto two dimensions. However, in Figure 3 each point is colored based on the classification given by the neural network, while in Figure 4, each point is colored based on the assigned cluster.

From these plots, it can be seen that while the methods of clustering used are

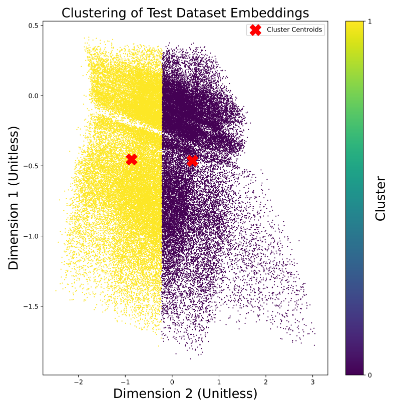

Figure 4: A scatterplot of the clustering done from the former set of embeddings and classifications.

far from perfect, there is a definite distinction in the latent space between samples classified into each group - a spatial distinction necessary for this algorithm to work.

4 Performance of Novel Algorithm

For each dimensionality level (of the VAE's encoded data) tested, the steps described in section 3 were repeated 5 times. The results of this across 4 different levels of dimensionality can be seen in Table 1.

Table 1: The final "average cluster accuracies" across several levels of dimensionality

	2 Dimensions	3 Dimensions	5 Dimensions	10 Dimensions
Average Cluster Probability	52.7%	68.8%	63.4%	63.4%
Standard Deviation	14.7%	5.46%	5.52%	9.63%

4.1 Conclusions

Based on the relative success achieved by the proposed algorithm, it's shown that the approach proposed in this paper does - to a significant degree - work, and our algorithm can find areas in the latent space the neural network is classifying from.

However, the relatively low level of average cluster probability, especially given the very simple test case, suggests that the proposed algorithm needs further refinement to be practically useful.

In particular, the drastic difference in accuracy between the number of latent dimensions suggests that the variational autoencoder - and its inherent stochasticity - may be a limiting factor for this algorithm.

References

[1] Gregory Cohen et al. *EMNIST: an extension of MNIST to handwritten letters*. 2017. arXiv: 1702.05373 [cs.CV].

[2] Robert Geirhos et al. "Shortcut learning in deep neural networks". In: *Nature Machine Intelligence* 2.11 (Nov. 2020), pp. 665–673. ISSN: 2522-5839. DOI: 10.1038/s42256-020-00257-z. URL: http://dx.doi.org/10.1038/s42256-020-00257-z.

[3] Ramprasaath R. Selvaraju et al. "Grad-CAM: Visual Explanations from Deep Networks via Gradient-Based Localization". In: *International Journal of Computer Vision* 128.2 (Oct. 2019), pp. 336–359. ISSN: 1573-1405. DOI: 10.1007/s11263-019-01228-7. URL: http://dx.doi.org/10.1007/s11263-019-01228-7.

[4] Ava Soleimany. *"MIT 6.S191: Deep Generative Modeling"*. Massachusetts Institute of Technology. URL: https://www.youtube.com/watch?v=QcLlc9lj2hk&list=PLtBw6njQRU-rwp5__7C0oIVt26ZgjG9NI&index=6.

[5] Hamed Taherdoost. “Deep Learning and Neural Networks: Decision-Making Implications”. In: *Symmetry* 15.9 (2023). ISSN: 2073-8994. DOI: 10.3390/sym15091723. URL: https://www.mdpi.com/2073-8994/15/9/1723.

[6] John R. Zech et al. “Variable generalization performance of a deep learning model to detect pneumonia in chest radiographs: A cross-sectional study”. In: *PLOS Medicine* 15.11 (Nov. 2018), pp. 1–17. DOI: 10.1371/journal.pmed.1002683. URL: https://doi.org/10.1371/journal.pmed.1002683.

[7] Brian Hu Zhang, Blake Lemoine, and Margaret Mitchell. *Mitigating Unwanted Biases with Adversarial Learning.* 2018. arXiv: 1801.07593 [cs.LG].

A Software, Materials, and Resources Used

All code for this project was run via Google Colaboratory, on a T4 GPU instance. All deep learning code was implemented via TensorFlow, and the EMNIST dataset was used to provide a minimal viable test case for the proposed algorithm.

B Experiment Code

There are two versions of the experiment code for this project. The first version - which was only used to produce Figures 3 and 4 - implements the algorithm described in this paper and tests it once, with a single level of dimensionality. This code can be viewed in the following Google Colab notebook: `https://colab.research.google.com/drive/1FTGRs-3IvW9vlPSZanOKO1IUsV3U1veA?usp=sharing`

The second version was used to produce the final data for the project, and, while far less organized, represents the true code used for the final data of the experiment, across multiple trials and levels of dimensionality. The code can be viewed in the following Google Colab notebook: `https://colab.research.google.com/drive/1fuzwOF32Cu7S0ivjZAi98YmLBc3i8Qk4?usp=sharing`

C Acknowledgment of Assistance

I'm deeply grateful to the supercomputing challenge and my sponsoring teacher for both the facilitation of the challenge itself and of my project, along with the feedback, criticism, and ideas - technical and otherwise - I was able to use to improve my project.