ESANN 2018 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 25-27 April 2018, i6doc.com publ., ISBN 978-287587047-6.
Available from http://www.i6doc.com/en/.

Vector Field Based Neural Networks

Daniel Vieira[1], Fabio Rangel[1] , Fabrício Firmino[1] and Joao Paixao[1] *

1- Universidade Federal do Rio de Janeiro (UFRJ)
Graduation Program in Informatics (PPGI)
Av. Athos da Silveira Ramos, 149, Rio de Janeiro, RJ - Brazil

Abstract. A novel Neural Network architecture is proposed using the mathematically and physically rich idea of vector fields as hidden layers to perform nonlinear transformations in the data. The data points are interpreted as particles moving along a flow defined by the vector field which intuitively represents the desired movement to enable classification. The architecture moves the data points from their original configuration to a new one following the streamlines of the vector field with the objective of achieving a final configuration where classes are separable. An optimization problem is solved through gradient descent to learn this vector field.

1 Introduction

Understanding how *Black Box* models [1] work, such as Deep Neural Architectures [2] and Support Vector Machines [3], is an extensively discussed problem in the literature [4], where many works focus in visualizing and understanding their behavior [5] [6] [7]. It is possible to address the problem of comprehending Neural Networks behavior in a geometrical sense by understanding them as universal function approximators [8]. Another possible interpretation considers the hidden layers in Neural Networks as nonlinear continuous transformations of the domain space [9]. Towards the end of his work, the author suggests that vector fields might be better to handle these transformations than traditional layers. Inspired by [9], the present work proposes to combine Neural Networks with vector fields in order to understand data separation with data points as particles moving along a flow.

Vector fields have also been recently used to analyze the optimization problem in Generative Adversarial Networks (GANs) [10], achieving remarkable results for visualization and comprehension of GANs limitations and how to extend them. This work introduces a novel architecture using vector fields as activation functions. We optimize vector fields parameters through stochastic gradient descent using binary cross entropy as loss (cost) function.

By applying the concept of vector fields in Neural Networks, a vast amount of established mathematical and physical concepts, abstractions, and visualizations arises for the Neural Networks. For instance, Euler's method for solving ordinary differential equations [11] is used in this work to implement the concept of data points as particles moving through a flow.

This paper presents a computational experiment performed over three nonlinear separable, two dimensional datasets, using a vector field generated by a

*The authors would like to acknowledge CNPq/CAPES for funding.

ESANN 2018 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 25-27 April 2018, i6doc.com publ., ISBN 978-287587047-6.
Available from http://www.i6doc.com/en/.

simple Gaussian kernel function. With different initialization hyperparameters the cost function shows consistent reduction through epochs and the results are further analyzed. Section 2 describes the architecture and the optimization problem.

2 Vector Field Neural Networks

A vector field on $\mathbb{R}^n$ is a smooth function $K : \mathbb{R}^n \to \mathbb{R}^n$. Consider the corresponding ordinary differential equation (ODE):

$$X'(t) = K(X(t)),\ X(t_0) = X_0$$

with $X \in \mathbb{R}^n$. A curve $X(t)$ solving the ODE is called an streamline of the vector field K. Given a particle in a position $X(t_0) = X_0$ at time t_0, a possible physical interpretation is where each vector $K(X)$ represents the velocity acting on the particle at a given point in space and the streamline is the displacement done on the particle as it travels along the path $X(t)$. At time $t_N > t_0$, the particle will be in position $X(t_N)$.

Given a family of vector fields $K(X, \theta)$ defined by some parameters θ, we proposed to find the best vector field in this family aiming to transform every point X_0 in the input space, in a point $X(t_N)$ in the transformed space such that points in distinct classes would be linearly separable. Intuitively, the vector field represents the desired movement to enable the classification.

Euler's method [11] is used to approximate $X(t_N)$ by X_N, the solution of the ODE, with $X_i \approx X(t_0 + ih)$ for the discretization and $K(X, \theta)$ as our vector field with the iteration:

$$X_{i+1} = X_i + hK(X_i, \theta) \qquad 0 \leqslant i \leqslant N, \tag{1}$$

where h (the step size) and N (number of steps), such that $t_N = t_0 + Nh$, are hyperparameters and θ represents the vector field parameters. Considering Euler's method, it is known that for $h \to 0$, the streamlines of $K(\theta, X)$ are calculated exactly.

Figure 1 presents the input data being transformed by the vector field layer of the architecture. It also presents the optimized vector field which its goal is to linearly separate the data. Note that the architecture's last layer is a linear separator, which can be implemented using a logistic function.

3 Methodology

3.1 Vector Field Optimization

A computational experiment was developed using a simple kernel function to define the family of vector fields. Kernel $K(X, \theta)$ is presented in Equations 2 and 3, where $V = \{V_1, V_2, \ldots, V_S\}$ are the vectors where $V_i \in \mathbb{R}^n$ and $M =$

ESANN 2018 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 25-27 April 2018, i6doc.com publ., ISBN 978-287587047-6.
Available from http://www.i6doc.com/en/.

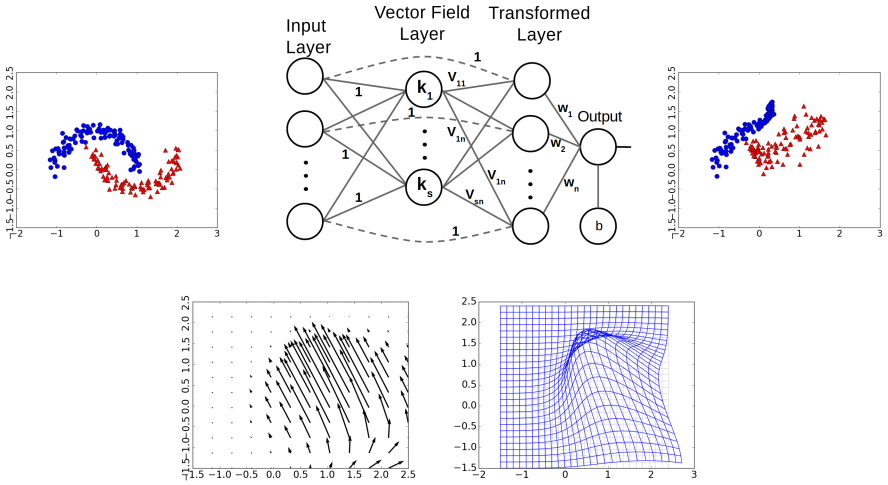

Fig. 1: From left to right, first row presents input data, the architecture, and the transformed data by the vector field layer. Second row presents the vector field and the space distortion.

$\{\mu_1, \mu_2, \dots \mu_S\}$ are the means of the Gaussians where $\mu_i \in \mathbb{R}^n$, $\theta = \{V, M\}$, and S is the number of Gaussians,

$$G(X, \mu) = e^{-||X-\mu||^2}, \qquad X,\ \mu \in \mathbb{R}^n, \tag{2}$$

$$K(X, \theta) = \sum_{i=1}^{S} V_i G(X, \mu_i), \qquad V_i \in \mathbb{R}^n. \tag{3}$$

Note that variance is controlled through the V_i parameters, which can be thought of as weight vectors that provide a direction to the vector field (see Figure 1 for an example). The final layer is a logistic function with a binary cross entropy cost that acts upon the transformed points X_N:

$$\hat{y} = \frac{1}{1 + e^{-(w^t X_N + b)}}, \qquad w,\ X_N \in \mathbb{R}^n \text{ and } b,\ \hat{y} \in \mathbb{R} \text{ st. } 0 < \hat{y} < 1. \tag{4}$$

Let w, V_i and μ_i initial values be random vectors with distribution $U[0, 1]$ in $\mathbb{R}^n$ and G_i the corresponding distribution and L2 regularization of vectors V_i. The gradient of $\mu_{j,i}$ and $V_{j,i}$, the respective j-th components of vectors μ_i and V_i, are presented in Equations 5 and 6, where x_j is the j-th component of X, w_j/w_k are the j-th and k-th components of vector w and λ is the regularization parameter.

$$\frac{\partial C}{\partial \mu_{j,i}} = 2(\hat{y} - y)G(X, \mu_i)(x_j - \mu_{j,i}) \sum_{k=1}^{n} w_k V_{k,i}, \tag{5}$$

$$\frac{\partial C}{\partial V_{j,i}} = (\hat{y} - y)G(X, \mu_i)w_j + \eta\lambda V_{j,i}. \tag{6}$$

ESANN 2018 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 25-27 April 2018, i6doc.com publ., ISBN 978-287587047-6.
Available from http://www.i6doc.com/en/.

3.2 Experiment Design

Two *scikit-learn* machine learning datasets [12] (moons and circle) and a sin dataset (created by the authors) were used in this paper. Figure 2 shows the datasets.

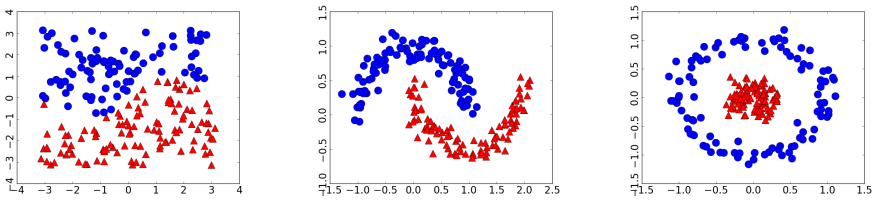

Fig. 2: Sin, Moons and Circle datasets, respectively.

For each dataset, the cost function is evaluated 30 times, using random initial values for the θ parameters, learning rates in $\{0.03, 0.30, 3.00\}$, and no regularization ($\lambda = 0$). Hyperparameters h and N are both set to one and $S = 2$ in this paper's experiments. Training is made with the entire dataset in a full-batch fashion, hence, there is no validation/test set. Results are presented for the cost throughout 10000 epochs in the *circles* dataset. Next, the boundary layer is calculated as a color map, with the original data points plotted over it. Thus, we can analyze the relationship between the boundary layer in the original and transformed space. Finally, the effects of regularization can be visualized and compared in a color map for *sin* dataset.

4 Results and Discussion

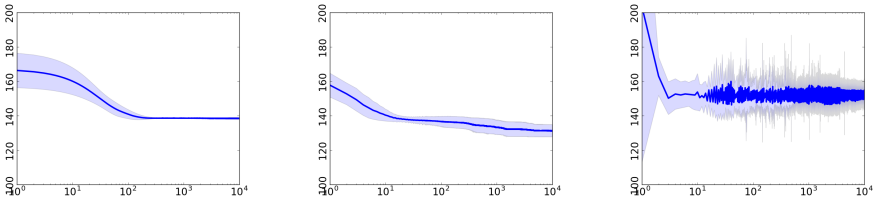

Fig. 3: Cost vs. Epochs. *Circles* dataset with $\theta = \{0.03, 0.3, 3.0\}$ respectively.

Analyzing the cost function along the epochs for different learning rates for the *circles* dataset shows the reduction of cost through epochs and an interesting pattern appears: as the learning rate increases the cost function and its deviation becomes less smooth and the standard deviation increases as well.

In Figure 4, it is possible to see that the original boundary layer turns into a hyperplane on the transformed space. Although the algorithm achieved good

ESANN 2018 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 25-27 April 2018, i6doc.com publ., ISBN 978-287587047-6.
Available from http://www.i6doc.com/en/.

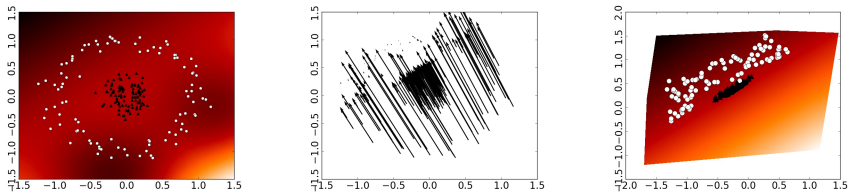

Fig. 4: Original space; vector field; and transformed space, respectively.

classification by bending the space and extracting the center of the circle to the outside, it generates a superposition of different points in the original space. Thus, in the region where this happens, misclassification occurs and this should be avoided. One way to diminish the algorithm's power to create such extreme movements is by the use of regularization. Figure 5 acting as a damper, smoothing the movements happening on the original space, preventing the overlapping of different points in the same region of the transformed space.

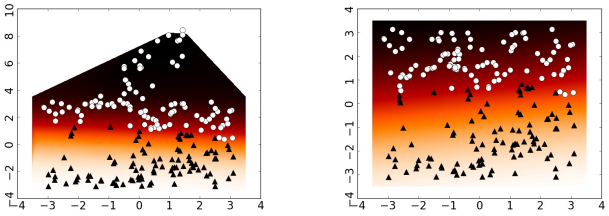

Fig. 5: Regularization over *sin* dataset (5000 epochs, $\eta = 3.0$ and $\lambda = 0.0005$).

The choice of $N = 1$ and $h = 1$ may facilitate overfitting, since the steps taken by data points are far greater and the constriction on how many movements can be made is at its maximum. It's possible to drawn an analogy of regularization been automatically made when we choose good values of h and N, as only small steps are taken and that streamlines are followed.

5 Conclusion

This work presents a novel neural network based on vector fields. The base of this network is to move points along a flow in the space, allowing the posterior separation of these points using a linear classifier. The vector fields are created using kernel functions. This approach brings the vast amount of well established vector fields' theory enabling the geometrical interpretation of how this neural network works. Initialization of parameter and hyperparameter plays an important role on some cases and must be taken into account. Regularization was shown to act as damper, reducing the capability of the vector field to move points

ESANN 2018 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 25-27 April 2018, i6doc.com publ., ISBN 978-287587047-6. Available from http://www.i6doc.com/en/.

and diminish disruption in the dataset original space. Experiments presented cost function reduction, which indicates learning capability, and the optimized flow used to move the points.

Further work needs to be done exploring real world datasets and evaluating learning performance with validation test sets. Another topic includes the investigation of the architecture performance when hyperparameter h is small and N is large.

References

[1] Alex A. Freitas, Daniela C. Wieser, and Rolf Apweiler. On the importance of comprehensible classification models for protein function prediction. *IEEE/ACM Transactions on Computational Biology and Bioinformatics (TCBB)*, 7(1):172–182, 2010.

[2] Yash Goyal, Akrit Mohapatra, Devi Parikh, and Dhruv Batra. Interpreting visual question answering models. In *ICML Workshop on Visualization for Deep Learning*, volume 2, 2016.

[3] Nahla Barakat and Andrew P. Bradley. Rule extraction from support vector machines: Measuring the explanation capability using the area under the roc curve. In *Pattern Recognition, 2006. ICPR 2006. 18th International Conference on*, volume 2, pages 812–815. IEEE, 2006.

[4] Alex A. Freitas. Comprehensible classification models: A position paper. *SIGKDD Explor. Newsl.*, 15(1):1–10, March 2014.

[5] Matthew D. Zeiler and Rob Fergus. Visualizing and understanding convolutional networks. In *European conference on computer vision*, pages 818–833. Springer, 2014.

[6] Jason Yosinski, Jeff Clune, Thomas Fuchs, and Hod Lipson. Understanding neural networks through deep visualization. In *Deep Learning Workshop*. 31st International Conference on Machine Learning, 2015.

[7] Chris Olah, Alexander Mordvintsev, and Ludwig Schubert. Feature visualization. *Distill*, 2017. https://distill.pub/2017/feature-visualization.

[8] Kurt Hornik. Approximation capabilities of multilayer feedforward networks. *Neural networks*, 4(2):251–257, 1991.

[9] Chris Olah. Neural networks, manifolds and topology. http://colah.github.io/posts/2014-03-NN-Manifolds-Topology/.

[10] Lars Mescheder, Sebastian Nowozin, and Andreas Geiger. The numerics of gans. *arXiv preprint arXiv:1705.10461*, 2017.

[11] J. C. Butcher. *The Numerical Analysis of Ordinary Differential Equations: Runge-Kutta and General Linear Methods*. Wiley-Interscience, New York, NY, USA, 1987.

[12] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.