AN ARTIFICIAL NEURAL NETWORK TO FIND
CORRELATION PATTERNS AMONG
AN ARBITRARY NUMBER OF VARIABLES

Alessandro Fontana

2016/06/22

Abstract

Methods to find correlation among variables are of interest to many disciplines, including
statistics, machine learning, (big) data mining and neurosciences. Parameters that measure
correlation between two variables are of limited utility when used with multiple variables. In
this work, I propose a simple criterion to measure correlation among an arbitrary number of
variables, based on a data set. The central idea is to i) design a function of the variables that
can take different forms depending on a set of parameters, ii) calculate the difference between
a statistics associated to the function computed on the data set and the same statistics com-
puted on a randomised version of the data set, called “scrambled” data set, and iii) optimise
the parameters to maximise this difference. Many such functions can be organised in layers,
which can in turn be stacked one on top of the other, forming a neural network. The function
parameters are searched with an enhanced genetic algorithm called POET and the resulting
method is tested on a cancer gene data set. The method may have potential implications for
some issues that affect the field of neural networks, such as overfitting, the need to process
huge amounts of data for training and the presence of “adversarial examples”.

1 Introduction

This paper is concerned with a new method to discover correlation patterns among an arbitrary
number of variables, based on a data set. It can be argued that finding correlations in a set of
variables is indeed the central objective of intelligence. The problem is of central importance in
many fields, such as statistics, machine learning, data mining and neurosciences. The method
that we are going to propose lies at the intersection of all these disciplines.

Many methods exist to measure correlation between two variables. Most of these measures
are based on some kind of comparison between the joint probability of the variables and the
product of their marginal probabilities. Two examples are given by the Pearson correlation
coefficient \[ 12 \] and mutual information \[ 1 \] (the correlation formula refers to the binary case,
P(X, Y) is the joint probability of X and Y, i.e. the probability of X ∩ Y).
2 Hyper-occurrence of features

Let us reformulate the correlation problem in more general terms. Let us suppose to have a data set RDS (the "real" data set) composed of N examples on M variables \((X_i)\) which can take real values in the \([0,1]\) interval and have arbitrary probability distributions. Based on the data, we want to find correlation patterns among the variables.

The first step towards a generalised definition of correlation consists in creating a new data set, called scrambled data set (SDS), starting from RDS. The procedure to create SDS is illustrated in Fig. 1 for an RDS with three examples and three variables. A generic row of SDS is obtained combining one of the possible values of each variable, and the whole SDS is created putting together all such combinations. In practise, SDS is a version of RDS in which the single
variables have the same probability distribution as in RDS, but where the correlations among variables are “broken”. Since the size of SDS ($N^M$) grows very rapidly, in practical applications it will be necessary to use samples of SDS. The product of probabilities $P(X) \cdot P(Y)$ in equation (1) can now be rewritten as $P(X \cap Y)|_{SDS}$. In this way, the numerator of equation (1) can be reformulated as:

$$P(X \cap Y)|_{RDS} - (P(X \cap Y)|_{SDS}$$

The second step to generalise equation (1) consists in replacing function $P(X \cap Y)$ through an arbitrary function $F(X_i)$ of the variables $X_i$. Then, we compute the probability $qr$ that $F$ is TRUE on RDS and the probability that $F$ is TRUE on SDS (for variables comprised in the $[0,1]$ interval, being TRUE means being $\geq 0.5$). Finally, we compare $qr$ and $qs$. This comparison can be done with many different formulas; in our computational experiments, the following expression has proved to be effective:

$$hoc = 1 - \frac{qs}{qr} = 1 - \frac{P(F(X_i)=TRUE)|_{SDS}}{P(F(X_i)=TRUE)|_{RDS}} \quad (3)$$

We propose this quantity, called hyper-occurrence ($hoc$), as a generalised definition of correlation among multiple variables. This parameter is defined only for $qs < qr$, in which case it is comprised in the $[0,1]$ interval. It is a measure of the difference between the distributions of values of function $F(X_i)$ on the two data sets RDS and SDS. More sophisticated difference measures, such as the Kullback-Leibler divergence, can also be used. Other variations, such as using the mean of $F$ instead of its TRUE-probability, are also possible.
3 Objective functions for unsupervised learning

So far, we have figured out how to assess whether a single new variable, function of the data set variables, captures correlation in its inputs. We can imagine to create many such variables and arrange them in layers, stack many such layers one on top of the other, and construct a neural network (Fig. 2). We call the new variables features.

If we optimise the function parameters with a search algorithm based on a certain objective function, we run the risk that two or more features found by the algorithm are identical or very similar, a result we are not interested in. The first idea to avoid it consists in computing the covariance matrix for all variables and discard variables that display a too high covariance value (if a couple has high covariance, we discard the variable with lower hoc). The covariance os calculated on SDS, as covariance on RDS is part of the correlation we want to discover.

The second idea derives from observing that, when two features are similar, they take low and high values in correspondence of the same examples. An indirect way to foster feature diversity consists in requesting that for each example there is a high number of active features, and that this number has a low variance across the data set. In other words, we want to avoid that for some examples there are many active features and for other examples very few: the data set “coverage” should be high and uniform (Fig. 3).

This result can be achieved with the following procedure. For each example e, for each feature $F_k$, we calculate the product of the feature value and the feature hoc value: $P_k(e) = F_k(e) \cdot hoc(F)$. Then, we compute the square root of the mean of the $H$ highest $P(e)$: $d(e) = \sqrt{E_k(P_k(e))}$ (H=10 in our simulations). Thanks to the square root, low covered examples have a disproportionately high value, which tends to favour uniformity of coverage. The objective function, which we call coverage, is the mean value of $d(e)$ on the whole RDS: $cov = E_n(d(e))$. The incorporation of hoc in the formulas guarantees that the data set coverage is obtained through features with high hoc values.
4 Simulation

In this work, the search of function parameters to optimise the quantities defined in the previous section will be carried out through an evolutionary algorithm called POET, based on an evo-devo model called Epigenetic Tracking (ET) \cite{2}. In ET artificial bodies are composed of two categories of cells: *stem cells* and *normal cells*. Development starts from a set of initial cells placed on a grid and unfolds in time through *developmental stages* regulated by a *global clock* shared by all cells. Stem cells direct the developmental process. When a stem cell is activated, it can orchestrate either a large-scale apoptosis (death of a large number of cells in the volume around a stem cell), or a cell proliferation, filling up the volume around the original stem cell. Cells, when created, take up place in a two dimensional grid. ET can be coupled with *genetic algorithm*, and becomes an evo-devo process able to generate complex structures (Fig. 4-left).

POET (for *Parameter Optimization using Epigenetic Tracking*) is a search method that builds upon ET. In POET each grid point is associated to a couple of values: \(k\) and \(c\). \(k\) represents the id number of a parameter, and \(c\) represents the parameter value. Both numbers can be modified by means of a set of change events, orchestrated by stem cells. In POET there are two types of change events: (1) proliferation, which changes the \(c\) values in the area around the activated stem cell, and (2) swap, which changes the \(k\) values. This is obtained by swapping the \(k\) values of the areas in the grid. Through a sequence of proliferation and swap events, the (\(c,k\)) values are modified in all grid positions (Fig. 4-right). This translates to a change of the encoded parameter set.

Essentially, compared to a standard genetic algorithm, POET allows to evolve chunks of the genome which encode changes to the parameter set. As evolution progresses, older changes are “frozen” (they cannot be evolved anymore). In previous work \cite{3} POET was used to train a neural

---

**Figure 3**: Data set coverage. Active features (value ≥ 0.5) are marked in red, inactive ones are marked in white. On the left: example of non-uniform data set coverage. Some examples are covered with many features, other examples are covered with few features or are not covered at all. On the right: example of uniform data set coverage. All examples are covered with approximately the same number of features. Features must have a high hoc value.
network, for a visual classification task represented by a subset of the MNIST written character dataset \[10\]. The choice of an evolutionary method to search the parameter space is motivated by its generality and applicability to an arbitrary objective function.

The method proposed has been tested on a subset of the tumor portal data set \[9\]. Each of the 100 rows of the data set represents a patient affected by Lung Adenocarcinoma. Each of the 1000 column contains a 1 if the corresponding gene is mutated, a 0 otherwise (these 1000 genes are those with the highest mutation rate for this kind of tumour). The purpose of this test is to see if the method is able to find correlation patterns on a real-word data set, not to gain insight into tumour biology. The rules searched have the form: if in the gene subset \(\{G_1, G_2, \ldots, G_k\}\) at least Q genes are mutated, the rule is TRUE. This is a quite general rule that includes logical And and logical Or as special cases.

We report the results of one simulation, in which POET has been allowed to run for 10000 generations. The network is composed of two layers (besides the input layer), each composed of 32 nodes, each node potentially fully connected to all nodes of all previous layers. Fig. 5 reports the 5 rules found with the highest hoc value for layers 1 and 2. Layer 2 rules tend to have higher qr values. The data set coverage value is 0.17 considering nodes of layer 1 and 0.21 considering nodes of layer 1 and nodes of layer 2. These preliminary results clearly show that the method is able to discover rules with high hoc values. Further conditions can be imposed on the the rules, on qr and qs values, to steer the search towards particular regions of the parameter space.

5 Discussion

The method described bears some resemblance to swap randomisation \[4\], a technique used to assess the statistical significance of itemsets found by data mining algorithms. The key differ-
Figure 5: Search results. The table reports the first 5 rules with the highest hoc value in layer 1 and layer 2. The field “bias” contains the number of genes in the subset that need to be mutated for the rule to be TRUE. Elements marked in yellow represent layer 1 nodes, which do not correspond to genes.

The difference between swap randomisation and hyper-occurrence is that in the first case the functions that capture correlation are chosen in advance and the aim of randomisation is to assess if the associations found are statistically significant. The only function used in this approach is the logical And of a subset of the inputs. This reflects the intended use, which is to discover frequent itemsets in a database of transactions.

In our approach, the difference between RDS and SDS is the driving force the shapes the choice of the functions, which can take different forms depending on a set of parameter values. Moreover, statistical significance is not the main criterion. Indeed, if the data set is very large, even small deviations between RDS and SDS are significant: our aim is to maximise this difference regardless of significance. Another difference concerns the structure of the randomised data set, which in [4] has the same row and column margins as the original data set, while in our case only the column margin is maintained. Finally, in our approach the layers of functions are stacked one on top of the other, building a neural network. We like to think of this work as a joint linking together the fields of statistics, data mining and neural networks.

The state of the art in the field of neural networks is represented by convolutional networks [8] trained with back-propagation, which appears to be immune from the gradient problem [6] when used with huge amounts of data. A recent method achieves an error rate of 3.57% on the ImageNet data set using a network with 152 layers [7]. However, in spite of the successes recorded, several issues remain unresolved.
As far as we know, the human cortex has a depth of 6 layers across all cortical areas [13]. Therefore, very deep networks do not seem to correspond to brain neurobiology. In addition, artificial neural networks require a very large number of training examples (60000 for MNIST, millions for ImageNet). Human beings, on their hand, can form new visual concepts with a more limited exposure to data.

Overfitting occurs when a complex model (many parameters) is used to interpret a small data set (few examples). The model ends up describing random error or noise instead of the relationship among variables, which translates to poor performance on unseen data. Several methods, such as regularisation [5], have been proposed to reduce overfitting, and new ones keep being proposed [14]. Deep networks do not seem to be much affected by the problem, simply because they are fed with huge amounts of data.

Adversarial examples are examples of images which appear as random noise to human observers, but that neural networks label with high confidence as belonging to one of the classes used for training [11]. Adversarial examples can be constructed by tweaking some pixels in a random image. This slightly depressing phenomenon has been explained by the presence of the long “shadow” cast by the network in regions of the weight space unexplored during training, in which the network cannot be used to make reliable predictions.

The method proposed discovers features that are statistically robust. Once this robustness is validated on the small subset of the data set. Statistical robustness could help to mitigate the overfitting phenomenon. Finally, since the features discovered have low occurrence on the scrambled data set, a classifier based on such features holds the promise of being immune from random-looking examples, such as the recently discovered adversarial examples. Altogether, a model based on HOC would come nearer to biological reality.

6 Conclusions

In this work we have described a new method to discover correlation patterns among an arbitrary number of variables. The method is based on a node-based statistical criterion called hyper-occurrence, and on a layer-based criterion which fosters node diversity through data set coverage. The parameter search is carried out with a special genetic algorithm called POET, it is however susceptible to be done with any suitable search algorithm. Future work will be aimed at exploring the application of this method to other real world data sets.

References

[1] T.M. Cover and J.A. Thomas. Elements of information theory. John Wiley and Sons, New York, NY, 1991.

[2] A. Fontana. Epigenetic tracking, a method to generate arbitrary shapes by using evo-devo
techniques. In Proceedings of the international conference on epigenetic robotics (EPIROB), 2008.

[3] A. Fontana, A. Soltoggio, and B. Wrobel. POET: an evo-devo method to optimize the weights of large artificial neural networks. In Proceedings of the 14th International Conference on the Simulation and Synthesis of Living Systems (ALIFE), pages 447–454, 2014.

[4] A. Gionis, H. Mannila, T. Mielikainen, and P. Tsaparas. Data mining results via swap randomization. In ACM Trans Knowl Discov Data. 1(3).

[5] F. Girosi, M. Jones, and T. Poggio. Regularization theory and neural network architectures. Neural Computation, 7:219–269, 1995.

[6] S. Hochreiter, Y. Bengio, P. Frasconi, and J. Schmidhuber. Gradient flow in recurrent nets: the difficulty of learning long-term dependencies. In A Field Guide to Dynamical Recurrent Neural Networks. IEEE press, 2004.

[7] H. Kaiming, Z. Xiangyu, R. Shaoqing, and S. Jian. Deep residual learning for image recognition. Arxiv, 2015.

[8] A. Krizhevsky, I. Sutskever, and G.E. Hinton. Imagenet classification with deep convolutional neural networks. In Advances in Neural Information Processing Systems 25, pages 1097–1105. Curran Associates, Inc., 2012.

[9] M.S. Lawrence et al. Discovery and saturation analysis of cancer genes across 21 tumour types. Nature, (505):495–501, 2013.

[10] Y. LeCun and C. Cortes. MNIST handwritten digit database. AT&T Labs [Online]. Available: http://yann. lecun. com/exdb/mnist, 1998.

[11] A. Nguyen, J. Yosinski, and J. Clune. Deep neural networks are easily fooled: high confidence predictions for unrecognizable images. In Computer Vision and Pattern Recognition (CVPR 15). 2015.

[12] K. Pearson. Notes on regression and inheritance in the case of two parents. In Proceedings of the Royal Society of London, pages 240–242. 1895.

[13] S. Shipp. Structure and function of the cerebral cortex. Current Biology, (17(12)):4439, 2007.

[14] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. Journal of Machine Learning Research, (15):1929–1958, 2014.