Université de Montréal

Improving Information Subsampling with Local Inhibition

par

Marc-André Piché

Département d'informatique et de recherche opérationnelle
Faculté des arts et des sciences

Mémoire présenté en vue de l'obtention du grade de
Maître ès sciences (M.Sc.)
en Informatique

September 2, 2022

Université de Montréal

Faculté des arts et des sciences

Ce mémoire intitulé

Improving Information Subsampling with Local Inhibition

présenté par

Marc-André Piché

a été évalué par un jury composé des personnes suivantes :

Glen Berseth

(président-rapporteur)

Irina Rish

(directeur de recherche)

Eugene Belilovsky

(membre du jury)

Résumé

L'apprentissage machine a parcouru beaucoup de chemin avec des succès marquants ces dernières années. Pourtant, les réseaux de neurones font encore des erreurs surprenantes en présence de corrélations factices. Le réseau basera sa décision sur des caractéristiques non pertinentes dans les données qui se corréleront avec son objectif lors de l'apprentissage. En revanche, les humains sont moins enclins à commettre de telles erreurs, car ils peuvent exploiter des sources d'information plus diverses pour leurs décisions. Bien que les représentations factorisées (démêlée) soient souvent présentées comme la solution pour une bonne généralisation, nous soutenons qu'un ensemble d'experts devrait apprendre aussi librement que possible avec des contraintes minimales, puis rééquilibrer leur sortie proportionnellement à leur similarité.Dans cette thèse, nous proposons une nouvelle approche : en nous inspirant de la façon dont les neurones biologiques sont connectés latéralement, nous introduisons un mécanisme pour rééquilibrer la décision des paires de neurones qui répondent de manière similaire aux entrées, favorisant la diversité de l'information par rapport à leur similarité. Nous démontrons par un cas de test extrême que l'inhibition locale peut avoir un impact positif sur la robustesse des décisions d'un réseau de neurones.

Mots clés: Precision Robuste, Robustesse, Corrélation Factice, Inhibition Locale, Réseau de neurones

Abstract

Machine learning has made remarkable strides in recent years, yet neural networks still make surprising mistakes in the presence of spurious correlations. The network will base its decision on irrelevant features in the data that happen to correlate with its objective during training. In contrast, humans are less prone to making such errors since we can leverage more diverse sources of information to make decisions. While maximally factorized or disentangled representations is often cited as the solution for generalization, we argue that an ensemble of experts should learn as freely as possible with minimal constraints and then rebalance their output proportional to their similarity. In this thesis, we propose a novel approach to do this: inspired by the lateral connections between biological neurons, we introduce a mechanism that rebalances the output of pairs of neurons that respond similarly to inputs, promoting information diversity over redundancy. We demonstrate by subjective the network to an extreme test case that local inhibition can have a positive impact on an ANN's decision robustness.

Keywords: Robust Accuracy, Robustness, Spurious Correlation, Local Inhibition, Neural Network

Contents

List of tables

List of figures

Liste des sigles et des abréviations

ANN	Réseau de neurones artifciel, de l'anglais *Artificial Neural Network*
MLP	perceptron multicouche, de l'anglais *Multilayer Perceptron*
VAE	auto-encodeur variationel, de l'anglais *Variational Auto-Encoder*
CNN	Réseau de neurones convolutionel, de l'anglais *Convolutional Neural Network*
SGD	Descente de gradient stochastique, de l'anglais *Stochastic gradient descent*
CL	Aprentissage continue, de l'anglais *Continual Learning*
ood	échantillion hors distribution, de l'anglais *out of distribution*
ISS	sous échantillionage d'information, de l'anglais *Information Sub-Sampling*
LI	Inhibition laterale ou locale, de l'anglais *Local Inhibition*

MI	information mutuelle, de l’anglais *Mutual Information*
MMA	maximisation des angles minimaux mutuels, de l’anglais *Maximisation of Minimal Angles*
SD	Découplage spectral, de l’anglais *Spectral Decoupling*
ReLU	fonction d’activation Unité Linéaire Rectifiée, de l’anglais *Rectified Linear Unit*

Chapter 1

Introduction

1.1. Robust accuracy

With the field of machine learning is continuously achieving more success, one of its most fundamental problems to solve seems ever out of grasp. It can be quite frustrating to witness the kinds error our neural networks will make. The robust accuracy domain aims at preserving accuracy beyond training, especially in situations that might not be identical to the training data. Quite a general goal with its lot of difficult problems. But that success is in part due to great strides made in : resilience to noise and neuronal death, resilience to new distribution, resilience to adversarial examples, and resilience to spurious correlation. In our goal to train models only to maximize accuracy, they tend to rely on the first features they can find that fits the training task. Usually, those features are not very robust and are easy to derail. And so, we try to control the inputs and outputs during training to try and find a structure that will still be valid at test time. But it can be quite difficult not to sacrifice accuracy in the process.

We will address the insidious problem of spurious correlations, but don't exclude that insights in this work may help other areas. We would like the model to learn a broad set of solutions so that at least one may survive in a different context.

1.2. Spurious correlation

As machine learning is tasked with solving difficult problems and finding complex solutions, one of the things that has been plaguing learning algorithms is the presence of easy solutions in the data. The problem of *spurious correlations* stems from the fact that the learner does not know what it is asked to learn. Am I classifying cows and camels, or grass and sand?[**5**] The problem falls mainly into two categories.

1.*Distributional spurious features* arise from the exploitation of the statistical correlations of the features present in the dataset by the learner to achieve the best average performance[**48**]. One usual example is how easier it to predict *gender:female* using hair colour from the CelebA[**36**] dataset since there are only a small percentage of blonde male present. As humans, we know that hair colour should have nothing to do with gender, but the statistical data shows otherwise. A problem particularly important when fairness to humans is an issue and intersectionality[**2**] plays a role. Hence, that feature should be considered spurious. Debiasing methods like rebalancing groups[**10**] in the dataset or retraining the last layer[**30**] aim to resolve this uneven distribution of samples per class by redistributing the training examples. Recent remarkable advances, like IRM[**5**], REx[**33**], and Fishr[**40**], stemming from causal theory consider invariant predictors: invariant features in regard to prediction as those to be learned, and so aim to be robust against distribution shifts.

2.*Shortcut spurious features* are undesired features strongly correlated with the target variable that are easier to learn then desired features underlying true[1] relationship between inputs and outputs. For example a learner might only need to recognize a leash to classify dogs. They are a particular problem because they starve the network from learning other informative features[**47**]. We are faced here with the shortcut learning[**20**] of a cheating student, with no way to discipline, who has learned nothing of the intended lesson. A network can rely so much on the details of certain textures[**21**], background, or other unexpected correlated features[**24**] so that even a tiny unnoticeable change in its input can shape its decision[**27**]. Not only are they hard to distinguish with the desired features, they can still be resilient to methods addressing spurious correlation from a distributional approach as they can also be invariant. Encouraging the network to learn a diverse set of features with methods like Dropout[**46**] fails as it only needs to learn the same feature twice. Representational diversity methods like MMA[**50**][**42**], constrain the network to learn orthogonal representations, but it still finds as many different ways of detecting the same feature. Finally, in our estimate, the most promising avenue is Spectral Decoupling[**39**] (SD), which imposes a cost on the network's confidence in its answers, and lets some gradient flow to other features.

Between these two types of undesirable predictive features in the dataset lies a dilemma: Should we follow Okam's razor[**4**] and look for invariance? Or should we extract as much information as possible for a decision? The best explanation should be even more invariant[**38**],

[1] As mentioned, it's important to keep in mind that there is not necessarily any truth for the model as this relationship is usually intended by the human designing the training. The model does not know what we are to teach it.

but might be more complex to decode. This work looks to address shortcut learning, and in this sense we are hoping to increase the information leveraged for a decision. We argue that whether spurious or not, an algorithm shouldn't rely on a single feature for a complex decision.

1.3. The Oracle

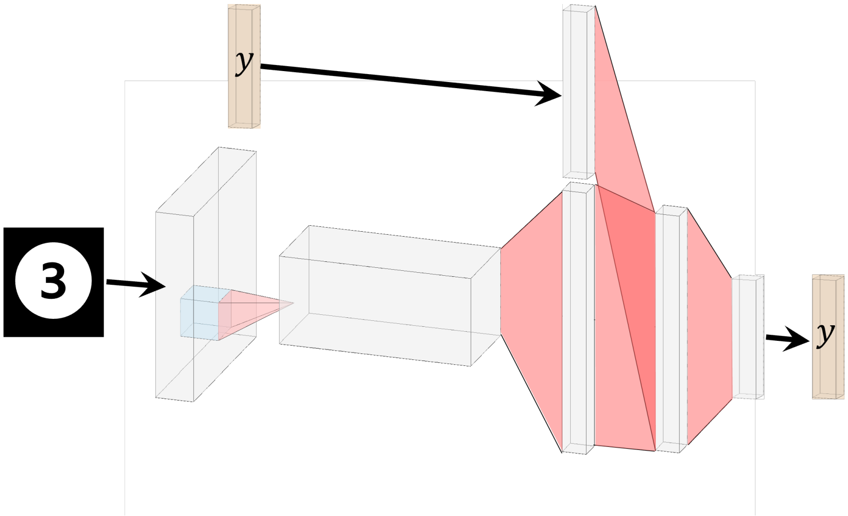

Fig. 1.1. Oracle-MNIST: To test a network's reliance on easy solutions, we test the model on the seemingly impossible task of learning a dataset in the presence of a perfect spurious correlation signal: by providing the target answer in the input. This oracle of c neurons, perfectly predicts the class with 100% accuracy during training. At test time, the oracle is permuted randomly between all classes except the correct one (0% accuracy). Success in this benchmark is any result above 11.5%.

In the Coloured-MNIST[**5**] benchmark, a modified version of the MNIST[**16**] dataset, each digit is given a colour which acts as an oracle with 80% 90% and 10% accuracy. Risk regularization methods look to learn invariant features in the distribution. These methods rely on the fact that there is some kind of clue in the dataset about which features are more robust. Diversity methods rely on sets of different representations. To test all these against shortcut learning, we can subject them to the simplest test. We integrate an oracle of c neurons directly into the network, perfectly indicating the class with 100% accuracy. In effect, we are adding y as an input with x. At test time, the oracle is permuted randomly between all class except the correct one (0% accuracy). This is how well these methods perform on Oracle-MNIST :

We can safely say that they are placing too much confidence on a single pixel. Currently, the state-of-the-art network to solve this task is one that simply always outputs 1. 11.5% accuracy, since they are 1150 1s in the test set.

Method	Train Accuracy	Test Accuracy
ERM	100.0%	0.0%
Dropout	100.0%	0.0%
IRM	100.0%	0.0%
V-REx	100.0%	0.0%
Fishr	100.0%	0.0%
SD	100.0%	0.0%
MMA	100.0%	0.0%
1*	11.5%	11.5%

Tableau 1.1. Results on Oracle-MNIST of the best known methods against spurious correlations. 1* is the baseline: a network or a function that only outputs 1. All models allocate their weights solely for to the 10 pixels of the oracle, ignoring features in the image. But at test time, these values have 0% predictability on the label.

Now, you might think that this test is unreasonably unfair. After all, have I not set up the model to fail? Should we leave some leniency so that it might have at least a slight chance to succeed? Some continual learning and transfer learning benchmarks will measure how fast the model adapts to a new environment. So maybe we could allow the model a very short period of grace, where it can learn and correct its optimism and show how much topology it has learned of the data beyond the oracle. We know that could be a useful because of last layer retraining[**30**]. But there is nothing to say that it won't pick up on another silliness. Or we could use an autoencoder so that we can decide and retrain on the latent space. But such solutions would only correct the output, and do not address how the same problem can affect every layer, deep into the network all the way to the input. The Coloured-MNIST[**5**] benchmark and its variants get around this problem by introducing artificial distribution steps. But such leniency would be unnecessary for a human to succeed the test. A human would notice and learn the extra information available associated with the oracle. And so, for a neural network passing this test, any results slightly above 10% would be a massive success; showing that the model does not rely entirely on the oracle and has learned other potential solutions.

1.4. Contributions

It is a perfectly reasonable assumption for the model to consider that if a feature in the data has 100% predictably then it is the right answer to learn. It is, however, an open question as to how we can design a model that learns more than just that, such that other solutions to act as redundancy.

This is what highlights the importance of this work. We first show that with a simple new experimental benchmark, there remains an important unsolved problem in machine learning. We then detail how we can think of two aspects of feature selection from information theory, redundancy and diversity, to understand how locality and inhibition can help address this fundamental problem. We propose two functions for them such that together, they help the network on a task that no current neural network model can solve: learning in the presence of a fully predictive feature. Finally, this neuro-inspired approach shows that there is a link between this problem and one of the most prominent structures in biological neural networks. Which reveals a new property of lateral inhibition that adds to the list of known properties it brings to a network of neurons.

Chapter 2

Motivation

2.1. InfoMax

When we train a machine learning model on a particular task, we typically assume that the training and testing data come from the same distribution. However, in practice, the distribution of the test data can be different from the training data, and the model may not perform well on out-of-distribution (OOD) data. Simply because the model has learned to rely on certain patterns or features in the training data that may not be present in the OOD data.

Our goal in this work is to avoid overfitting on a subset of information in order to minimize the chances that the model will make decisions without the features it relies on. And so, address robust accuracy in the context of spurious correlations.

And so, this goal of robust accuracy, a reliable testing accuracy that is as good or close to the training accuracy, is about developing models that can learn from data in a way that maximizes the amount of useful information extracted from it. Simply memorizing the training data is not sufficient. Instead, a good model should be able to extract relevant features from the data and make predictions based on these features.

If a dataset for a task is OOD from another one, then there must exist mutual information between the two, otherwise it would not be the same task. If we want our models to be generalizable, the decision should cover the intersection of possible environments. Future environments being unknown, the model should learn a maximum number of solutions and decide on a maximum amount of information.

Linker described as the InfoMax[**34**] principle a set of properties that an optimal information processing system should possess. He showed that neurons under a learning rule, similar to how biological neurons learn, performed principal component analysis that finds the directions of maximum variance in the dataset. These directions are called the principal components, and they represent the most important features of the dataset. For a network with these neurons, the InfoMax principle prescribes that an optimal function that maps a set of inputs X to a set of outputs Y should be learned so as to maximize the Shannon mutual information between them.

The mutual information between X and Y is defined as:

$$I(X;Y) = H(X) - H(X|Y) \tag{2.1.1}$$

Where $H(X)$ is the entropy of X and $H(X|Y)$ is the conditional entropy of X given Y. Intuitively, the mutual information measures how much information about X is conveyed by Y. The Infomax principle can be formulated as an optimization problem, where the goal is to find the parameters θ of a function f that maps X to Y such that $I(X;Y)$ is maximized:

$$\underset{\theta}{\operatorname{argmax}}\, I(X; f_\theta(X)) \tag{2.1.2}$$

One common approach to implementing Infomax is to use a neural network with a bottleneck layer, where the number of neurons in the bottleneck layer is smaller than the number of neurons in the input and output layers, so that information is compressed on a lower dimensional space. The input is projected onto the principal components and then reconstructed at the output with a linear combination of them.

For the output to be reliably predicted from the input, an optimal model must learn a set of components as diverse as possible. This would indeed contradict that a model should rely on a few features for its decision boundary in a noiseless scenario. However, to maximize mutual information in the presence of noise, the model may find that allocating for redundancy is the best solution. There may be so much noise that the model can only rely on the most prominent feature, and should allocate all of its resources to it to ensure redundancy.

Therefore, depending on the capacity of the network and noise, the model's optimal solution will lie in a trade-off between redundancy and diversity. There is a competition between redundant neurons learning the same linear combination of inputs to prevent the loss of information, and neurons learning different responses for their information value. It is with these two principles in mind that we start our work on mitigating problems of spurious correlation by aiming to build a network that learns as much as possible from the data.

2.2. Sparsity

Nowadays, sparsity is considered a critical factor for performance, and is commonly studied for its computational efficiency. But, this property also bring strong benefits to robust accuracy. Iterative training and pruning[**25**] can lead to equivalent and even better models; with less energy consumption, less memory usage, faster computation times. A network f is sparse if it has a low ratio of connectivity $\rho(f)$ with respect to all possible connections. We define the sparsity of a vector as $\rho_{\boldsymbol{x}} = \frac{||\boldsymbol{x}||_0}{d}$, with $||\boldsymbol{x}||_0$ denoting the number of non-zero elements, and $\boldsymbol{x}^0$ the binary vector of non-zero elements of $\boldsymbol{x}$.

By training a dense network, prune the less important connections and retraining the network for the remaining connections, we find a model with better accuracy than if sparsity was achieved with for example L_1 regularization. There exists subnetworks that when trained in isolation reach similar performance as the original network[**19**]

When two neurons share a same receptive field, we say there is cross sampling. The greater the intersection, the greater the chances that a message intended for one is received by the other. This can be a good thing, when redundancy is intended (2.1), but can be a threat to robustness. This is why the property that we are most interested here, is that a sparse coding is more resistant to noise[**1**]. When a random pattern is mixed with an intended signal, the corrupted signal is easier to match while the random noise is exponentially harder.

We count $\Omega(\boldsymbol{x}_i,\boldsymbol{x}_j, b)$, all the ways $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ overlap by b bits.

$$\Omega(\boldsymbol{x}_i, \boldsymbol{x}_j, b) = \binom{||\boldsymbol{x}_i||_0}{b}\binom{n - ||\boldsymbol{x}_i||_0}{||\boldsymbol{x}_j||_0 - b} \tag{2.2.1}$$

These two binomial coefficients $\binom{a}{b}$ count the number of ways to choose a subset of b bits from the $||\boldsymbol{x}_i||_0$ active bits in $\boldsymbol{x}_i$, and the number of ways to choose a subset of $||\boldsymbol{x}_j||_0 - b$ active bits from the remaining $n - ||\boldsymbol{x}_i||_0$ inactive bits in $\boldsymbol{x}_j$.

From the ratio of the number of ways that the two vectors can overlap by at least k bits, to the total number of ways that $\boldsymbol{x}_j$ can be selected from all possible n-bit binary vectors with $||\boldsymbol{x}_j||_0$ non-zero elements; we have the probability that two vectors can overlap by at least k bits:

$$P(\boldsymbol{x}_i^0 \cdot \boldsymbol{x}_j^0 \geq k) = \frac{\sum_{b=k}^{||\boldsymbol{x}_i||_0} \Omega(\boldsymbol{x}_i, \boldsymbol{x}_j, b)}{\binom{n}{||\boldsymbol{x}_j||_0}} \tag{2.2.2}$$

By simply increasing the size of the network, we have an easy solution to attain greater sparsity. But we should really consider the effective sparsity ρ^* as f samples $\mathcal{D}_\infty$: the activated space from all possible variations in the input. Simply, we want to exclude dead neurons with 0 probability of ever activating from our sparsity estimation. It's the effectively sampled space that determines the probability of cross sampling. We also have to remember that the network is free to choose its connections, and therefore offers no guarantee over cross sampling.

$P(\mathbf{x}_i^0 \cdot \boldsymbol{x}_j^0 \geq k)$ alone as a theoretical quality of resisting perturbations only refers to generic vectors with random distribution, i.e., how likely is this pattern of activation to be disturbed if we don't know what they encode. It does not, however, consider the information value of these bits, mainly that these bits of information can be perturbed from a source that is not noise, ood for example. We should therefore also keep in mind how representations really are compressed in ρ^*.

We want sparse, but not maximally sparse, as that would mean each neuron encodes a maximum amount of information, and result in entangled representations.

2.3. Information subsampling

From 2.1 & 2.2, we know that both the information space and the information flowing through it should be as diverse as possible, with the remaining space optimized with regard to sparsity occupied by redundancy. Increasing the space, the size of our network, of course, immediately allows more room for sparsity and redundant features. But how can we reconcile that with, for example, constraining diversity with an information bottleneck?
An information system, may it be an algorithm, a biological or artificial neural network, generally only needs only a subset of the information sampled to make an accurate decision. We only have to look at compress sensing[**11**] or recent advances in diffusion models[**18**][**41**] to be impressed by how small that quantity can be. And that can be true at every step as information passes through the system. There are a few desirable properties that determine the possible quality of this subsampling. How the information space is sampled is just as important as the space itself. We want to improve not just the space of representations, but how it is sampled by the weights θ and activation functions σ_l of the network.

Considering these two aspects of the process, we have :
sparse sampling : Each unit sample only a few units, θ is sparse.
sparse representation : only a few of the elements can be varied at a time and we have sparse activation[**8**]
diverse sampling : Generally no two units sample the same unit, or at least a minimal amount of cross-sampling [**42**].
diverse representation : Generally no two units represent the same feature, we have distributed representations[**8**] with independent responses to independent features.
redundant sampling : Multiple units sample the same unit and share similar receptive fields (non-zero entries of θ).
redundant representation : Multiple units have similar response to the same feature.

Notation :
$Z \to X \to Y$
Let X be the ensemble of values sampling Z, and Y sampling X. For simplicity, we can think of Z as the previous layer and Y as the next layer in a network. x y z are subsamples or subset of XYZ respectively.
Let ρ_X be the representational sparsity of X.
A *feature*, composed of *sub-features* recursively, is a specific aspect or characteristic of the input data that can be detected or extracted.
Let a signature $s \in S$ be an ensemble of sub-features that uniquely identifies a feature with probability $1 - \epsilon$.
From this, we can define the saliency of s as its magnitude $||s||$ [**43**].
Let H be the representational space of S, with a *representation* as the state of H.
Let resolution r be the number of active units of H available for sampling. Resolution is an important factor, as it describes the density of the subset of signatures of S in a sparse set X for decision y; ideally $r_s/r_H = 1$

For S to be robust, for a feature to be reliably identified, the signatures in S_A and S_B of two features should be unique enough to be easily distinguishable from each other, even in the presence of noise or other variations in the input data. This means that the probability of confusion between any two set of signatures S_A and S_B should be very low.

Ideally, the sampling should be done in such a way that each signature in S is sampled with high probability, so that there is a low probability of missing a signature or sampling it incorrectly. This would ensure that the set of signatures in S is both complete and accurate, and would lead to a more robust representation.

Therefore, for S to be robust it must be resilient to two types of perturbations. Redundancy for variations of a single sub-feature (accuracy), and diversity for variations in the sub-features of S (completeness). The number of signatures $|S|$ is not determined by a simple combinatorial count since the values are continuous, shared by other signature sets, and weighted by Y.

Diversity is usually considered as the best form of redundancy. However it might not be the optimal choice for the learner. Redundancy is especially important for the representation of very simple objects, like edges in the early visual cortex. Diversity is important for the representation of complex objects so that a signature not rely on any one feature.

A first overview of these leads to the initial impression that diverse sampling implies diverse representation. We could for example constrain θ to an orthogonal basis[**42**] or even maximize its pairwise angle with MMA[**50**] to minimize the chances that two features share information. These methods perform well when sampling an ordered space[**7**], like the receptive field of a CNN. However, there is no guarantee that a convolution of orthogonal filters will not sample the same features. A single feature may even be detected in a mutually exclusive way by two orthogonal vectors, and so be fully decorrelated[1]. As it is easy to show with toy examples, it is probably also easy for SDG to find such solutions. If X has no guarantee on feature independency, Z itself is also most surely rife with redundant features. We consider the abstract space Z^* of all the independent features possibly decodable from Z. Z^* is most surely too vast for any non-trivial dataset, since how well X effectively samples it through Z is determined by how well X encodes Z^* with its inductive priors[**23**].

Humans for example, are better at classifying written digits because when they see a digit, they also sample from the penmanship intent. We have intricate knowledge about how the tip of the pen behaves from the muscles of our fingers and our writing intent. Knowledge that an ANN, unless it would specifically be trained to understand the kinematics behind, will never have. The information is there in the image Z, but impossible to sample for X. We must conclude that there can be no guarantee of independency and no guarantee of diversity. Different words, for example, with completely different letters, especially given the context, can have the same meaning. So, even if we could constrain X to be a factorization of elements of Z, we have no way to know that two values of X do not respond to the same feature in Z^*.

Barlow suggests that nervous systems extract statistically relevant and independent features without losing information[**6**]. But how does a neuron know it is the only one to encode a certain pattern? (It doesn't: it's just a neuron.) There must some sort of process, mechanism or architecture such that a neuron encodes unique information or reacts to a unique subset of events, competing for independence even[**14**]. With no direct measure of feature information independency, we must posit that if a unit is independently firing with consistency, it is more likely to uniquely react to a feature. After all, finding a set of disentangled representations is finding a group structure and latent space s.t. independent actions act on different dimensions of the vector space[**26**]. Therefore the extracted information should be decorrelated, and the information value of a neuron determined by that measure.

Finally, we can address X^*, the set of independent features encoded by X. Following the terminology of space and weights, let's state that some information "mass" is the total amount of weight accorded to a sub-feature over a signature. With diversity and redundancy determining the number of signatures in S, we now also have the mass of each sub-feature in a signature s. Of course, sub-features that are by themselves unique identifiers s are

[1] For example a learner might want to identify the colour blue and produces two filters, one that detects vertical blue bars and one that detect horizontal bars. A simple example from ColoredMNIST[**5**] showing how difficult it can be to factorize representations.

more important to S. Should an important feature be distributed over many units for redundancy, its mass should likewise be divided[2]. Otherwise a learner can freely allocate many neurons to it to increase its mass, and even bypass constraints on weights, as shown with the OracleMNIST benchmark.

What we can do is adjust the outputs of neurons such that hopefully no single feature may outweigh the others disproportionately. For this, we propose that for a given set of perfectly correlated neuron outputs $\boldsymbol{x}$, we use a scaling function $\iota(\boldsymbol{x})$ to rescale $\boldsymbol{x}$ such that the sum of the rescaled outputs is less than or equal to the output of the neuron with the highest original output:

$$\sum_{i=1}^{n} \iota(\boldsymbol{x}, x_i) \leq x_{\max} \tag{2.3.1}$$

In a sense, with this "feature normalization", a feature is only counted once, and the weight a network would accord to it is preserved. This requires two steps, where first we evaluate how every neuron in a layer is correlated pairwise, and then renormalize the outputs of correlated neurons. The learner is then constrained to maximize the global saliency of X by recruiting new features to minimize its loss (learn uncorrelated activations); optimizing for diversity in a sparse information space. That way, in the presence of a very predictive feature, its influence on the network is limited.

To summarize what we are proposing, we conclude this section with the following analogy :

Let’s imagine a senate, an arena of democracy, where each seat carries a vote of equal weight. Some would argue that this is the optimum deciding structure. Others would argue that the seat which performed best in the past should be elected to carry all decisions. We know that at any moment one voice in particular carries the best choice, and it may even be the same one frequently. But also with certainty that it is not the best for all choices[**17**]. Ideally, we want to have as many votes as possible towards a decision, but all for different reasons. We can’t know the mind of each voter, but if two seats consistently carry the same vote, are we really interested in both their opinions? What we propose is to examine how each seat behaves and rebalance their vote accordingly.

[2] We can see later in figure 3.1 that this is not the case in an ANN without inhibition, and the impact of the number of neurons a feature is distributed over.

Chapter 3

Problem and Solutions

3.1. Shortcut Learning

Neural networks have a tendency to rely on simple features to solve their task. This is usually a desirable feature reflecting Occam's razor explaining why deep neural networks generalize remarkably well[**49**]. Since we don't give our models any real instructions on what to learn, they most often take the shortest path to fit the data. We feel that our artificial intelligence cheated and is really not intelligent at all. But an animal or a human will probably do the same thing in a similar situation[**20**]. This can be quite frustrating as we must first detect that the algorithm has used a shortcut, which may not happen until the application stage. Cats and dogs have similar features, so it might be easier to detect `leash&outdoor` vs. `indoor&couch`, and the model will use those clues. The only sure way to add contrasting data, is to add *a lot* more data. But there is no guarantee that this will not make matters worst. And for some datasets, like medical imaging, each data point can be very expensive to produce; only to have the model learn to recognize the X-ray texture of that particular table of the oncology department. Which is why the best solution is to recast your classification problem as a segmentation problem: detect exactly where the cat, or tumour, is in the image. However, SGD will not optimize to learn everything there is to know about dogs, but rather learn a set of features that separates them from cats. The latter is task accuracy oriented, while the former is better for future tasks or a distribution different from the training set. Do we really want to minimize mutual information between cats and dogs? We shouldn't be surprised that it fails to generalize to real-world ood data.

Let's take a look at the impact an oracle has on the network. In figure 3.1, we see how fast the network learns the oracle depending on its size. The oracle, a one-hot vector of the label, is duplicated n times and fed fully connected into an input layer of the same size : `nb_oracles*nb_classes`. We have included size 1 2 4 8 16 for visual clarity between `sigmoid` and `ReLu`, but the trend continues as we increase the size. We can see how faster and

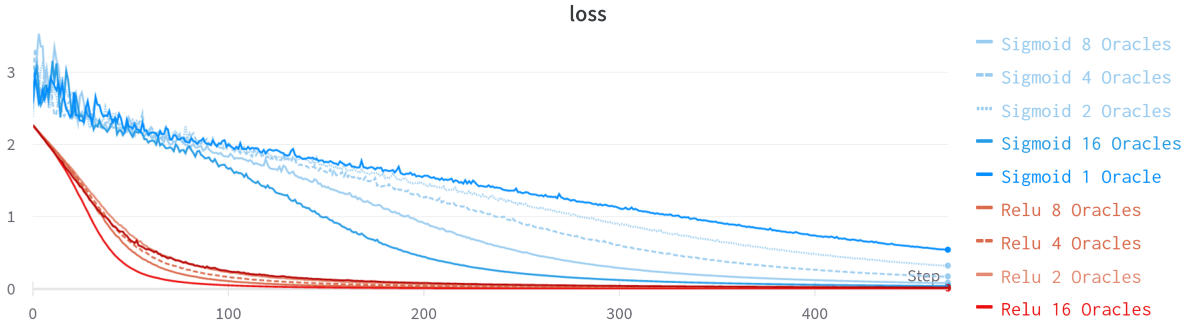

Fig. 3.1. Learning speed of `sigmoid`(blue) and `ReLu`(red) depending on the size of the oracle. In this experiment, we duplicate the oracle to show how fast a network will capture a predictive feature depending on the size of input values of that feature. the x axis is the number of samples: with an oracle of size 1 the feature is learned almost instantly inside an epoch. Duplicated oracles translates directly into faster learning. `ReLu` learns significantly faster than `sigmoid`

smoother `ReLu` is compared to `sigmoid`, which might explain its success and now ubiquitous use. It is not surprising that with more pixels the network is able to learn the signal faster. What is surprising is that more green pixels shouldn't influence our decision about whether we are seeing a cow or not, but there it is: redundant information is exploited as a stronger signal. How can we distinguish between a saturated and a true saliency ?
In this chapter, we will look at how we can find how two signals are similar and how we can normalize them.

3.2. Locality

Exploring the space of possible solutions for the MNIST[**16**] benchmark, we can reflect about, not a continuous space, but a combinatorial space of features that can be combined for a decision. From the subjective view of humans, the list of features describing the calligraphy of numbers is not very long. But in fact, they are an arbitrary number of features, as the boundary between them is continuous. For a network, that number can be at most related to its capacity, its number of neurons. For each neuron to respond distinctively, they must respond to dissimilar features. However, it is challenging to quantify the similarity of these features. To do so, we must rely on a frequentist view of similarity: correlation. In statistics, correlation refers to any statistical relationship between two random variables. Although it usually refers to how they are linearly related, we will use the term more broadly: how easy an activation value is to predict from another.

Cones in the retina have a distinct advantage with other cones nearby. When a red photon hits a cone, there is a higher probability that the cone next to it also receives a red photon. And so, they can modulate their message as a group accordingly. This statistical relationship is a result of their topological relationship: they are physically close to each other. That relationship can also be carried on into the visual cortex. In a CNN, we have that same relationship in the 2D plane of the image, but also in each layer of the convolution, since they are each the result of one filter, and so each value of that layer must have feature similarity. That information space is ordered, which is not the case with a MLP where a layer is fully connected with random weights. But there may be a topology to the abstract space of that layer, just like smell or sound. The structure is unknown whether the space emerging is disentangled or not. Can we discover a relationship that maps topologically an unordered set just like how physically close neurons in the brain share common inputs?

One classic benchmark to test the learning capabilities of a network is PermuteMNIST[**22**], which produces variants of the MNIST[**16**] dataset by shuffling the position of each pixel. This task requires the network to classify the images without relying on the spatial relationships between adjacent pixels to identify patterns in the images. Let's perform a thought experiment to explore the relationship between neurons in the same layer that share inputs and produce similar outputs, and how we can identify these "sister neurons". Instead of classifying the images without spatial features, we are tasked with unpermuting a variant of the dataset, such that the spatial features of the original image are available to the network, and that a single classification solution to MNIST[**16**] remains the same regardless of the permutation. Given an ordering O_π as a random permutation of all possible permutations of O_0, the original order of the pixels of the images composing a dataset, could we recover O_0 using some function $\mathcal{C}(x_i,x_j)$ measuring how correlated x_i and x_j are? The incoming stream looks like random vectors with no spatial relations, just like a fully connected layer, but the structure behind exists: it has been encrypted using the same permutation for all images in the dataset. Using the values of $\mathcal{C}(x_i,x_j)$ as a weighted graph, could we use this graph to reconstruct the 2D plane and solve "UnpermuteMNIST"[1]? And how large a sample of the dataset would be needed? Interestingly, our brain already has a mechanism to do exactly that. As we can see in figure 3.2, the neurons in the auditory cortex of a mouse have managed to order themselves in a few weeks of training. An ordering ability that would be quite useful for this work; but how the brain does this remains not clear[**44**].

This allows for multiple responses to the same category while, with rebalancing, only the strongest signal survives and pinpoints a specific instance of that category. This is why,

[1] Interestingly enough, most state-of-art models for permuted MNIST are LSTM-GRU type models that tries to establish long-range relationship between inputs. There is surely some cryptographer out there, who would find a solution to this problem, since the same encryption is used for every image in the stream.

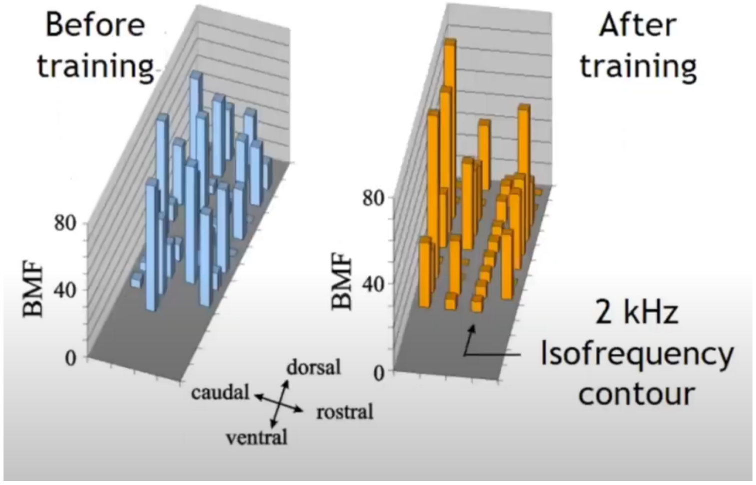

Fig. 3.2. Ordering of neuronal responses to Best Modulation Frequency in a mouse's auditory cortex before and after training to tasks in the 2kHz range. Neurons that are close will learn to respond to similar signals, which may help to better perform on those tasks[**44**]. Image also from [**44**]

from 2.3, we are neither trying to maximize nor minimize MI between variables, just rebalance their outputs. To allow for our network to make distinctions between similar signatures.

Just like the brain, we are interested in what kind of map of *locality* would produce $\mathcal{C}(X)$ on an unordered layer. A correlation map $\mathcal{C}^l$ for each layer, that is a sparse weighted graph, with ideally edge values in [0,1], so that we can simply use matrix multiplication to apply a function on this pairwise relation. One advantage we have over biological neurons for an abstract space is that we are not limited to a 2D projection of the relation. The valance of a neuron can be of size h_l.

This notion of correlation should be the degree of redundant information between two "bits", a degree of mutual information between two streams of values. Although unlike MI, which would give us $\mathcal{C}(x_i,x_j) = 1$ for mutually exclusive sets, which should not inhibit each other, we do not want a measure of predictability about whether a variable x_i is on or off given x_j; we want to determine how easy it is to predict the value x_j should neuron j be firing, given that neuron i is firing. In other words: any non-zero entropy might require a neuron to make that distinction when the moment needs it.

For example, in a sparse stream, it is easy to predict x_i from x_j. You can simply bet it will be a zero. So our coefficient should be invariant to sparsity $\implies$ $(x_i = 0,x_j = 0)$ has no influence on $\mathcal{C}$. It should also be at least invariant to the size of the vectors or at best get more accurate.

As we look for a common dependence in Z^* from x_i and x_j through correlation, we are faced with what seems to be a bit of a paradox that reveals an important question. We have already stated profusely that we can expect that redundant information will be correlated. However, would we not see all features of an object every time we see that object? All features of a cow every time we see a cow? And so, a diverse set of features recognized consistently about a class will also be correlated. How can we distinguish between redundancy in a signature (correlated signals on the same feature) if the elements of the ensemble itself activate together (correlated signals on different features)? High correlation does not imply low diversity, just as high correlation does not imply causality. With an invariance to sparsity, there is surely a difference between the two cases now that we eliminated mutual zero entries.

For a signature to be rich, it must be informative. We come to the surprising conclusion that we must consider carefully about optimizing for invariance: the model needs a certain type of variance. Not invariance, but variance in group invariance for complementary values, and invariance in group invariance for redundant values. Invariance of S for y, but variance in S.

- invariance : the variables vary together, they remain invariant w.r.t each other. (redundancy)
- variance : the variables vary to one another, they vary w.r.t each other. (diversity)
- group variance : the variables vary w.r.t each other within a group. (they are not a group, this is just variance)
- group invariance : the variables vary together as a group. (this defines a group)
- variance in group invariance : the variables vary w.r.t each other within a group AND together as a group. (they are partially correlated)
- invariance in group invariance : the variables vary together within a group AND together as a group. (this is just group invariance)

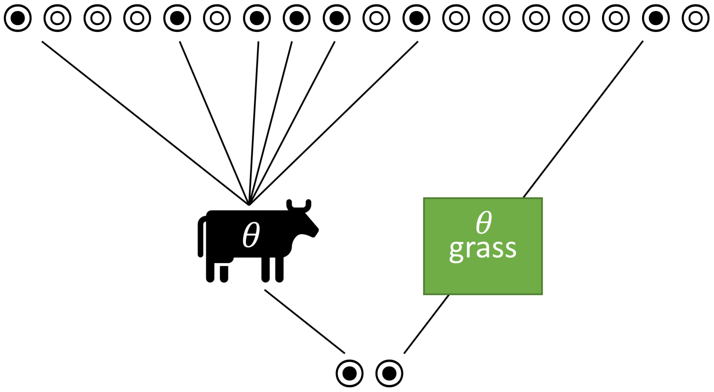

Fig. 3.3. How we hope features of a cow and grass would be aggregated in a network. Variance in group invariance for complementary values, and invariance in group invariance for redundant values.

Let's look at the rudimentary drawing of a network's view of a cow on grass. The features of a cow are correlated, because they appear together as an ensemble of the feature `cow`. The apparent correlation paradox is not a paradox after all, as it already solved by the hierarchisation of features in a deep neural net.

When we see a cow, we in fact never really see all the features of a cow, just like when we see a cube we don't see all its faces. Because the object is complex, that is what makes it salient. The variables of a cow vary w.r.t each other even though they like appear together. But when we see simple features of grass, they nearly always appear together because they are few. Signatures of grass might even slip into the signature of a cow: $s_{grass} \subseteq s_{cow}$. However, should the message `grass` be important and carried on to the next layer, it will still be correlated with `cow`. There may be just a slight statistical difference, yes. But we can exploit this difference for saliency. Just like cow and grass, we aim to exploit this difference for digit features and oracle. In our particular case of OracleMNIST, the oracle will always find its way into neurons of every layer up to the last. The model will then have to decide between oracle or digit features in a conflict.

3.3. Inhibition

Lateral or local inhibition (LI), also known as surround suppression or non-classical receptive field (NCRF)[2], is a well-known integral feature of biological neural circuitry that is extensively researched in neuroscience. LI refers to the phenomenon where a neuron's activity reduces the activity of its neighbors in a lateral or surrounding region of the nervous system. When a neuron has an inhibitory connection to another neuron, it will suppress its activation or effect negatively its probability to activate. Unlike artificial neural networks (ANNs) where neurons are usually only connected forward to neurons in the next layer, biological neurons are often connected laterally to suppress the activity of their neighbors in the same layer or region. When neurons with similar inputs in the same layer or region are connected by lateral inhibition, they compete with one another to become active, creating a winner-take-all effect where a few neurons dominate the output of the circuit. This competitive aspect of lateral inhibition has important implications for neural processing as it enhances the ability of the nervous system to extract relevant information from noisy or ambiguous inputs [**12**]. For example, LI can enhance contrast, making it easier for the brain to detect edges and boundaries[**29, 37**]." It is present in the sensory systems of nearly all animals, and despite its importance, few machine learning (ML) papers and models consider

[2] Non-classical since neurons respond differently to signals outside of their classical receptive field (signals coming through its input from previous layers). All synonyms composed of {Local, Lateral and Surround}×{Inhibition, Suppression} are encountered in neuroscience literature. The *lateral* aspect is important as ANN layers are usually only connected forward. We prefer to keep our notion of *Local* to reflect the topology of the abstract space of a layer.

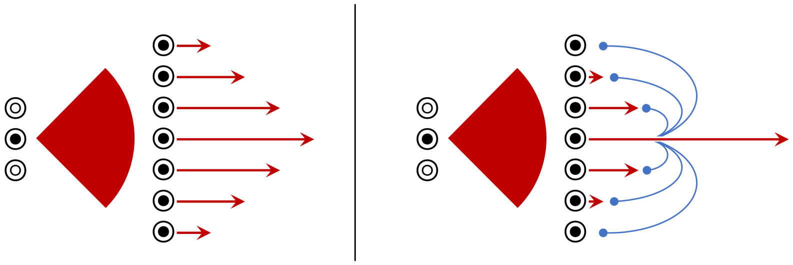

Fig. 3.4. Selection mechanism of signal interaction from lateral inhibition. With all neurons connected to the same feature in their input, they respond according to their weight to it (left). With lateral inhibition (right), the strongest output suppresses its neighbours.

lateral inhibition. One possible explanation for this, is that ML researchers rely on the fact that the weights of a networks can be negative, and so that the optimization process will find a similar solution on its own.

Although neurons in our visual cortex share similar orientation preferences, the response of nearby neurons is very sparse. Inhibition produces a non-linear effect not only in the response of the neurons but also in the definition of the neuron's receptive field[**45**]. There is no doubt of the importance of local inhibition for the somatosensory information space of an animal. It is also easier to visualize how it refines images and sounds than it is to guess at all the potential benefits deeper into cognition[**13**]. With this structural property, downstream neurons are more likely to only activate when they receive a strong and consistent signal, therefore improving signal-to-noise ratio. It does more than just serve as noise reduction of competing neighbours, however. It could also be a selection mechanism just like attention. A strong signal aimed at receiver might get caught by many of its neighbours, who would in turn propagate it to their receivers. Without the regulation of the intended receiver to its neighbours, its own message could get lost in the cacophony downstream. Even if its message is received it would be blurred, and so this sharpening effect is also important for an abstract information space. This means that neurons deep in the hierarchy should only fire when increasingly more complex signatures are sampled, so that semantically similar concepts might not be confused. We can take camouflage as an example of adversarial attack in nature, which would be pressured to be ever closer to a certain signature.

Local inhibition makes neurons more sensitive to locally varying stimulus as locally uniform stimulus will self-suppress. This increases the network's sensitivity to variance in invariant group(2.3). It likewise reduces the network's sensitivity to invariance in invariant group, while still detecting the group as the magnitude is only renormalized.
It is a mechanism that is necessarily needed at inference time, and we cannot place our

hope that some loss or optimization for minimization or maximization of MI will replace it. (It would also have to double the number of layers.)

Lateral Inhibition was first introduced in for neural nets in AlexNet[**32**] under the name Local Response Normalization as a way of normalizing the unbound nature of ReLu. LRN square-normalizes values in a same channel of a CNN based on their proximity.

$$b_c = a_c \left(k + \frac{\alpha}{n} \sum a_{c'}^2 \right)^{-\beta} \tag{3.3.1}$$

The constants k, α, β, and n are hyper-parameters, with α and β for inhibition and n the neighbourhood.

By providing contrast enhancement, it pre-encodes a certain type of information : information about information. And that relational information can be revealed just by looking at the highest values. The network can detect contours that can be used by its attention mechanism[**29, 12**], a pre-computation of what would otherwise have to be computed every time it is needed downstream. But how it would serve us against placing too much weight on one feature, spurious or not, is by filtering out volume. Because it reveals the most confident expert in an ensemble of experts. Which becomes a mixture of competing experts; many voices on that are similar with slight distinctions, with only one winner. Neurons with the same opinion compete for accuracy. And by selecting an expert, rather than averaging the ensemble, the model gains increased precision using multiple similar representations. And therefore suppress redundant information.

Since the solution to how neurons self-regulate and balance their outputs is already found nature; let's look again there to ponder how we could rebalance our outputs. Biological Neurons branch out laterally inside their layer and are able to send inhibitory signals to sisters in close proximity. This works well because they depend on how many discrete signals over time they send to convey the strength of their signals. Advantage goes to neurons who fire first and consistently, as inhibited ones must take the time re-accumulate a sufficient potential to fire every time they receive an inhibitory signal.

The process is dynamic as inhibited neurons may still be able to fire their own signal, lowering in turn the inhibitor. An inhibited neuron will fire less and in turn inhibit less the inhibitor that will fire more and so on, until an equilibrium is reached. This equilibrium is usually simulated in a non-dynamic process, for the last layer of an ANN, with the Softmax function $\sigma(x_i) = \frac{e^{x_i}}{\sum_{j=1}^{k} e^{x_j}}$ as it has the nice property that it always sums to 1 and selects the strongest output. However, Softmax does not conserve $||s||$ since its own magnitude is always 1[3]. The value of a signature is lost if the activation is invariant to its magnitude. We could multiply $\sigma(x_i)$ by $x_{\max}$, the maximum incoming value, to conserve $||s||$ but is that correct?

True cardinality of S is usually unknown, since $\mathbf{z}^*$ is unknown. Let's go back to the oracle problem in our model, where subsampling in our case means $|S|$ is of size one. The sampling θ is optimized so as to select only one feature z, and the representation space of the sampling layer collapses to maximum effective sparsity[4], also of cardinality 1. This is the poorest quality of information subsampling (or close, it's not zero). In the presence of an oracle, any two neurons connected to it might be perfectly correlated. For them not to be correlated, they would have to be connected to other features in the signature they sample, with positive or negative weights, "to try to hide the oracle". If those features are noise, the oracle is suppressed. If those features are digit features however... digit features are associated with the oracle. Since two perfectly correlated neurons might as well be one neuron and subsequent neurons downstream would not truly have a distributed representation by having a different receptive field which includes a linear combination of them. Lets therefore consider correlated neurons as a node. A node is a group of neurons under a certain valence: all the neurons in the node are topologically close. First for simplicity, we will consider a disentangled representation, where a neuron belongs to only one node. We expect that all neurons in a node respond to $\mathcal{C} = \psi(\Delta_m^{-1})$: a response proportional to a function of the proximity to the neuron with maximum response in the node.

Neurons in node A are perfectly correlated s.t. they nearly always output the same values. So are neurons in node B. Node A and B are correlated by a factor $\mathcal{C}_{AB}$. Let's consider for the moment that $\mathcal{C}_{AB} = 1$. With a signature S with unit value that is duplicated and sent to A and B s.t. activations of A and B are bound by 1: the activations lie in the closed unit interval $\sigma\mathbb{R} \to$ [0,1]. So the total output of A and $B \leq 2$. There is obvious redundancy as the message is duplicated. However since the first neuron to respond to it cancels out the response of its neighbour in that moment, the redundant messages are filtered out and the sum of the message remains the same.

Node A and B share signatures S for a feature, $s_A \cap s_B$. Depending on θ_A and θ_B and the context of other source signals, node A and node B will fire with probability $P(\sigma(A))$

[3] For example if k correlated neurons fire with an equal value of `0.5`, the output of the neurons relaying that message should be `0.5`, not `1.0`

[4] One reason why sparsity is not sufficient even if desirable

and $P(\sigma(B))$ s.t. they are mutually exclusive as a pair of binary streams.

$$P(\sigma(A)|S) = 1 - P(\sigma(B)|S) \tag{3.3.2}$$

However, since A and B are different nodes but are correlated by $\mathcal{C}_{AB} = 1$, they must have a conflicting message. What is the probability that A is the intended receiver of S? The probability that A is not an outlier is proportional to $\frac{|A|}{|A|+|B|}$: how many neurons share opinion A vs. how many neurons share opinion A or B. Which implies that $||S||$ is conserved iff $\sigma(A(S)) = \sigma(B(S))$ and that the sum of activations of correlated neurons $\leq \max\{||s_A||,||s_B||\}$. A is modulated by the ratio of its signal in the global signature S. And so is B.

$$\iota\left(\sigma(A)|\sigma(B)\right) = \sigma(A) \times \left(\frac{\sigma(A)}{\sigma(A) + \sigma(B)}\right) \tag{3.3.3}$$

With two important properties :

- Sum of correlated activations $\leq$ maximum output response to S.
- Lower global magnitude on conflicting outputs of a node.

For a total output to be conserved, a single neuron, or a single neuron in a set of neurons, must activate: perfect diversity. Or the activation values of a group of neurons in a set must activate with the same value: perfect redundancy. For example, if one or a few neurons in a set don't activate, all others activate with the same value, their output is preserved. This redundancy translates into a specific resilience to noise, where neurons will sometimes fail to activate. Which means that for a strong and consistent response neurons must encode distinct signals. Responses that disagree but should agree to a same feature are penalized; as they must not be responding to their intended feature or are receiving noise. Constraints on diversity and redundancy respectively. Distinction is information, and the network is encouraged to make that distinctions since the inhibition $\iota(X)$ is considered in the backpropagation : $\theta_{A\cup B}$ is optimized to minimize the intersection of θ_A et θ_B. We can use $\iota\left(\sigma(A)|\sigma(B)\right)$ to any dimension. Specifically, scaling each element in a vector by its proportion to the sum of all elements in the vector (as long as all entries are non-negatives)[5]:

$$x'_i = x_i \times \frac{x_i}{\sum x_j} \tag{3.3.4}$$

Which can easily computed using element-wise squaring ($\boldsymbol{.}^{\odot 2}$). We define Proportional Inhibition of vector $\boldsymbol{x}$ as :

$$\iota(\boldsymbol{x}) := \boldsymbol{x}^{\odot 2} / \sum x_i \tag{3.3.5}$$

This is a form of divisive normalization (like LRN[**32**]), in the sense that it reduces the

[5] We only conider here non-negative activation functions. And, since $\lim_{a\to 0} \frac{a\times b}{a} = 0$ we consider $\frac{0}{0} = 0$, should all entries of $\boldsymbol{x} = 0$.

activity of neurons by dividing the output of each neuron by a normalization factor that depends on the activity of the other neurons; with the important property that the sum is conserved, $\sum \boldsymbol{x}_i \leq \sum \boldsymbol{x}_i$, relative to how different in scale the entries are. Only when a single entry is non-zero, do we have $\sum \boldsymbol{x}_i = \sum \boldsymbol{x}_i$. We constrain saliency through distinction. With 3.3.5 all entries are equally redistributed, but we can consider the relationship of locality by including a weighted adjacency matrix of a directed graph $\mathcal{C}$. We can simply use the Hadamard division $\oslash$ to compute this weighted inhibition :

$$\iota(\boldsymbol{x},\mathcal{C}) \coloneqq \boldsymbol{x}^{\odot 2} \oslash \mathcal{C}\boldsymbol{x} \tag{3.3.6}$$

Where activation i inhibits activation j by factor c_{ij}.

3.4. The Value of Event $\boldsymbol{E}$

With an inhibition function in hand, we can finally address *locality* properly and choose a similarity coefficient or "correlation map[6]" $\mathcal{C}$. But first, let's list the properties for an ideal function that we arrived at in section 3.2 on locality :

(1) Predict simple relations
(2) Outputs in closed unit interval, $f : \mathbb{R}^{2\times n} \to [0,1]$
(3) Invariant to pairs of zero entries
(4) Invariant to isotropic transformations : $\mathcal{C}(X,Y) = C(\alpha X,\beta Y)$ for $\alpha, \beta \in \mathbb{R}^+$

The costly part: the result of every gradient step will impact the distribution of the activations to one that cannot be predicted. Will a little bit less of this feature means it will be less correlated than this one and more to that one ? Which means we have to reevaluate $\mathcal{C}$ empirically on the dataset frequently. Experimentally, if the computational cost is too high, either training will be unmanageable or we will be forced to evaluate on small batches and obtain poor statistics, not knowing why we failed. $\mathcal{C}$ should be the result of simple or efficient computation, like matrix multiplication.
For example, estimating Mutual Information or KL divergence on two streams of continuous variable is done using k-nearest neighbours with a cost of $\mathcal{O}(h^2 n\sqrt{kn})$[**31**] in the best case, which would be much too expensive ($\sim$ 1000× for a simple network on MNIST[**16**]). MI would also violate our desire to detect simple relations and invariance to zero entries. But more importantly, MI and entropy would measure the average predictability of whether a

[6] The matrix is named so in our code : `cmap`

neuron fires or not[7]; not how similar their values are when they both fire together. We also do no want to inhibit mutually exclusive pairs.

When we are thinking about how two neurons i and j will inhibit each other, we must not forget to consider the value of their message. If a neuron is constantly firing, its value is not very surprising compared to one that rarely fires. This important *asymmetry* is the final criteria for our similarity coefficient. Simply because the value of each neuron's role is unique. An equality would result in all timid neurons constantly be suppressed by the loud one in the layer. If neuron i usually output values much lower than j, $\sigma(x_i) << \sigma(x_j)$ s.t. $c \times \sigma(x_i) < \sigma(x_j)$, then its rare message of strength c will be suppressed as well. The information value of an event E is usually defined as that surprisal:

$$I(E) = \log_2(\frac{1}{P(E)}) \tag{3.4.1}$$

For a discreet stream of bits, we could easily estimate how surprising x_j is to x_i, and factor in the number of samples n in a change of base to get a coefficient of inhibition :

$$\mathcal{C}_{\{0;1\}}(x_i,x_j) = 1 + \log_n(P(x_j|x_i)) \tag{3.4.2}$$

How *unsurprising* x_j is to x_i is how much neuron i suppresses the activation of neuron j. For two discrete neurons, only one outcome is important for mutual inhibition : when they are both firing. We are trying to compute the probability that "we are trying to say the same thing": but I have more information. Unfortunately, the neurons in our network live in $\mathbb{R}^+$. If we were not to account for the low value of "chatter" in the network, noise, most neurons would be estimated to have good degree of correlation. Because of how sigmoid and ReLu behave and that the distribution they output is unknown, we cannot determine a change of base that would allow us to use a form of $I(E)$. However, even without the benefits of log space, we can still take inspiration from 3.4.1 and define :

$$\mathcal{C}(\mathbf{A}) := \frac{\mathbf{E}[a_i \cdot a_j]}{\mathbf{E}[a_i]} \tag{3.4.3}$$

The expected "information content" of measure Y given X compared to the expected information content of a measure X. A ratio of information content that since they are on the same number of events, is in the range [0,1] if the activations are in [0,1]. This dot product based coefficient is similar to cosine similarity but respects most of our axioms. It is not perfect, as it not invariant to isotropic transformations. It can also easily be computed with matrix multiplication:

$$\hat{\mathcal{C}}(\mathbf{A}) = \mathbf{A}\mathbf{A}^t / \mathbf{A}\mathbf{1}_{n\times 1} \tag{3.4.4}$$

[7] As mentioned in 3.2, the entropy of a sparse sequence is low since most values are 0

With $\mathbf{A}$ the matrix $h \times n$ of all the activations of a layer l of size h. In our experiment, we will compare $\hat{\mathcal{C}}$ with Pearson correlation, which also has the nice property of being invariant to adding any constant to all elements: filtering out the lowest values of the estimation.

3.5. Proposition

To incorporate Local Inhibition in a neural network, we propose to modify the training and architecture of standard neural network in those two parts respectively: locality and inhibition. Looking at the standard training algorithm of a neural network (next page): Firstly, during the inference phase, lateral inhibition is applied in the forward pass of each layer (lines 8 to 10). This inhibitory mechanism is applied to each neuron in the layer by normalizing it with respect to the activations of all neurons according to an adjacency matrix C. Secondly for locality, in the learning phase after each gradient step, the algorithm updates its pairwise correlation matrix C(lines 28 to 34) computed from all the activations of each layer over a large sample of the dataset. Note that there is no inhibition in the evaluation of locality, as we are evaluating their predicted activations pre-inhibition. More important to note is that since we are adding this loop inside the loop of training batches, this is where the computation is most expensive. The size of the sample directly influence the accuracy of the statistics of C. The algorithm iteratively updates the weights and biases using backpropagation with the standard gradient descent approach normally.

Algorithm 1 Neural network training with Lateral Inhibition

Require: input data X, label Y, number of hidden layers L, weight matrices W^l, adjacency matrices C^l, bias vectors b^l, non-negative activation function f, learning rate η

Ensure: trained weight matrices W^l and bias vectors b^l

```
Initialize weight matrices $W^l$ and bias vectors $b^l$ randomly with small values
while not converged do
    for each batch $(X_b, Y_b)$ in $X$ do
        $Z^1 \leftarrow X_b$
        for $l = 1$ to $L$ do                                    ▷ Forward pass
            $A^l \leftarrow Z^{l-1} W^l + b^l$
            $Z^l \leftarrow f(A^l)$
            for all $z \in Z^1$ do
                $z_i \leftarrow z_i \times \frac{z_i}{Z^l {C_i}^l}$                  ▷ Lateral inhibition
            end for
        end for
        $A^{L+1} \leftarrow Z^L W^{L+1} + b^{L+1}$
        $\hat{Y_b} \leftarrow f(A^{L+1})$                        ▷ Output of the neural network
        $d\hat{Y_b} \leftarrow \nabla_{\hat{Y_b}} \mathcal{L}(Y_b,\hat{Y_b}) \odot f'(A^{L+1})$   ▷ Gradient of loss w.r.t output of network
        $dW^{L+1} \leftarrow Z^{L\intercal} d\hat{Y_b}$                   ▷ Weight gradient
        $db^{L+1} \leftarrow \sum_{i=1}^{m} d\hat{Y}_{bi}$,                 ▷ Bias gradient
        $dZ^L \leftarrow d\hat{Y_b} W^{L+1\intercal}$                    ▷ Hidden layer gradients
        for $l = L$ to $1$ do
            $dA^l \leftarrow dZ^l \odot f'(A^l)$                 ▷ Calculate gradients at layer $l$
            $dW^l \leftarrow Z^{l-1\intercal} dA^l$                      ▷ Calculate weight gradient
            $db^l \leftarrow \sum_{i=1}^{n} dA_i^l$                      ▷ Calculate bias gradient
            $dZ^{l-1} \leftarrow dA^l W^{l\intercal}$                     ▷ Calculate hidden layer gradients
        end for
        for $l = 1$ to $L + 1$ do                                ▷ For all layers
            $W^l \leftarrow W^l - \eta dW^l$                     ▷ Update weights using gradients
            $b^l \leftarrow b^l - \eta db^l$
        end for
        $Z^1 \leftarrow X' \in X$                    ▷ Feed a large sample of the dataset to the network
        for all layers do                                        ▷ Forward pass
            $A^l \leftarrow Z^{l-1} W^l + b^l$
            $Z^l \leftarrow f(A^l)$
            $H_{b,i,j} \leftarrow \sum_k {Z^l}_{b,i,k} {Z^l}_{b,j,k}$            ▷ Collect $Z^\intercal Z$ for each sample
            ${C^l}_{ij} = \frac{\sum_k H_{ijk}}{\sum_k (A_{ik}+\epsilon)}$       ▷ Compute pairwise correlation matrix $C$ for layer $l$
        end for
    end for
end while
```

3.6. Related Work

3.6.1. Normalization

- **Batch Normalization.** Although LRN[**32**] is not used anymore, it opened the idea that outputs should be rebalanced. `ReLu` was chosen then because saturating activation function like `sigmoid` are harder to train on deep networks as their gradient can vanish. It is Batch Normalization[**28**] however that opened the whole field of research on the topic. It improved results on ImageNet[**15**] by a significant margin using only a few epochs. BN is a trainable layer aimed at addressing the issues of Internal Covariate Shift: how each neuron's input distribution changes at every step[8]. The covariate shift forces training to be done with only small training steps, meaning slow. By normalizing the input over a mini-batch in the statistical sense, rotating and scaling and centring around zero, the neurons see consistent inputs that look less like random noise. It stabilizes the distribution of the activations resulting in smoother the loss surface so their inputs can be learned.

3.6.2. Regularization

Regularization method go in the other direction by rebalancing the gradient. Here we present a few methods aimed at learning decorrelated representations.

- **Dropout**[**46**]**:** The most prominent method aimed at promoting representational diversity is Dropout. An ISS strategy that randomly only selects a subset of inputs. By masking neurons randomly, it imposes a simple constrain on the network : no single neuron can be too important. It is intended at promoting both diversity and redundancy as the network must learn to work with a random significant portion of neurons absent (usually 30%-50%). With random self-inhibition, comes random normalization. But also random regularization as with outputs zeroed out, the gradient of these neurons is also zeroed out. Since each random subnetwork is trained for the task, it is similar to bagging as it trains an ensemble of models that can later be used as an averaged ensemble of experts. A certainly effective technique that fell out popularity, however, when it was discovered that Batch Norm is so effective that we can forgo this method in training our models. Its very goal of packing the network with representations, however, is not compatible with sparsity without an information bottleneck. The great robustness to missing values provided by sparsity also ensures it is ineffective at dropping a spurious feature. By subjecting the network to this powerful neural death noise, the model gains robustness to it. But as predicted by the infoMax theory, it forces the model to choose between learning redundant or

[8] the same problem we face to evaluate $\mathcal{C}$

diverse features. And in the case of a powerful predictor, like a shortcut spurious feature, the model will hold on to it until it is the last to go.

- **Orthogonality:** To address the redundancy problem in the representations of neural networks, some methods are aimed at optimizing for the angular diversity of the weight vectors and filters. However, since orthogonality is not hard to obtain in high dimensions, Minimum Hperspherical Energy[**35**] and Maximum Minimum Angle[**50**] try also to learn weights so that their distances on the hypersphere are also maximized. The representations are therefore subsequently decorrelated and useful for sparse coding. They also have the advantage that orthogonal regularization can be combined with almost any other method by only adding its loss to the penalty.
- **SLNID :** Sparse Coding Through Local Neural Inhibition[**3**] was inspired similarly to our work by LI. It was aimed for continual learning by promoting organized sparsity of diverse representations. By pair the pairing the activation of each neuron with the training example, each active neuron receives a penalty from every other active neuron that corresponds to that other neuron's activation magnitude. In other words, if a neuron fires, with a high activation value, for a given example, it will suppress firing of other neurons for that same example. As a loss, however, it is actually the gradient that incurs local inhibition and not the activations. It therefore plays no role at inference time does not gain the benefits associated with local inhibition as a structural feature.
- **Spectral Decoupling**[9] [**39**] : This work proposed a theoretical explanation for the phenomenon behind Gradient Starvation that plays a vicious loop in shortcut learning. By focusing on a single feature, the network becomes so confident that this becomes the only source for its output, resulting in a learning signal directed back only at it. Other features are effectively starved of gradients, never to be learned. The suggested Spectral Decoupling is the simplest of loss: penalize its confidence by the square of its output. SD outperforms IRM[**5**] and REx[**33**] on ColoredMNIST without the need to consider an environment or distribution.

We end this section with the note that none of these methods could perform better than 11.5% accuracy on the Oracle-MNIST benchmark (By setting a loss λ so high it would not learn). As it leaves us wondering if maybe we took a bite out of something too big to chew.[10]

[9] Shoutout and congratulations to the authors, as their paper and method proved to be the most useful to our work.

[10] Not to worry the reader too much, as in the experimental results we managed to show a positive impact for a network.

3.7. Last Considerations : Sampling

Weight management : At every gradient step, the optimization will try to place weights for best accuracy. Since our inhibition function is part of the forward computational graph and differentiable, it is also considered in the back propagation steps. We can already anticipate that strong (maybe spurious) features will hack their way into importance by simply attributing a lot of weight to that one feature faster than we can recompute $\mathcal{C}$. We will have to consider putting a maximum on them, weight clipping, to prevent this. Weight clipping, however, doesn't always play nice with backpropagation as it can assign weight just so it can subsequently be erased. Which is a likely behaviour given how strong a signal the oracle is. Our best bet will be to use Weight Decay with our method.

Activations: Since our similarity coefficient is dot product based but not normalized by the square of the norms, it only stays in the range [0,1] if its inputs are in that range. We are not sure what it would mean semantically if we normalized `ReLu` activations to then evaluate their similarity. And so we will only use `sigmoid` in all our experiments. Considering also how `ReLu` raced headlong in 3.1, it seems like a wise choice.

Chapter 4

Experiment

4.1. Experimental Results

4.1.1. Oracle MNIST

To test if all these considerations can be leveraged for positive impact on a network's reliance on easy solutions, we task it to learn a dataset in the presence of a perfect spurious correlation signal: an *oracle*. The oracle predicts the target perfectly. With this benchmark, Oracle-MNIST, we train a model on a simple task normally with the added "benefit" of providing the target answer in the input. The oracle, consisting of c neurons, perfectly predicts the class with 100% accuracy during training and is fed directly into the network in a dedicated layer, which is then concatenated with a hidden layer in the network. At test time, the oracle is randomly permuted between all classes except the correct one (0% accuracy). Success in this benchmark is any result above 11.5% (Outputting 1, the most common digit in the dataset). The aim of this experiment is to answer a simple question: Can our network, enhanced with Local Inhibition, still learn features in the presence of an oracle?

Based on the fact that all methods have 0% accuracy on this benchmark, it may be considered an extreme test case. Since accuracy is affected by the complexity of the task, the level of noise, uncertainty in the data, or the difficulty in solving the task, one might wonder if it is too hard, unreasonable and pointless. However, it tests directly how the network allocates its resources to spurious features. We can even look at how much weight exactly is connected to the oracle, giving us a direct insight into how the model learns. The testing method is so simple that it can be applied to almost any supervised learning dataset. Moreover, we can draw conclusions easily by comparing Oracle-MNIST with the MNIST dataset; the original task is not complex, there is no noise or uncertainty in the data, and it is very easy to solve. We are simply adding 1 active pixel to the image, a one-hot vector of 10 bits. The challenge lies in only one factor: how fast the model learns those pixels.

With the model trying to learn a way out of the inhibition map, we knew from figure 3.1 that we had no choice but to anticipate a very rapid learning of the oracle. How many batches before the network falls for it ? This means we have a computation dilemma between time and accuracy, as calibrating $\mathcal{C}$ on all 60 000 images was going to be too expensive. The final choice was to do an update with probability `p(update)` on every gradient step and how many samples used for the estimation. Most of our experiments were done using these hyperparameters that we haven't felt the need to tune.

Hyperparameters:

- `p(update)=0.3`
- `p(Eye)= 0.3` (discussed later)
- **`calibration batch size= 2048`** ($\sim$ 10× computation time)

- All Oracles experiments were done using a simple model composed of a CNN layer with 32 filters and a single hidden fully connected layer of 728 neurons.
- This baseline scored 97.5% on the 10 000 images of the test set in fewer than 10 epochs.
- All Oracles experiments were done using an oracle of size 10× 10 connected to 100 neurons concatenated to the 728 digit neurons. After which inhibition is applied.
- All Oracle experiments we performed on a P100 with a running time of about 4 minutes per epoch for the OracleMNIST benchmark.
- Due to computational constrains, each experiment instance was run no more than 3 times.
- All experiments, statistics and hyparameters have been logged in WandB[**9**] which has been used to generate the plots.
- The code is available publicly on my Github page.

Looking at the initial results in figure 4.1, we get the impression that our efforts end in failure. The only positive results obtained for a while were these. Where the network "survives" at least for few epochs. However, looking closely at the chart, we can notice something curious. That there is a period where the accuracy is between 11.5% and 0%. Other classes guessed could explain why the results were not always 11.5%. I 1s are the most numerous, there might be a digit that represents around 8% of the dataset (5s). But they are values around 4% in that chart. Which meant the network must be choosing between its guessing digit and the misleading oracle at a certain moment in its training.

Pulling out the results of correct predictions actually showed it had learned to discriminate between 3 and 4 classes even in the presence of the oracle! This can only be explained by the fact that the network managed to learn true features of datasets. For it found enough of a boundary classifies four different digits. A victory in itself.

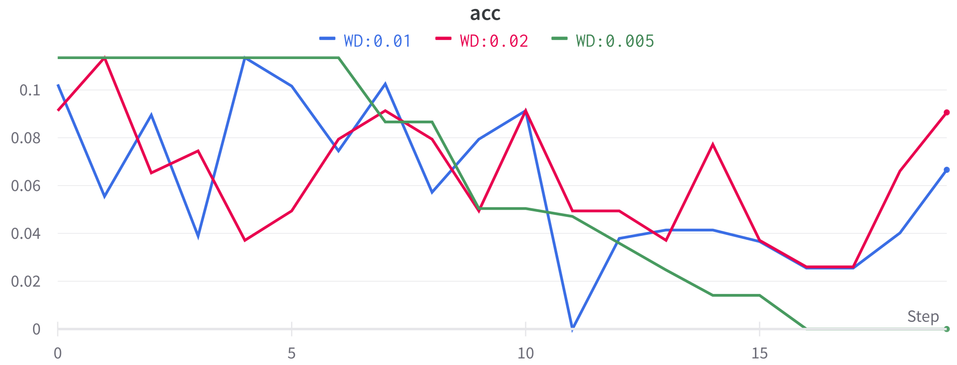

Fig. 4.1. Learning progression of initial results using lateral inhibition and weight decay. The oracle still dominates the learning. Accuracy oscillate between 0% and 11.5%, showing some positive effect but not enough to beat the baseline (outputting only 1s).

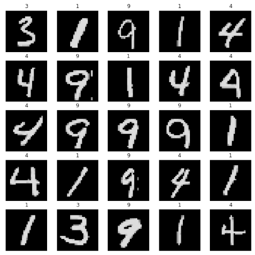

Fig. 4.2. Sample of the predicted classes in the experiment of figure 4.1. The model is trying to predict one of four classes that contradict the oracle; showing experimental success: it has learned some digit features in the image.

The gravitational pull of the oracle is just too overwhelming, however, and it ends taking all the confidence of the model. Our intuition in the moment that the inhibition network was also too strong and might be preventing a learning signal; as it is calculated in the gradient. Even with the oracle signal rebalanced to 1, the network can still assign a lot of weights to it or minimize neighbours if we impose a L_1 or L_2 penalty. To bootstrap the learning, a random Identity map ($\mathcal{C} = I$) was to be sent to replace the Correlation one with probability `p(Eye)= 0.25` every $\mathcal{C}$ was to be updated. So that some learning would flow to desired feature that we know are there now. Like flashing some correct answers.

Looking at the resulsts with bootstraping in figure 4.3, we see that the network is able to learn ever more features as it is trained. But the weights assigned to the oracle features are still too high, and overwhelm the contributions of other features. To address this issue, we tested two techniques that aim to limit the amount of weight the network assigns to features.

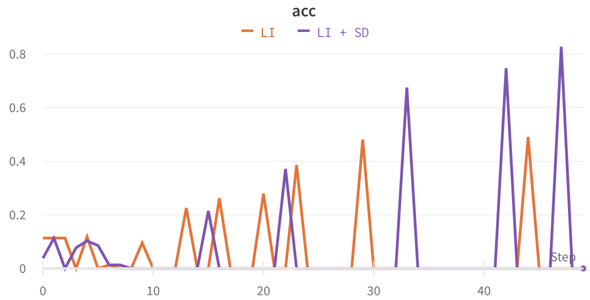

Fig. 4.3. Results with flashing Identity. In this experiment, we randomly swap the correlation matrix $\mathcal{C}$ with the identity with probability `p(eye)=0.3`. The result is that for brief moments, the network learns without inhibition. Local Inhibition with Spectral Decoupling yields the best results.

Weight Decay was found to be ineffective in helping Lateral Inhibition. We hypothesized that Spectral Decomposition (SD) could be more effective as it is similar to Weight Decay as a regularizer but also comparable to Lateral Inhibition in that both techniques limit the amount of weight the network can put on a feature by addressing gradient starvation, i.e. allowing the gradient to flow to other features, but also because it is applied at the output layer, where there is no inhibition. Alone SD failed the test, but combined with Lateral Inhibition it improved its learning.
We can take a peek during training at a map that shows just the oracles during training. We can see that the network learns to ignore redundant oracles as most show no correlation towards the end of the training cycle (sparse activation).

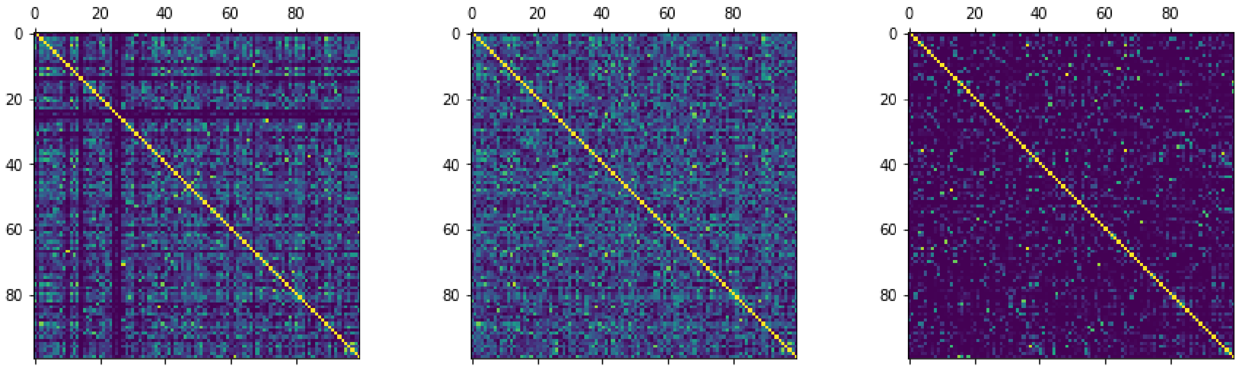

Fig. 4.4. Progression of correlation of every oracle neurons over the epochs : `5-> 25-> 50`. Over training with inhibition, the network learns sparse decorrelated activations as the number of features is very low (single feature for each class).

With this poor result, however, it leaves us to question how both sets of neurons are interacting. What if we consider that we know which neurons belong to nodes A or B ?

This time, we apply inhibition separately to both vectors before applying it to the next layer.

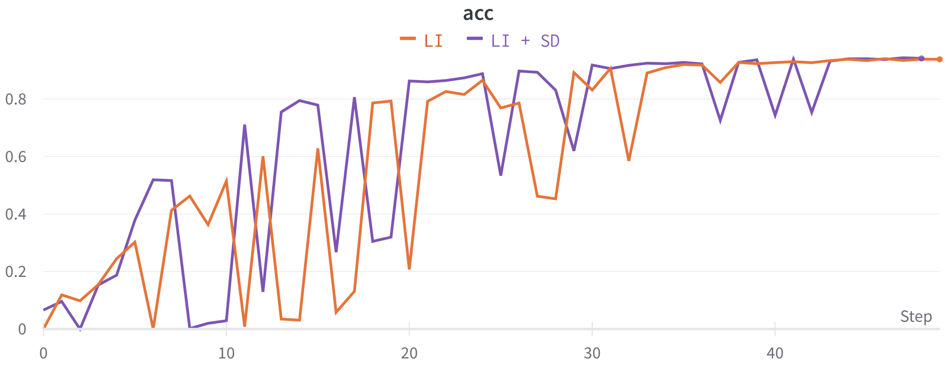

Fig. 4.5. Applying inhibition separately to the oracle input layer and the image input layer before the two layers are then connected into the final layer with inhibition. This help our network with it's topology of locality, showing that $\mathcal{C}$ does not getting accurate enough for the benchmark. Accuracy reaches the same level if no oracle was present as the networks learns to rely on the diverse features of the digit images.

With this result in hand, we can see that the evaluation of similarity was so not good enough to separate both signals in the original experiment. Looking at cosine and Pearson yields worst results and oscillation between 0% and 11.5% for the original experiment. More computation by increasing the evaluation frequency did not yield better results until we exhausted all the computation we could afford. Which seems to indicate that an inhibition mechanism, the map itself, cannot evolve in parallel with training as a similarity score cannot gather sufficient statistics. Or simply that our chosen function is wrong.

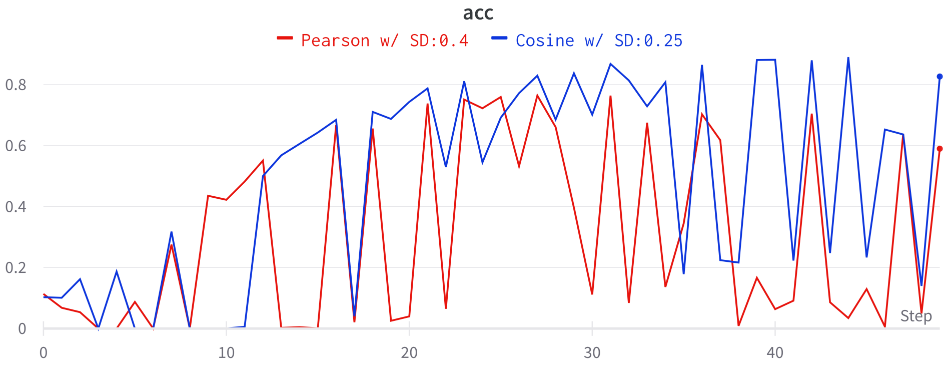

Fig. 4.6. Applying inhibition separately, but this time using Pearson and Cosine similarity coefficients for $\mathcal{C}$, showing that many notions of similarity could be used for locality.

4.2. Drawbacks

Although we have seen some positive impact on the internal representation, they are still two big drawbacks before it can be useable. It is hard to know if we could get better accuracy with a more accurate as we are limited by computation and solution time.
Computation time exacerbate hyperparameter tuning which is in our case what determines how stable the method is. And this instability is exactly that second-biggest drawback. Without a smooth curve, we have no guarantee of accuracy, and without a guarantee of accuracy sadly... there cannot be a notion of *Robust Accuracy*

Whatever the method, we can always find examples in which it will fail[**17**]. If I have shown one thing, is that it's not hard to give a network a task it is doomed to fail. Even with perfect accuracy on an oracle of size 100, all we would have to do is present them sequentially to bypass any similarity measures, and so in turn there would be no inhibition. Which the biggest risk of learning variance : learning noise.

Chapter 5

Discussion

5.1. Future Work

The results in this work left us with a lot of room to explore beyond. It seems we are left with more questions than answers. The main item to address for improvement is its biggest drawback. LI is easily computed at inference time, but despite our best efforts, we still haven't found a way to compute this correlation map in an online fashion, or another coefficient that can be computed online. If only the covariate shift could be estimated in the gradient ...An online metric would lend great weight to the method presented in this work as it would greatly reduce computation at training time and accuracy fluctuations, and so robust accuracy. We could perhaps use the correlation estimation for backpropagation; after all, the learning algorithm is the most important aspect of a neural network. Maybe we could redistribute the gradient towards neurons that yielded surprising information. Which raises that question: How should we distribute the gradient among correlated outputs ? As fully connected layers where the type targeted with this work, we could look to see how LI impacts other important NN structures. But, given our hope that local inhibition improves ISS, the most immediate future directions would to explore how LI would benefit other domains requiring internal diversity like ood generalization, adversarial robustness or continual learning. We are already working on combining the information maximization properties of constructive autoencoder with local inhibition and are hopeful of the results. How well would LI marry with orthogonalization techniques? We could in that sense add another metric in the computation of $\mathcal{C}$ like IOU $(\frac{\cap}{\cup})$ or Wasstersein distance of θ. And since any similarity coefficients work, there has to be an optimal one out there that some savvy mind will come up with one day. One particular point we took painfully as a failure was not finding a pair of inhibition function and similarity coefficients that would work well with `ReLu`. And although we haven't solved ColoredMNIST yet, as is now a rite of passage for anti-spurious method, we have kept a clue[1] to ourselves on how we could grey scale accuracy; the result of which would go far beyond that success.

[1] (hint: The clue is in 3.3)

Finally, as a special bonus, they are yet more potent properties of LI left unexplored. For example...with the right weights, lateral inhibition allows for soft XOR to be computed in a single layer!, which cannot be accomplished with current architectures.

5.2. Conclusion

We hope this work has shown that the mysterious affliction plaguing our ANNs might have already been in part cured in biology. As we see that we have varying degrees of success. We cannot conclude that we may just need any mutual information relationship estimation paired with an inhibition/normalization to have a positive impact on robustness accuracy.

We have explored how local inhibition is not only a selection mechanism, but also a enhancing rebalancing mechanism. But we are left wondering : How do we reconcile rich representation & best explanation[**38**] & Ockham & invariant representation? If they are n other solutions to a problem, then there should exist a solution that explains these $n + 1$ solutions. And this is exactly what abstraction is for. If there is an easy solution, then it should not be relied on, as all solutions should be explored, which our networks are not designed for. We have shown that we can increase decision robustness in ANN by included a mechanism from biological networks: local inhibition.

Why does it work ?

Our original intent was that the saliency of digit features should be greater simply because of their number. A rich signature that would be picked up by any human and at least be associated with the oracle rather than ignored. Like in figure 3.3 ("cownet"), the saliency of the signature of `grass` will invariably be weakened at every layer as it correlated with cow features, whether it is part or not of some representation `cow` before the final layer. The saliency of digit features is sometimes suppressing that of the oracle because every feature of a digit suppresses the oracle more than they suppress each other, making for a very small value for the oracle at the last layer. That last layer is forced to assign greater and greater weight to it, so it can abuse of its predictive value. Making for high instability during training. This is where Spectral Decomposition came in as a partner to help tamper this excess. Whether this hypothesis is proved by the Oracle experiment remains to be debated. But it is certainly interesting to see the impact of a mechanism from biological networks on an ANN.

Références bibliographiques

[1] Subutai AHMAD et Jeff HAWKINS : How do neurons operate on sparse distributed representations? a mathematical theory of sparsity, neurons and active dendrites, 2016.

[2] Martha ALBERTSON FINEMAN et Rixanne MYKITIUK : Mapping the margins: Intersectionality, identity politics, and violence against women of color. *In The Public Nature of Private Violence;*, pages 93–118. New York: Routledge, 1994.

[3] Rahaf ALJUNDI, Marcus ROHRBACH et Tinne TUYTELAARS : Selfless sequential learning, 2018.

[4] Roger ARIEW : *Ockham's Razor: A Historical and Philosophical Analysis of Ockham's Principle of Parsimony.* Thèse de doctorat, University of Illinois at Urbana-Champaign, 1976.

[5] Martin ARJOVSKY, Léon BOTTOU, Ishaan GULRAJANI et David LOPEZ-PAZ : Invariant risk minimization, 2019.

[6] H.B. BARLOW : Unsupervised Learning. *Neural Computation*, 1(3):295–311, 09 1989.

[7] Anthony J. BELL et Terrence J. SEJNOWSKI : The "independent components" of natural scenes are edge filters. *Vision Research*, 37(23):3327–3338, 1997.

[8] Yoshua BENGIO, Aaron COURVILLE et Pascal VINCENT : Representation learning: A review and new perspectives, 2012.

[9] Lukas BIEWALD : Experiment tracking with weights and biases, 2020. Software available from wandb.com.

[10] Mateusz BUDA, Atsuto MAKI et Maciej A. MAZUROWSKI : A systematic study of the class imbalance problem in convolutional neural networks. *Neural Networks*, 106:249–259, oct 2018.

[11] Emmanuel J. CANDES et Michael B. WAKIN : An introduction to compressive sampling. *IEEE Signal Processing Magazine*, 25(2):21–30, 2008.

[12] Matteo CARANDINI et David J HEEGER : Normalization as a canonical neural computation. *Nature reviews. Neuroscience*, 13(1):51–62, 2012.

[13] Thomas CLELAND et Christiane LINSTER : On-center/inhibitory-surround decorrelation via intraglomerular inhibition in the olfactory bulb glomerular layer. *Frontiers in integrative neuroscience*, 6:5, 02 2012.

[14] Gustavo DECO et Wilfried BRAUER : Higher order statistical decorrelation without information loss. *In* G. TESAURO, D. TOURETZKY et T. LEEN, éditeurs : *Advances in Neural Information Processing Systems*, volume 7. MIT Press, 1994.

[15] Jia DENG, Wei DONG, Richard SOCHER, Li-Jia LI, Kai LI et Li FEI-FEI : Imagenet: A large-scale hierarchical image database. *In 2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255, 2009.

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

[17] Luc Devroye, László Györfi et Gábor Lugosi : *A Probablistic Theory of Pattern Recognition,Ch.7*, volume 31. 01 1996.

[18] Prafulla Dhariwal et Alex Nichol : Diffusion models beat gans on image synthesis, 2021.

[19] Jonathan Frankle et Michael Carbin : The lottery ticket hypothesis: Finding sparse, trainable neural networks. 2018.

[20] Robert Geirhos, Jörn-Henrik Jacobsen, Claudio Michaelis, Richard Zemel, Wieland Brendel, Matthias Bethge et Felix A. Wichmann : Shortcut learning in deep neural networks. *Nature Machine Intelligence*, 2(11):665–673, nov 2020.

[21] Robert Geirhos, Patricia Rubisch, Claudio Michaelis, Matthias Bethge, Felix A. Wichmann et Wieland Brendel : Imagenet-trained cnns are biased towards texture; increasing shape bias improves accuracy and robustness, 2018.

[22] Ian J. Goodfellow, Mehdi Mirza, Da Xiao, Aaron Courville et Yoshua Bengio : An empirical investigation of catastrophic forgetting in gradient-based neural networks, 2015.

[23] Anirudh Goyal et Yoshua Bengio : Inductive biases for deep learning of higher-level cognition, 2020.

[24] Suchin Gururangan, Swabha Swayamdipta, Omer Levy, Roy Schwartz, Samuel R. Bowman et Noah A. Smith : Annotation artifacts in natural language inference data, 2018.

[25] Song Han, Jeff Pool, John Tran et William J. Dally : Learning both weights and connections for efficient neural networks, 2015.

[26] Irina Higgins, David Amos, David Pfau, Sebastien Racaniere, Loic Matthey, Danilo Rezende et Alexander Lerchner : Towards a definition of disentangled representations, 2018.

[27] Andrew Ilyas, Shibani Santurkar, Dimitris Tsipras, Logan Engstrom, Brandon Tran et Aleksander Madry : Adversarial examples are not bugs, they are features, 2019.

[28] Sergey Ioffe et Christian Szegedy : Batch normalization: Accelerating deep network training by reducing internal covariate shift, 2015.

[29] Frederick A. A. Kingdom : Mach bands explained by response normalization. *Frontiers in Human Neuroscience*, 8, 2014.

[30] Polina Kirichenko, Pavel Izmailov et Andrew Gordon Wilson : Last layer re-training is sufficient for robustness to spurious correlations, 2022.

[31] Alexander Kraskov, Harald Stögbauer et Peter Grassberger : Estimating mutual information. *Physical Review E*, 69(6), jun 2004.

[32] Alex Krizhevsky, Ilya Sutskever et Geoffrey E Hinton : Imagenet classification with deep convolutional neural networks. *In* F. Pereira, C.J. Burges, L. Bottou et K.Q. Weinberger, éditeurs : *Advances in Neural Information Processing Systems*, volume 25. Curran Associates, Inc., 2012.

[33] David Krueger, Ethan Caballero, Jörn-Henrik Jacobsen, Amy Zhang, Jonathan Binas, Rémi Le Priol et Aaron C. Courville : Out-of-distribution generalization via risk extrapolation (rex). *CoRR*, abs/2003.00688, 2020.

[34] R. Linsker : Self-organization in a perceptual network. *Computer*, 21(3):105–117, 1988.

[35] Weiyang Liu, Rongmei Lin, Zhen Liu, Lixin Liu, Zhiding Yu, Bo Dai et Le Song : Learning towards minimum hyperspherical energy, 2018.

[36] Ziwei Liu, Ping Luo, Xiaogang Wang et Xiaoou Tang : Deep learning face attributes in the wild. *In Proceedings of International Conference on Computer Vision (ICCV)*, December 2015.

[37] Damiano Melotti, Kevin Heimbach, Antonio Rodríguez-Sánchez, Nicola Strisciuglio et George Azzopardi : A robust contour detection operator with combined push-pull inhibition and surround suppression. *Information Sciences*, 524:229–240, 2020.

[38] Giambattista Parascandolo, Alexander Neitz, Antonio Orvieto, Luigi Gresele et Bernhard Schölkopf : Learning explanations that are hard to vary, 2020.

[39] Mohammad Pezeshki, Sékou-Oumar Kaba, Yoshua Bengio, Aaron Courville, Doina Precup et Guillaume Lajoie : Gradient starvation: A learning proclivity in neural networks, 2020.

[40] Alexandre Ramé, Corentin Dancette et Matthieu Cord : Fishr: Invariant gradient variances for out-of-distribution generalization. *CoRR*, abs/2109.02934, 2021.

[41] Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu et Mark Chen : Hierarchical text-conditional image generation with clip latents, 2022.

[42] Pau Rodríguez, Jordi Gonzàlez, Guillem Cucurull, Josep M. Gonfaus et Xavier Roca : Regularizing cnns with locally constrained decorrelations, 2016.

[43] Nicole Rust et Vahid Mehrpour : Understanding image memorability. *Trends in Cognitive Sciences*, 24, 05 2020.

[44] Holger Schulze, Heinrich Neubauer, Frank W. Ohl, Andreas Hess et Henning Scheich : Representation of stimulus periodicity and its learning induced plasticity in the auditory cortex: Recent findings and new perspectives. *In Acta Acustica united with Acustica, Volume 88, Number 3*, pages 399–407. S. Hirzel Verlag, 2002.

[45] Lothar Spillmann, Birgitta Dresp et Chia-huei Tseng : Beyond the classical receptive field: The effect of contextual stimuli. *Journal of Vision*, 15(9):1–23, 07 2015.

[46] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever et Ruslan Salakhutdinov : Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(56):1929–1958, 2014.

[47] Remi Tachet, Mohammad Pezeshki, Samira Shabanian, Aaron Courville et Yoshua Bengio : On the learning dynamics of deep neural networks, 2018.

[48] Antonio Torralba et Alexei A. Efros : Unbiased look at dataset bias. *In CVPR 2011*, pages 1521–1528, 2011.

[49] Guillermo Valle-Pérez, Chico Q. Camargo et Ard A. Louis : Deep learning generalizes because the parameter-function map is biased towards simple functions, 2018.

[50] Zhennan Wang, Canqun Xiang, Wenbin Zou et Chen Xu : Mma regularization: Decorrelating weights of neural networks by maximizing the minimal angles. 2020.