On my search for locally activating neurons and its application in interpretability

Suman Sapkota [1]

1. Introduction

This document contains the summary of my search for locally activating neuron. I primarily use 2-dimensional input space to visually explain the findings along with some equations. First, I explain the background on the problems, the solutions explored, and finally the solution: distance/metric based ϵ-softmax neuron, which captures the essential properties required for locally activating neuron, and its application for neuron disentanglement and interpretability.

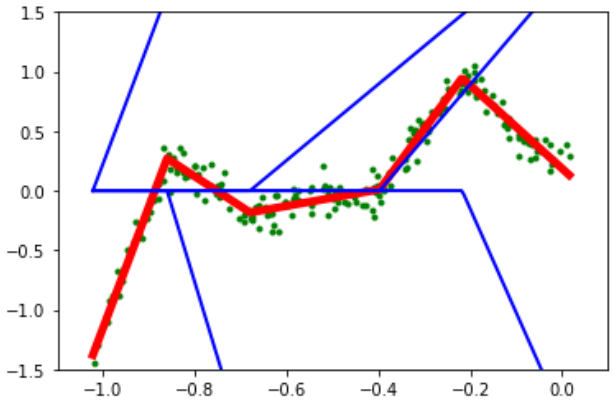


Figure 1. 1D regression dataset showing a non-linear function constructed by superposing multiple rectified linear units with various slopes.

2. Background

Initially, I was interested in understanding the underlying working mechanism of various Machine Learning algorithms, particularly of Artificial Neural Network (ANN), which lead me to work on Neural Network Architectures. I worked on understanding the simplest form of 1D dataset for regression (Figure 1). I found that due to global influence of ReLU neurons I could not easily add or remove neurons without causing large change to the learned function. This motivates the search for locally activating neuron, which could only affect a local region, and would not affect the global function.

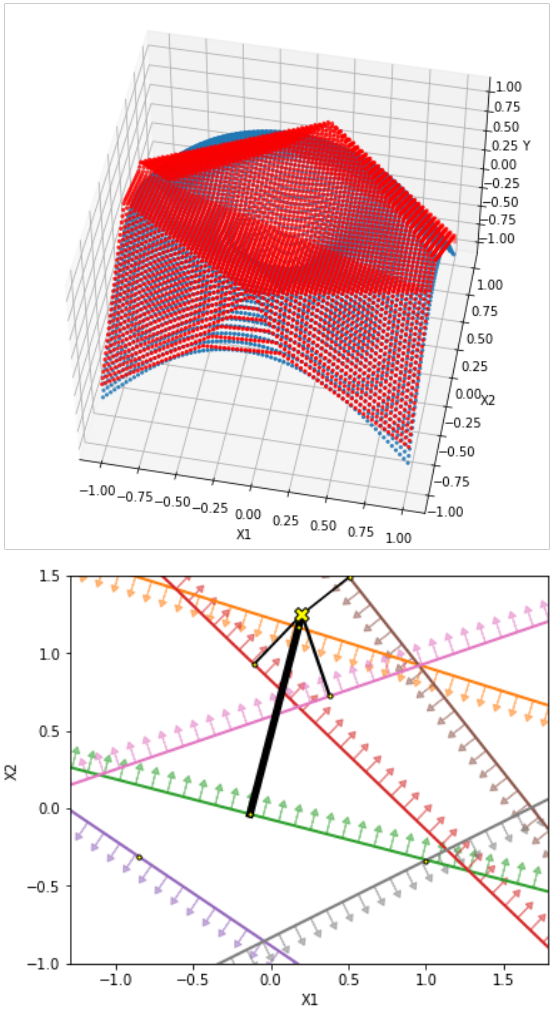


Figure 2. Top: 2D regression dataset showing approximation with superposition of multiple ReLUs in 2 layer MLP. Bottom: region of neuron firing and its magnitude for an input data point ($\times$) represented by the thickness of line.

[1]Independent Researcher. Correspondence to: Suman Sapkota <natokpas@gmail.com>.

This is a write-up of preliminary work and has not been peer-reviewed or published publicly. This document contains reference links to the repository of jupyter notebooks containing development of this research idea.

2.1. Network Morphism and Pruning

While tackling the problems with higher than 1 input dimensions, the ReLU neuron seems to have similar effect of intersecting with other neurons, producing highly entangled function. And, removing neurons with large slope would highly degrade the whole function, requiring finetuening.

In the piecewise function in Figure 2, we can see the effect of a neuron in a region. The far away the sample from the decision boundary, the more it activates. Moreover, the activation region of a neuron is highly superposed with others.

There could be some function to define a local region which could be modified without effecting the whole of data space. This was very similar to Radial Basis Function(RBF), and I quickly realized that RBF had the benefit of locally activating around a center, which I found very interesting for what I was looking. I also found similar idea mentioned in a blog post (Olah, 2014).

Why Locally Activating Neuron ? As shown in the Figure 2, the effect of neurons are global, and when trying to modify a small part, we can add new neurons. But to fit a new neuron properly, other neurons must adjust as well (fine-tuning). However, a locally activating neuron (like RBF) do not effect regions far away from centers, making it suitable to modify a small region. But we later find that, we need to use softmax function for disentanglement where radial basis function does not produce desirable property.

2.2. Metrics/Distance

The local effect of metrics like l^p-norm (and cosine-angle on a spherical input space) produce locally bounded distance and similarity. The general dot-product on euclidean space is not a metric in a sense that it does not produce bounded distance or similarity.

With metrics, I was able to compute similarity of input data with reference points (centers or weights). However, with dot product, the output depends not only on cosine-angle but also on the magnitudes, which cause neurons with large weights to activate higher despite the angle (reference work).

The application of l^1 (Chen et al., 2020) and l^2 (Li et al., 2022) norms in Neural Networks has been already explored for efficiency reasons. My approach lies in using metrics for local activation of neurons, which can further be used to create interpretable architectures.

Dictionary Learning: One of the ways to create mapping of input to output can be done using Dictionary.

$$Y = \text{similarity}(Q, K).V$$

Key - The reference that stores the Value
Value - The respective mapping value of some Key
Query - The input that is matched with Key to get the interpolated Value.

Softmax for normalized similarity: The softmax function has been used widely to create probability distribution given un-normalized inputs. Use of softmax for interpretable features is not new. It has been used in MLP layers for interpretability (Sukhbaatar et al., 2015), and also in the Attention layer itself where, the K, Q, V are dependent on data. Moreover, SoLU (Elhage et al., 2022a) also uses softmax to create interpretable neurons.

$$Y = \text{softmax}\{\text{similarity}(Q, K); \tau\}.V$$

Where, τ is a temperature to change the hardness of softmax activation. This is useful to create activations that are unimodal, disentangled or activate very sharply for certain Keys.

3. Distance Transform based ϵ-softmax Neurons

Using metrics/distance as transform, I was able to create neurons that activate locally.

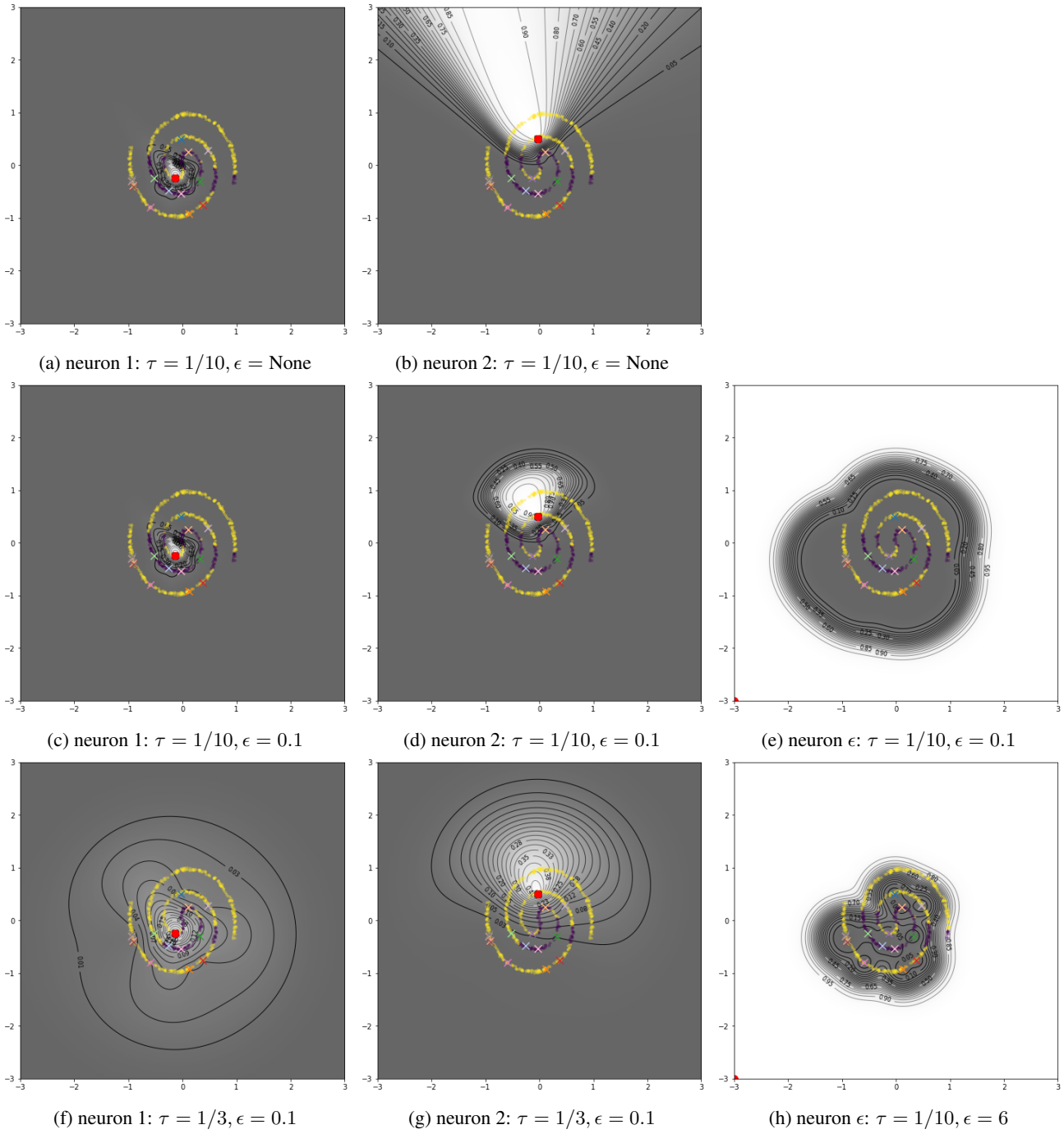
$$A = \text{softmax}\{-\text{distance}(W, X); \tau\} \tag{1}$$

Here, negative distance is equivalent to similarity not normalized to sum to 1.

Upon close inspection, neurons can produce high activation in region very far away from the reference centers (W) under certain condition of temperature and position of other centers (see Figure 3). The activation grows radially in direction of center from other centers, which is undesirable to create a neuron with local influence.

0-softmax: I was facing the above mentioned problem when I found about 0-softmax (Miller), where the intuition is that 0 entry is added to the softmax probability calculation so that when similarity measure $(Q.K)$ do not match, it does not normalize attention score to 1. Hence, it allows the attention to attend nothing as well, instead of ordinary softmax that sum to 1.

ϵ-softmax: This is my interpretation and modification of 0-softmax applied to distance based neuron. I found that I can set a value of epsilon when calculating neuron activation A (equation 1). The epsilon(ϵ) value sets a threshold on similarity, and if the similarities goes below the threshold, the epsilon neuron produces the highest activation. Hence we can use the epsilon neuron to represent region where the similarity to any Key is low. (reference notebook)



(a) neuron 1: $\tau = 1/10, \epsilon = \text{None}$ (b) neuron 2: $\tau = 1/10, \epsilon = \text{None}$

(c) neuron 1: $\tau = 1/10, \epsilon = 0.1$ (d) neuron 2: $\tau = 1/10, \epsilon = 0.1$ (e) neuron ϵ: $\tau = 1/10, \epsilon = 0.1$

(f) neuron 1: $\tau = 1/3, \epsilon = 0.1$ (g) neuron 2: $\tau = 1/3, \epsilon = 0.1$ (h) neuron ϵ: $\tau = 1/10, \epsilon = 6$

Figure 3. Visualization of region of neuron firing in toy 2D setting. *TOP:* without ϵ-neuron (or $\epsilon = -\infty$). *MID:* same setting but with ϵ-neuron. *BOT: (f, g)* with increased temperature, *(h)* with higher epsilon value for high ϵ-neuron activation.

This is the function that I was searching for! The one with local activation and can be used to change function without having global consequences.

Application of Local Neurons for Residual MLP: We can make use of locally activating neuron to create locally shifting residual function.

$$A = \epsilon - \text{softmax}\{-\text{distance}(W, X); \tau\}$$
$$Y = X + A.V \tag{2}$$

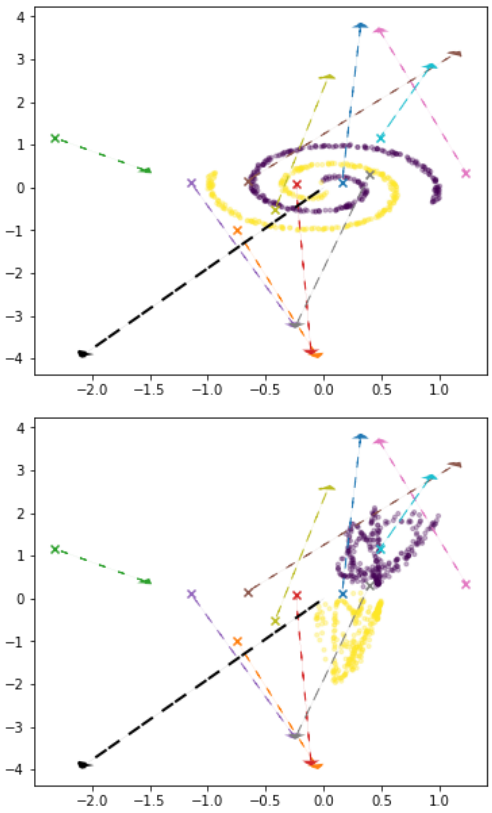


Figure 4. The centers($\times$) represent the weights (W) and the corresponding vectors (arrows) represents shift of region represented by the neurons. After the residual stream, the samples are shifted so that they are linearly separable. *TOP: input space, BOT: output space.* The black line represents shift for epsilon neuron.

Figure 4 shows local residal MLP. Due to small temperature, the highest activation of a neuron $<< 1$, i.e. the activation A is smooth and hence overall shift is also smooth. Importantly, we are able to interpret the non-linear classification of 2-spirals dataset using the concept of centers, similarity and shift (reference notebook).

Application of Local Neurons for MLP: Similar to the equation above, we can create an MLP using local neuron of form:

$$Y = \epsilon - \text{softmax}\{-\text{distance}(W, X); \tau\}.V \tag{3}$$

, where the Value directly points to the required output such as class information rather than shifting the residual stream.

Application for Classification: Similar to local activation for neuron, we can create local classification neuron. Rather than outputting corresponding value to certain key, it just outputs logits activation for N-class classification. The region of epsilon can also be used to reject outliers. The Figure 5 shows the region of class represented by local neuron.

Initialization with data points: We can easily initialize the weights/centers of distance layer with data samples and initialize its respective values with the target class. I was able to get very good accuracy just by selecting the weights and values from the dataset; and better with noisy selection (reference notebook).

Application of ϵ-softmax for Uncertainty Estimation and Adversarial Rejection: We aim to make use of "Locally Activating Neurons" of epsilon-Softmax to detect adversarial examples. To start at this problem, we may want to tackle uncertainty estimation. Using the epsilon neuron, we can calculate the uncertainty represented. The higher the epsilon activation, the farther the input is from known centers (W). The prediction is maximally certain when the data point x is the same as one of the W. However, if we input random sample, or adversarial example, then it has higher chance to activate epsilon(ϵ) neuron.

Experiment on MNIST: (reference notebook) With experiments on 2 layer MLP as shown in Equation 3, I tried to reject random samples. I was able to do it easily by checking if the epsilon(ϵ) neuron was firing.

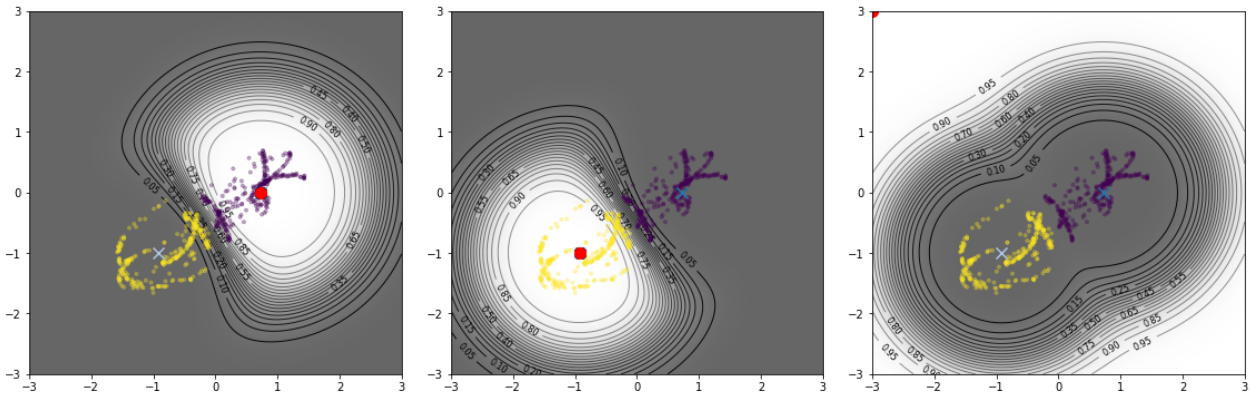
Moreover, I tested if the ϵ-neuron would fire highest for adversarial examples, and after tuning value of epsilon, I was able to reject about 50% of adversarial examples. It also rejected some of the inputs ($\sim 6\%$), nevertheless, it seems like a big success for me.

With later experiments, I found that if I were to allow centers(W) to change drastically from data points (possible Xs), or if the minimum distance between centers and data points is large, the MLP can still predict classes well, however, it did not succeed in rejecting the adversarial examples.

Although there is lot to understand, I believe the concept of using data point dependent W and V is important to understand and reject adversarial examples, as well as quantify uncertainty.

4. Discussion

I have used the above described techniques - namely distance transform and ϵ-softmax to achieve locally activating



(a) Class 0 (b) Class 1 (c) Class ϵ (no class)

Figure 5. Local neurons for classification: neurons representing different classes.

neurons and tested it in a toy 2D classification and MNIST dataset. I have yet to apply this to interpret different datasets and modalities. Moreover, I do not have a straight-forward path to use this method to interpret in Convolutional Neural Networks(CNN). However, due to high similarity with Attention (in terms of softmax based dictionary), I believe above method can be used to interpret and modify Attention Layer.

Relation to previous works on Interpretability: "Toy model of superposition" (Elhage et al., 2022b) works on similar idea of disentangling representations and effect of superposition. I was able to use softmax to disentangle activation regions, and control the unimodality with temperature. In cases where neurons/regions $>>$ input dimension, multiple neurons activate highly (or entangle), here, temperature is decreased for activating only highest similarity neurons.

Moreover, use of metrics allow us to use weights that can match the input, hence we can easily visualize the weights/centers and even initialize the parameters with data. However, I have yet to make this work with CNN, Deep Neural Networks and Transformers. I hope to work on that in future.

I was also able to use repetitive local neuron addition and pruning to optimize the 2 layer local-MLP and local residual-MLP on 2D classification and MNIST dataset.

References

Chen, H., Wang, Y., Xu, C., Shi, B., Xu, C., Tian, Q., and Xu, C. Addernet: Do we really need multiplications in deep learning? In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 1468–1477, 2020.

Elhage, N., Hume, T., Olsson, C., Nanda, N., Henighan, T., Johnston, S., ElShowk, S., Joseph, N., DasSarma, N., Mann, B., Hernandez, D., Askell, A., Ndousse, K., Jones, A., Drain, D., Chen, A., Bai, Y., Ganguli, D., Lovitt, L., Hatfield-Dodds, Z., Kernion, J., Conerly, T., Kravec, S., Fort, S., Kadavath, S., Jacobson, J., Tran-Johnson, E., Kaplan, J., Clark, J., Brown, T., McCandlish, S., Amodei, D., and Olah, C. Softmax linear units. *Transformer Circuits Thread*, 2022a. https://transformer-circuits.pub/2022/solu/index.html.

Elhage, N., Hume, T., Olsson, C., Schiefer, N., Henighan, T., Kravec, S., Hatfield-Dodds, Z., Lasenby, R., Drain, D., Chen, C., Grosse, R., McCandlish, S., Kaplan, J., Amodei, D., Wattenberg, M., and Olah, C. Toy models of superposition. *Transformer Circuits Thread*, 2022b. https://transformer-circuits.pub/2022/toy_model/index.html.

Li, X., Parazeres, M., Oberman, A., Ghaffari, A., Asgharian, M., and Nia, V. P. Euclidnets: An alternative operation for efficient inference of deep learning models. *arXiv preprint arXiv:2212.11803*, 2022.

Miller, E. Attention is off by one. https://www.evanmiller.org/attention-is-off-by-one.html. Accessed: May 2024.

Olah, C. Neural networks, manifolds, and topology. *Blog post*, 2014.

Sukhbaatar, S., Weston, J., Fergus, R., et al. End-to-end memory networks. *Advances in neural information processing systems*, 28, 2015.