Nets versus trees for feature ranking and gene network inference

Nicolas Vecoven, Jean-Michel Begon, Vân Anh Huynh-Thu, Pierre Geurts *

Institut Montefiore, University of Liège, Belgium

Abstract. We propose to tackle the challenging problem of gene regulatory network inference, using variable importance measures derived from artificial neural networks (ANN). When combined with a L1-regularized selection layer, these measures allow ANN to be competitive with state of the art techniques for this problem based on random forests.

1 Introduction

In many supervised learning applications in bioinformatics, one is more interested by the interpretability of the trained model than by its actual predictive performance. One way to gain such interpretability is through the application of feature selection (or ranking) techniques, which aim at identifying the most relevant features for predicting a given output. In bioinformatics, as well as in other domains, Random Forests (RF) is one of the most popular methods for feature selection/ranking, as there are several very effective ways to derive variable importance scores from a forest [1]. On the other hand, despite their often excellent predictive performance, artificial neural networks (ANN) are often considered black-box models, which makes them less popular in bioinformatics than they are for example in computer vision. Recently however, motivated by the advent of deep learning, there has been a resurgence of interest towards (old and new) techniques to derive variable importance scores from ANN (see [2, 3] for reviews). Despite their genericity, these methods have been mostly evaluated on image classification tasks, where they highlight image regions that are responsible for the prediction associated to a given test image, on a per sample basis.

In this paper, we propose and evaluate several approaches to derive global variable importance scores from ANN from a feature selection perspective and compare them to variable importance scores derived from RF. We carry out experiments both on benchmark feature selection tasks and on a challenging bioinformatics task, namely gene regulatory network inference, where RF have been shown to perform very well.

2 Variable importances from neural networks

The methods described here compute, for each feature x_i, with $i \in \{1, \ldots, p\}$, an importance score $Imp(x_i)$ that assesses its relevance for predicting the output. The first two methods work for any pre-trained network with an arbitrary feed-forward structure, while the third method requires to train a specific structure.

*Vân Anh Huynh-Thu is a postdoctoral researcher at the FNRS, Belgium.

We assume here for simplification that all hidden neurons use ReLU activations and we expose each method in a (multi-output) regression setting. In the case of classification, a softmax layer is added and variable importances are derived considering the inputs of the softmax layer as multiple regression outputs.

Gradients (GRAD). A standard variable importance measure for ANN averages over a set of N examples (e.g., the training set) the absolute (or squared) value of the derivative of the ANN output according to the input to score [2]:

$$Imp(x_i) = \sum_{n=1}^{N} \left| \frac{\partial f(\mathbf{x}_n)}{\partial x_i} \right|, \tag{1}$$

where f denotes the ANN output and $\mathbf{x}_n \in \mathbb{R}^p$ denotes the nth training example. When there are multiple outputs, the importance score can be summed over the different outputs. These scores can be efficiently computed using backpropagation.

Layer-wise relevance propagation (LRP). Although commonly used, the gradient method has the drawback that it does not explain the output of the network but instead how the output varies when the input is changed [3]. Clearly, an input could be relevant even if (1) is zero at a given point. Several alternative importance measures have been proposed to circumvent this limitation. As a representative of these methods, we use below a particular instance of the generic LRP method proposed in [4]. Given a test example $\mathbf{x}$, this method associates an importance score $Imp(n_i, \mathbf{x})$ to each neuron n_i computed through a back-propagation scheme. Let us denote by $O(n_i, \mathbf{x})$ the output of the activation function of n_i at $\mathbf{x}$ (in the case n_i is an input neuron, $O(n_i, \mathbf{x})$ is the value of input i for $\mathbf{x}$). Assuming there are l output neurons, $Imp(n, \mathbf{x})$ for a neuron n of the last hidden layer is initialized as $Imp(n, \mathbf{x}) = \sum_{k=1}^{l} |O(n, \mathbf{x}) w_k|$, where w_k is the weight of the connection between n and the kth output. Let us then denote by $\{n_1, \ldots, n_m\}$ and $\{n'_1, \ldots, n'_n\}$ the neurons of two successive hidden layers and by w_{ij} the weight of the connection between n_i and n'_j. Then, the importance of a neuron n_i ($i \in \{1, \ldots, m\}$) is computed by:

$$Imp(n_i, \mathbf{x}) = \sum_{k=1}^{n} \frac{ReLU(O(n_i, \mathbf{x}) w_{ik}) Imp(n'_k)}{\sum_{j=1}^{m} ReLU(O(n_j, \mathbf{x}) w_{jk})}. \tag{2}$$

This propagation rule corresponds to the LRP rule with $\alpha = 1$ and $\beta = 0$ [4]. The importance of an input x_i is eventually computed as the sum of $Imp(n_i, \mathbf{x})$ over all training examples, with n_i the neuron corresponding to input x_i.

Selection layer (SL). The last method is inspired by sparse linear regression. A one-to-one connected layer with linear activations, called selection layer, is introduced between the inputs and the first hidden layer of the network. The network is then trained, starting from unit weights in the selection layer, to minimize the regular cost function (cross-entropy or least-square) plus a L1

penalty term on the weights of the selection layer. A new hyper-parameter $\alpha \geq 0$ is introduced that balances the L1 penalty term in the cost function, with $\alpha = 0$ meaning no penalty. Let us denote by w_i^{sl} the weight of input x_i in the selection layer, $Imp(x_i)$ is simply set to $\left|w_i^{sl}\right|$ after training. A similar idea with however a more complex regularization scheme has been proposed in [5]. Below, we also experiment with hybrid strategies, called GRAD+SL and LRP+SL, that train the network using the selection layer but then use variable importance as derived with the two previous techniques. This amounts to multiplying the SL weights with the importances computed on the network without the selection layer.

3 Experiments on benchmark problems

Datasets. To gain some insights about these methods, we first carry out experiments on four different artificial problems:

- LR a linear regression problem generated using the *make_regression* function in scikit-learn [6]. Output y is computed as $\sum_{i=1}^{25} w_i x_i$, where weights w_i are randomly selected in $[0, 100]$ and inputs x_i are $N(0, 1)$ distributed.
- LC a linear classification problem generated by thresholding the LR problem output so that the two classes are perfectly balanced.
- NLR A non-linear regression problem generated using the *make_friedman1* function in scikit-learn, which generates the following problem: $y = 10 * sin(pi * x_0 * x_1) + 20 * (x_2 - 0.5)^2 + 10 * x_3 + 5 * x_4 + 0.1 * \epsilon$, where ϵ is a $N(0, 1)$ noise and the x_i's are uniformly distributed in $[0, 1]$.
- NLC a non-linear classification problem generated using the *make_classification* function of scikit-learn with 25 relevant features. Briefly, a class among two is associated randomly to each vertex of a hypercube of dimension 25 and training examples of the corresponding class are generated in the neighboorhood of each vertex by using a normal distribution centered on the vertex (with $\Sigma = I$).

For each dataset, we add a varying number of irrelevant features (generated following a $N(0, 1)$ distribution).

Protocol. Five ANN-based importance scores are analyzed (GRAD, LRP, SL, SL+GRAD, SL+LRP). They are compared with RF mean decrease impurity (MDI) scores, using Gini entropy in classification and variance in regression [1]. Since the relevant features are known, the variable rankings are assessed using the area under the precision-recall curve (AUPR). The AUPR will be equal to 1 iff the ranking is perfect (i.e., all relevant variables receive a higher importance than the irrelevant ones), while the AUPR will be close to the proportion of relevant variables if the ranking is close to random. We also report the predictive performance of each model, ie., the error rate in classification and the mean squared error (MSE) in regression.

Table 1: AUPR and Error rate (ER)/MSE for the four datasets, with 5000 variables in total in each dataset.

		GRAD+SL	LRP+SL	SL	GRAD	LRP	RF
	ER		**0.027±0.005**		0.369±0.006		0.227±0.007
LC	AUPR	**0.927±0.039**	0.926±0.040	0.925±0.041	0.752±0.030	0.711±0.038	0.744±0.041
	ER		**0.143±0.034**		0.387±0.031		0.188±0.019
NLC	AUPR	0.747±0.070	0.810±0.063	**0.810±0.064**	0.597±0.089	0.595±0.079	**0.994±0.011**
	MSE		**0.037±0.005**		0.900±0.006		0.597±0.022
LR	AUPR	**0.984±0.019**	0.969±0.011	0.962±0.010	0.870±0.033	0.821±0.067	0.827±0.056
	MSE		**0.197±0.044**		0.733±0.003		0.215±0.006
NLR	AUPR	0.842±0.079	0.852±0.076	0.850±0.077	0.802±0.001	**0.918±0.079**	**1.000±0.000**

In all experiments, unless otherwise stated, each ANN is composed of 4 hidden layers of 500 ReLU [7] neurons each and is trained for 60000 steps on batches of size 50 using dropout and AdamOptimiser. RF models are composed of 1000 unpruned trees. For each dataset we use 2000 training samples and 8000 test samples. All inputs are centered and rescaled according to their standard deviation prior to training. The parameter α of SL, as well as the parameter K of RF (i.e., the number of randomly chosen variables at each tree node) are tuned by cross-validation[1], using error rate and MSE as a proxy for AUPR (since the latter can not be computed when relevant variables are unknown). All experiments are repeated five times (with newly generated data) and the means and standard deviations over these five runs are provided for each metric.

Results and discussion. Table 1 reports ER/MSE and AUPR for all methods on the four datasets with 4975 irrelevant variables for LC, LR, and NLC and 995 for NLR, and Table 2 shows the impact of the number of irrelevant variables on the NLC problem. Clearly, adding a selection layer is crucial to obtain good performance, especially when the number of irrelevant variables is large. Without this layer, ANN are worse than RF along both criteria. With SL, ANN outperform RF in terms of ER/MSE. They also clearly outperform RF in terms of AUPR on the linear problems (LC and LR), but RF are better at highlighting the relevant variables on the non-linear problems, despite less good predictive performance. Among the three SL methods, GRAD+SL seems to be inferior (resp. superior) on the non-linear (resp. linear) problems, while LRP+SL and SL are undistinguishable. Figure 1 shows the impact of the number of hidden layers for the NLR problem (left) and the impact of α for the NLC problem (right). Tuning the number of layers and α seems to be necessary to obtain good performance. ER/MSE seem to be good proxies for AUPR, although their optima do not perfectly coincide.

4 Application to gene network inference

An open problem in computational biology is the reconstruction of gene regulatory networks (GRNs) from gene expression data. A GRN aims at explaining the joint

[1]Values of α are optimized in $\{0, 10, 60, 100, 500\}$, while values of K are optimized in $\{\sqrt{p}, \log(p), p/3, p/2, p\}$ with p the number of inputs.

Table 2: Results of the different methods on the non-linear classification problem with increasing number of irrelevant features (from 25 to 7475).

# feat.		GRAD+SL	LRP+SL	SL	GRAD	LRP	RF
50	ER		**0.033 ± 0.004**		0.039 ± 0.003		0.106 ± 0.006
	AUPR	1.000 ± 0.001	**1.000 ± 0.000**	1.000 ± 0.001	1.000 ± 0.001	**1.000 ± 0.000**	**1.000 ± 0.000**
2500	ER		**0.078 ± 0.018**		0.324 ± 0.020		0.160 ± 0.013
	AUPR	0.873 ± 0.059	**0.906 ± 0.039**	0.901 ± 0.044	0.636 ± 0.118	0.628 ± 0.126	**0.999 ± 0.001**
5000	ER		**0.143 ± 0.034**		0.387 ± 0.031		0.188 ± 0.019
	AUPR	0.747 ± 0.070	0.810 ± 0.063	**0.810 ± 0.064**	0.597 ± 0.089	0.595 ± 0.079	**0.994 ± 0.011**
7500	ER		**0.159 ± 0.018**		0.408 ± 0.018		0.211 ± 0.006
	AUPR	0.738 ± 0.053	**0.784 ± 0.059**	0.784 ± 0.061	0.579 ± 0.088	0.571 ± 0.084	**0.996 ± 0.005**

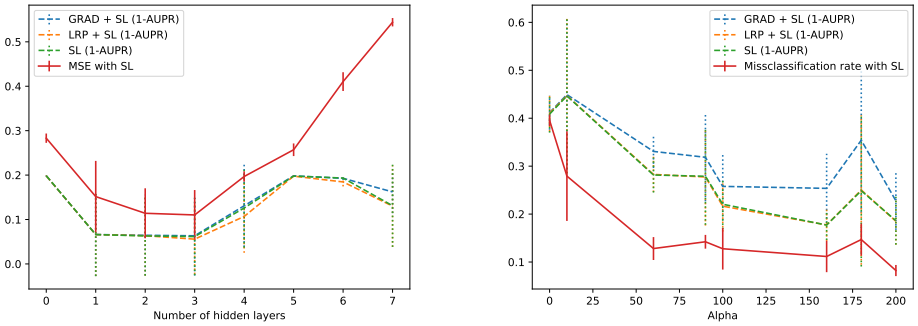

Fig. 1: Impact of the number of hidden layers (500 neurons each) for NLR with 995 irrelevant variables (left) and α for NLC with 4975 irrelevant variables (right).

variability in the expression levels of a group of genes, through a directed graph where an edge e_{ij} going from gene g_i to gene g_j indicates that g_i regulates the expression of g_j. Often, one aims at reconstructing a *weighted* network, where each putative edge is associated with a confidence weight. One approach to the reconstruction of weighted GRNs consists in solving one regression problem for each gene g_j in turn, with the expression of g_j as output variable and the expressions of the other genes as input variables. The variable importance score of gene g_i in the model predicting the expression of g_j is then used as weight for the edge e_{ij}. Using this framework, the RF are currently one of the state-of-the-art approaches for GRN inference [8].

We used our ANN-based variable importance scores to reconstruct the five networks of the DREAM4 multifactorial challenge [9]. Each network is composed of 100 genes, for which the expressions in 100 samples are available. In each case, we selected the ANN architecture and the parameter α in SL that optimize the accuracy computed using cross-validation[2]. Table 3 shows the AUPR obtained with ANN and RF. We again observe that the addition of SL improves the performances, without however a clear winner among SL, GRAD+SL and LRP+SL. Although the ANN approaches do not outperform the state-of-the-art RF, they nevertheless yield very good performances with respect to the latter.

[2]The number of hidden layers was optimized in {2,3}, the number of neurons per layer was optimized in {50,150} and the value of α was optimized in {5,60,300,800,1500}.

Table 3: AUPR for the five DREAM4 networks

	GRAD+SL	LRP+SL	SL	GRAD	LRP	RF
Net 1	**0.148**	0.143	0.126	0.118	0.100	**0.155**
Net 2	0.109	0.101	**0.121**	0.085	0.095	**0.153**
Net 3	0.178	**0.193**	0.191	0.146	0.166	**0.225**
Net 4	0.184	0.172	**0.192**	0.144	0.149	**0.208**
Net 5	**0.187**	0.180	0.166	0.133	0.143	**0.199**

5 Conclusion

We evaluated several feature ranking techniques based on ANN and compared them on several problems with RF, chosen as a state of the art reference. ANN have clearly the upper hand over RF in linear settings and they are also able to withstand themselves in non linear settings, although slightly inferior to RF. The introduction of a L1-regularized selection layer turns out to be crucial both for feature selection and predictive performance when the number of irrelevant features is large. The different scores are very similar, although the gradient seems to lag behind in non-linear settings. Results on GRN inference are promising and could probably be improved further, as future work, with more extensive parameter tuning. Future works will also include experiments on larger real GRN applications (where computing times will be an important issue), the inclusion of other importance scores and regularizations, as well as a better characterization of the different importance scores.

References

[1] G. Louppe, L. Wehenkel, A. Sutera, and P. Geurts. Understanding variable importances in forests of randomized trees. In *Advances in Neural Information Processing Systems 26*, pages 431–439. 2013.

[2] P. Leray and P. Gallinari. Feature selection with neural networks. *Behaviormetrika*, 26(1):145–166, 1999.

[3] G. Montavon, W. Samek, and K.-R. Müller. Methods for interpreting and understanding deep neural networks. *Digital Signal Processing*, 73(Supplement C):1 – 15, 2018.

[4] S. Bach, A. Binder, G. Montavon, F. Klauschen, K.-R. Müller, and W. Samek. On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation. *PloS one*, 10(7):e0130140, 2015.

[5] Y. Li, C.-Y. Chen, and W. W. Wasserman. Deep feature selection: Theory and application to identify enhancers and promoters. *RECOMB2015*, pages 205–217, 2015.

[6] F. Pedregosa *et al.* Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[7] V. Nair and G.E. Hinton. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th international conference on machine learning (ICML-10)*, pages 807–814, 2010.

[8] V. A. Huynh-Thu, A. Irrthum, L. Wehenkel, and P. Geurts. Inferring regulatory networks from expression data using tree-based methods. *PLoS ONE*, 5(9):e12776, 2010.

[9] D. Marbach, T. Schaffter, C. Mattiussi, and D. Floreano. Generating realistic in silico gene networks for performance assessment of reverse engineering methods. *Journal of Computational Biology*, 16(2):229–239, 2009.