Interactive Visualization of Neural Network Activities

Zijian Li
University of Washington

Callin Switzer
University of Washington

Yue Zhao
University of Washington

Zhengde Zhao
University of Washington

ABSTRACT

A clear and well-documented LaTeX document is presented as an article formatted for publication by ACM in a conference proceedings or journal publication. Based on the "acmart" document class, this article presents and explains many of the common variations, as well as many of the formatting elements an author may use in the preparation of the documentation of their work.

KEYWORDS

Neural networks, Data Visualization, Interactive

ACM Reference Format:
Zijian Li, Callin Switzer, Yue Zhao, and Zhengde Zhao. 2019. Interactive Visualization of Neural Network Activities. In *Proceedings of ACM Conference (Conference'17).* ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

Background

Insects like moths have very simple brains, yet they are capable of precise and subtle flying maneuvers, which involve complex fluid dynamics. They generate locomotor force by activating the flight muscles to move their wings, and the aerodynamic forces and torques it generates enable they to preform various flying behaviors including fast forward, odor plume tracking, hovering in front of flowers, decelerating upon approach and conpensating for enrionmental perturbations [?] [?] . To understand how moths control their flight is already by itself interesting, not to mention that bio-inspired flapping wings system has potential application to micro air vehicles according to Chen in [?].

[Picture of a realistic Moth]

Conference'17, July 2017, Washington, DC, USA
© 2019 Association for Computing Machinery.
ACM ISBN 978-x-xxxx-xxxx-x/YY/MM...$15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

Multiple recent researchers has targeted this question. In [?], the authors treated this question as an inverse problem, where the input of the system is the initial position, initial velocity, targeting position and targeting velocity while the output of the system is the wing motion. They developed an aerodynamic model illustrating how a particular motion in the wings of a moth could result in a kinetic motion. In [?], they found that not only the wings but also the shape of the body plays an important role in a moth's flight control. New variables such as the angle of the body of the moth were introduced into the model. Further study has also shown that structural deformation also affects flapping wing energetics, see [?]. Nevertheless, to solve the inverse problem of flight control is intrinsically difficult. It involves solving a highly nonlinear dynamically system.

On the other side, people has developed theory to solve dynamical systems with machine learning techniques. For example see [?]. Moreover, recent study show some very successful examples in solving model based motion control problem with deep learning neural networks. For instance, see [?] and [?] for legged robot locomation and [?] for aggressive control of autonomous quadrotors. With these successes, people are also looking into such a deep learning solution to the dynamical system of moth flight control.

Neural Network & Visualization

Dr. Switzer, a researcher in the Biology Department of University of Washington, and his colleagues developed several deep learning neural networks in order to solve the moth flight control problem. More specifically, they want to know the following. Suppose I am a moth and I know where I am and where I want to go, how do I get there? To answer this, they first built up a virtual moth. With this virtual moth, they simulated a dataset consisting of the actions the moth is taking and its resulting kinetic motion. After that, they trained several different neural networks, with different scales, using this dataset and lastly they evaluated the predictions by the neural network. It turns out that the best of their model is able to recreate a large share of the variance of the dependent variables in this model, with $r^2 > 0.999$.

[Picture as PPT page 10]
[Picture prediction result]

Now with these neural networks, we are able to generate an action for a particular intended motion that a moth is to take. The next step is to see how meaningful information we are able to extract from this neural network solution.

2 METHODS

Neural networks are a special type of graph networks, which are highly organized in their structures. For a feedforward neural network used in this study, nodes in the graph are divided into layers, and the group of nodes in each layer only connect with nodes in the adjacent layers. [?] described the visualization of feedforward neural networks, which aligns the position of the nodes according to their layer number. The following Figure 1 shows the structure of one example of a fully-connected feedforward neural network used in the virtual moth modeling. The figure also shows a typical way of visualizing such a neural network: nodes are aligned onto different lines, using the horizontal position to encode their layers. Weights are then drawn as lines connected each two nodes in adjacent layers.

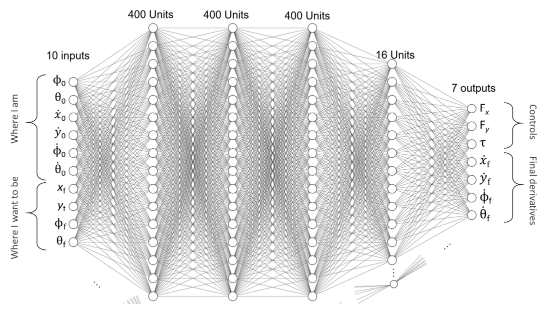

Figure 1: Feedforward Neural Network
A example of a feedforward neural network structure used in the modeling of a virtual moth. The network is fully-connected between adjacent layers.

The above visualization of a neural network is clear in understanding the model structure, but has some drawbacks as well. First of all, this visualization is static, which does not respond to the changes in the network, such as activation on the nodes, cutting of of weight connections. Secondly, it is bad under scaling, and can only serve as a conceptual illustration. For example, there are 400 units in the middle layers, but the visualization is incapable of showing all of them, thus it is impossible to show the complete activities and weights architecture in the network. The alignment of all the nodes in one layer, while adjustable to different network structures is a big problem. Lastly, such visualization is only suitable to simple constructions of neural networks such as the feedforward network. With the rapid development of new network structures nowadays, such as the recurrent neural network based on [?], a simple static framework for visualizing all types of neural networks is impossible. We tackles these problems with the following methods.

Computational Graph Construction

For a fully-connected feedforward neural network, the connections between each two adjacent layers are in the form of a bipartite graph, which can be represented by a weight matrix W, where W_{ij} represents the weight between the i-th node in the previous layer and the j-th nodes in the next layer. The weight matrices are also the typical data structure of storing a neural network. To consider an arbitrary construction of neural network, however, we consider the most general super-class of neural networks, the computational graph model. The directed graph

$$G = (V, E)$$

consists of a set of nodes V and the set of edges E. Each edge $e \in E$ goes from a node $v_1 \in V$ to another node $v_2 \in V$. In the neural network, the elements in the graph have their properties as follows:

$$\begin{aligned} v \in V &: \text{ node ID } i, \text{ layer number } l, \text{ activation value } u, \\ e \in E &: \text{ source } i, \text{ target } j, \text{ weight } w. \end{aligned}$$

Note that the activation value u on each nodes is determined not only by the neural network itself, but also depends on the inputs to the network. Therefore, the network construction should be real-time in response to interactive input selection. To solve this problem, we construct the graph together with all the element properties listed above. For the feedforward neural network case in our study, we implement the forward pass algorithm that starts from the input layers, and iterates throughout the whole network to the output layer.

Since many neural networks are highly pruned in our study, a large portion of weights are zeros or close to zero. It is undesirable to show these edges, and thus we throw away these edges in the construction phase. This would help largely reduce the amount of computation in the following visualization stage. In addition to edge trimming, we also walk through the network after construction, and throw away any node which is not connected to any edge. This way the trimmed network is kept as a connected graph, in preparation for the force-directed layout in the next stage.

Force Directed Layout

In stead of aligning the nodes according to the layers into strict lines, we use the force directed layout scheme [?]. Each node v_i is regarded as a charged particle with charge q_i. Each two nodes v_i and v_j repel each other with the force

$$F = \frac{q_i q_j}{d_{ij}^2}.$$

Each edge e is considered as a spring, applying the attracting force to the nodes it connects:

$$F = k(L - d_{ij}).$$

Besides, each node in motion is subject to air resistance

$$F = -bv_i.$$

At each time step, the forces acting on each node are calculated, which are then integrated to update its velocity and position. In our system, we implement the force directed layout scheme in d3-force, which uses a velocity Verlet numerical integrator for simulating physical forces on particles.

To show the layered structure of the neural network better, we apply external force field to form a beeswarm plot. Similar to plotting nodes in different layers onto different X-positions, we apply attractive X-forces to nodes in each layer towards the corresponding X-position. A constant attractive force in the Y direction is also applied to ensure reasonable vertical scale of the plot. The whole graph is also attracted to the center of the plot. To make the algorithm scalable to different sizes of networks, we choose the charges and forced according to the total number of nodes in the graph:

$$F_X \sim \frac{1}{|V|^2}, \quad q \sim \frac{1}{|V|}.$$

3 TABLES

The "`acmart`" document class includes the "`booktabs`" package — https://ctan.org/pkg/booktabs — for preparing high-quality tables.

Table captions are placed *above* the table.

Because tables cannot be split across pages, the best placement for them is typically the top of the page nearest their initial cite. To ensure this proper "floating" placement of tables, use the environment **table** to enclose the table's contents and the table caption. The contents of the table itself must go in the **tabular** environment, to be aligned properly in rows and columns, with the desired horizontal and vertical rules. Again, detailed instructions on **tabular** material are found in the *LaTeX User's Guide*.

Immediately following this sentence is the point at which Table 1 is included in the input file; compare the placement of the table here with the table in the printed output of this document.

To set a wider table, which takes up the whole width of the page's live area, use the environment **table*** to enclose the table's contents and the table caption. As with a single-column table, this wide table will "float" to a location deemed more desirable. Immediately following this sentence is the point at which Table is included in the input file; again, it is instructive to compare the placement of the table here with the table in the printed output of this document.

Table 1: Frequency of Special Characters

Non-English or Math	Frequency	Comments
Ø	1 in 1,000	For Swedish names
π	1 in 5	Common in math
$	4 in 5	Used in business
Ψ_1^2	1 in 40,000	Unexplained usage

4 FIGURES

The "`figure`" environment should be used for figures. One or more images can be placed within a figure. If your figure contains third-party material, you must clearly identify it as such, as shown in the example below.

Figure 2: 1907 Franklin Model D roadster. Photograph by Harris & Ewing, Inc. [Public domain], via Wikimedia Commons. (https://goo.gl/VLCRBB).

The 1907 Franklin Model D roadster.

Your figures should contain a caption which describes the figure to the reader. Figure captions go below the figure. Your figures should **also** include a description suitable for screen readers, to assist the visually-challenged to better understand your work.

Figure captions are placed *below* the figure.

The "Teaser Figure"

A "teaser figure" is an image, or set of images in one figure, that are placed after all author and affiliation information, and before the body of the article, spanning the page. If you wish to have such a figure in your article, place the command immediately before the `\maketitle` command:

```
\begin{teaserfigure}
 \includegraphics[width=\textwidth]{sampleteaser}
  \caption{figure caption}
  \Description{figure description}
\end{teaserfigure}
```

5 SIGCHI EXTENDED ABSTRACTS

The "`sigchi-a`" template style (available only in LaTeX and not in Word) produces a landscape-orientation formatted article, with a wide left margin. Three environments are available for use with the "`sigchi-a`" template style, and produce formatted output in the margin:

- `sidebar`: Place formatted text in the margin.
- `marginfigure`: Place a figure in the margin.
- `margintable`: Place a table in the margin.

Table 2: Some Typical Commands

Command	A Number	Comments
`\author`	100	Author
`\table`	300	For tables
`\table*`	400	For wider tables

ACKNOWLEDGMENTS

To Robert, for the bagels and explaining CMYK and color spaces.

A RESEARCH METHODS

Part Two

Etiam commodo feugiat nisl pulvinar pellentesque. Etiam auctor sodales ligula, non varius nibh pulvinar semper. Suspendisse nec lectus non ipsum convallis congue hendrerit vitae sapien. Donec at laoreet eros. Vivamus non purus placerat, scelerisque diam eu, cursus ante. Etiam aliquam tortor auctor efficitur mattis.

B ONLINE RESOURCES

Nam id fermentum dui. Suspendisse sagittis tortor a nulla mollis, in pulvinar ex pretium. Sed interdum orci quis metus euismod, et sagittis enim maximus. Vestibulum gravida massa ut felis suscipit congue. Quisque mattis elit a risus ultrices commodo venenatis eget dui. Etiam sagittis eleifend elementum.

Nam interdum magna at lectus dignissim, ac dignissim lorem rhoncus. Maecenas eu arcu ac neque placerat aliquam. Nunc pulvinar massa et mattis lacinia.