Feature extraction in evolved/optimized spiking neural networks

Cristian Jimenez-Romero

Short description

In order to get a model of a complex system to exhibit a particular desired dynamics, it is necessary to tune its parameters. Typically the set of parameter configurations resulting in the desired dynamics is a small - and possibly non-contiguous - region of the total configuration space. When the complex system of interest is a neural network, an additional complexity is that the network should not simply exhibit a particular dynamics, but should be able to solve a given task or set of tasks. Due to the high dimensionality of the configuration space (minimally: all connection strengths and delays) and the lack of heuristics connecting parameter configurations to network performance, locating good (let alone optimal) parameter configurations is an intractably hard problem.
The framework *Learning to Learn* (L2L) enables a researcher to explore the parameter configuration space and to optimize models in an automated fashion. The performance of the model is evaluated based on a fitness function, which produces a fitness value. The parameters are optimized according to the fitness using optimization methods such as Genetic Algorithms. This procedure is iterated until convergence is reached.
However, the relationship between the evolved/optimized parameters and the observed fitness is not clear. Thus, there is a need to find and extract parameter features which correlate with the fitness, leading to a reduced feature space. As different feature extraction methods can be applied, a visualization of the fitness-feature relationship can help to compare the different methods. In this project an exploration of different **feature extraction strategies and analytical approaches** (e.g. information theory, approximate inference) combined with **visulaization techniques** are applied to an example data set. Is it possible to detect parameter features that correlate with the fitness?

Experimental setup and data description

The spiking neural network (SNN) depicted in Fig. 1a is used to navigate an agent which is searching for food in a maze (Fig. 1b). While the agent (yellow ant) scavenges for food (green dots) it has to avoid harmful objects (red dots) or the walls (white).
As depicted in Fig. 1a, the network includes six input neurons numbered from 11 to 16. These input neurons provide stimulation to the next (middle) layer of neurons numbered from 20 to 25. The neurons in the middle layer are recurrently connected in an all-to-all fashion, and to a third layer of neurons numbered from 30 to 31. The output neurons 30 and 31 serve as motor neurons, controlling the forward movement and rotation of the agent.
The **data set** consists of 96 csv files. Each file corresponds to an individual agent from the final generation of an L2L run using a genetic algorithm for optimization. Each csv file contains two rows of 78 values separated by commas, and a third row with a single value.

- The first row contains the weights of the 78 connections between neurons in the network. The range of the values is from -20.0 to 20.0
- The second row contains the transmission delays of the 78 connections between neurons in the network. The range of the values is from 0.0 to 10.0
- The third row contains the fitness value of the individual. The higher the value, the better the performance.

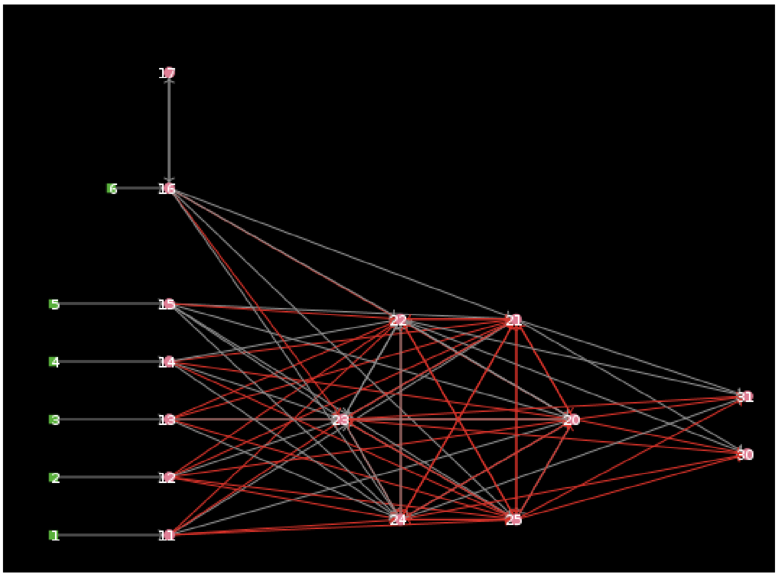

Figure 1a. A spiking neural network.

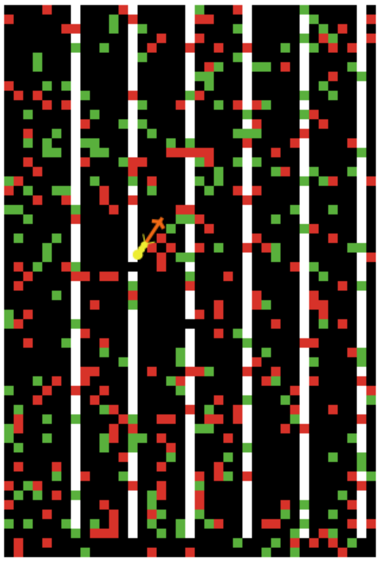

Figure 1b. A yellow ant is searching for food in a maze.

The **mapping** of the 78 connections is carried out as follows (in pseudo-code):

```
read_from_csv weights_row
read_from_csv delays_row
index = 0
For neuron_L1 in range(0 to 5):
   For neuron_L2 in range(0 to 5):
       ConnectFromTo(11 + neuron_L1, 20 + neuron_L2, weights_row[index], delays_row[index])
       index = index + 1

For neuron_L2_from in range(0 to 5):
   For neuron_L2_to in range(0 to 5):
       if neuron_L2_from != neuron_L2_to:
           ConnectFromTo(20 + neuron_L2_from, 20 + neuron_L2_to, weights_row[index], delays_row[index])
           index = index + 1

For neuron_L2 in range(0 to 5):
   For neuron_L3 in range(0 to 1):
       ConnectFromTo(20 + neuron_L2, 30 + neuron_L3, weights_row[index], delays_row[index])
       index = index + 1
```

Figure 1a. Mapping of the 78 connections.

The basic challenge

Since the parameter space is high-dimensional (in total 156 dimension: 78 connections, each with weight and delay) it is not possible or desirable to plot all parameters simultaneously. Additionally, there may be a linear or non-linear relationship between the fitness and the specific parameter combinations.
The first approach to address this challenge is to reduce the dimensionality of the data with the help of principal component analysis (PCA). PCA projects the data into a lower dimensional sub-space preserving essential parts of the data in terms of its variance. We suggest to plot the sub-space that is defined by the first three (and subsequent) principal components of the data combined with a color coded fitness (see [7]). Is there a relation between principal components and the fitness of the data?
As an alternative to this general dimensionality reduction method, we suggest you check whether some direct features of the data are more meaningful. Extract the mean weights of the three different connection types of the network (layer 1 to layer 2, within layer 2, layer 2 to layer 3; see Fig. 1a) and plot them in an analogous manner to the PCA plot. Which of the two methods works better?

After becoming acquainted with the data set using these two different methods, the challenge can be extended in several directions in order to get as much insight as possible into how the parameter configuration influences the fitness. Firstly, you can explore alternative feature extraction methods in a systematic manner. Secondly, more advanced plotting methods such as heat maps combining two or more PCA, may facilitate a comparison between the different approaches. Thirdly, an analysis based on information theory, e.g. entropy methods, might give further insight in the relationship of parameter features and fitness. Fourthly, an exploration of the underlying parameter distributions using inference strategies could be linked to the fitness outcome. Finally, you may think of an entirely different way to analyse and visualize the relationship between structural data and fitness.

Status quo

The relationship between network parameters such as connectivity weights and the network performance, e.g. with respect to the navigation in a maze, is largely unknown. More insight into this question would not only allow for a better understanding of how the network structure affects the network performance, it would also allow heuristics to be developed to accelerate the search for high-performing parameter configurations [3, 5].
Several approaches are known that can be used to reduce the dimensionality of a system and thereby extract the most informative features. Common methods include independent component analysis (ICA), principal component analysis (PCA) and autoencoders. Further methods are reviewed and applied to different contexts in [4, 6]. The relation between dimensionality reduction and visualisation is explained in e.g. [1, 2]. Importantly, although various ways of visualizing high dimensional data in a low dimensional space are known, the additional information of how the data relates to the fitness still needs to be visually integrated.

The real challenge

What makes the fittest individual different in comparison to the other individuals regarding the parameter space? Can this difference be quantified in a way that would provide a metric for a search heuristic? Could you predict from a given set of parameters which individuals are going to have the best fitness using your applied methods?

References

[1] Jeremie Bresson. Visualizing high-dimensional data. `https://sop.tik.ee.ethz.ch/publicationListFiles/bres2009a.pdf`, 2008.

[2] Shusen Liu, Dan Maljovec, Bei Wang, Peer-Timo Bremer, and Valerio Pascucci. Visualizing high-dimensional data: Advances in the past decade. *IEEE Transactions on Visualization and Computer Graphics*, 23(3):1249–1268, 2017.

[3] Abdirashid Nur, Nor Radzi, and Ashraf Osman. Artificial neural network weight optimization: A review. *TELKOMNIKA*, 12, 09 2014.

[4] Laurens Van Der Maaten, Eric Postma, Jaap Van den Herik, et al. Dimensionality reduction: a comparative. *J Mach Learn Res*, 10(66-71):13, 2009.

[5] Yasunori Yamada and Tetsuro Morima. Weight features for predicting future model performance of deep neural networks. In Subbarao Kambhampati, editor, *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence, IJCAI 2016, New York, NY, USA, 9-15 July 2016*, pages 2231–2237. IJCAI/AAAI Press, 2016.

[6] `https://www.sciencedirect.com/topics/computer-science/feature-extraction` [7] `https://drzinph.com/pca-visualized-with-3d-scatter-plots/`