Linear Order Statistic Neuron

Charlie Veal, *Student Member, IEEE,* Alex Yang, *Student Member, IEEE,* Alex Hurt, *Student Member, IEEE,* Muhammad Aminul Islam, *Member, IEEE,* Derek T Anderson, *Senior Member, IEEE,* Grant Scott, *Senior Member, IEEE,* Timothy C. Havens, *Senior Member, IEEE,* James M Keller, *Fellow, IEEE,* and Bo Tang, *Member, IEEE*

***Abstract*—Herein, a generalization of the ordered weighted average (OWA) is put forth relative to pattern recognition. The resultant linear order statistic neuron (LOSN) is unique in that it bridges fuzzy sets, specifically fuzzy data/information aggregation, with neural networks. This article discusses the gradient descent-based optimization and geometric interpretation of the LOSN. An advantage is that the LOSN is an efficient shared weight encoding of $N!$ perceptrons, relative to N inputs. Open source codes are provided to facilitate reproducible research. Experiments are conducted to both validate the method and show its non-linear geometric expression.**

***Index Terms*—neural networks, perceptron, linear order statistic, ordered weighted average**

I. Introduction

It is an undeniable fact that *neural networks* (NNs) have shown remarkable results, in recent times, relative to tasks like classification and regression across numerous fields, e.g., computer vision and natural language processing. However, the fame of these models is commonly associated with state-of-the-art architectures–e.g., Xception [1], NasNet [2], ResNet [3], etc.–and less with the exploratory development of the underlying mathematics that power them. From a theory perspective, many common limitations of these approaches can be traced back to the fundamentals. For example, nearly all modern solutions are based on the perceptron and convolution. Herein, we investigate an alternative: A neuron based on fuzzy set theory.

The field of fuzzy sets is no stranger to NNs. In [4], Keller and Hunt investigated a fuzzy perceptron. In 1992 [5], Yager put forth an *ordered weighted average* (OWA) [6] neuron [5]. Specifically, Yager discussed the derivative of an OWA, a pre-requisite for gradient descent solvers [5]. In [7], Verma investigated fuzzy inference neurons. In 1995, Sung-Bae utilized the OWA for NN aggregation at the decision/output level [8]. In 1995, Sung-Bae et al. also explored the Sugeno fuzzy integral for NN aggregation [9]. In 2017 [10], [11], Scott et al. used the Sugeno and Choquet integrals for deep convolutional neural network fusion. The reader can also refer to the work of Pal and Mitra [12] for neuro-fuzzy pattern recognition. In [13], Won et al. proposed mathematical morphology, specifically a shared weighted hit-and-miss transform.

Charlie Veal is with the Electrical Engineering and Computer Science (EECS) Department, University of Missouri (MU), MO, USA e-mail: ctvqfq@mail.missouri.edu. Derek Anderson, Grant Scott, James Keller, Alex Yang, and Alex Hurt are with the University of Missouri, USA, Timothy Havens is with Michigan Technological University, MI, USA, and Muhammad Aminul Islam and Bo Tang are with Mississippi State University, USA, MS.

TABLE I: Acronyms and Notation

OWA	Ordered weighted average
LOS	Linear order statistic
LOSN	LOS neuron
$\mathbf{x} = (x_1, x_2, ..., x_N)^t$	Inputs, s.t., $x_i \in \Re$
$\mathbf{x}_\pi = (x_{\pi(1)}, x_{\pi(2)}, ..., x_{\pi(N)})^t$	Sort s.t. $x_{\pi(1)} > ... > x_{\pi(N)}$
$\mathbf{w} = (w_1, w_2, ..., w_N)^t$	Weight vector
$\mathbf{w}_\pi = (w_{\pi(1)}, w_{\pi(2)}, ..., w_{\pi(N)})^t$	Sort s.t. $w_{\pi(1)} > ... > w_{\pi(N)}$
$f_1(\mathbf{x}; \mathbf{w}) = \mathbf{w}^t\mathbf{x} + w_{N+1}$	Perceptron
$f_2(\mathbf{x}; \mathbf{w}) = \mathbf{w}^t\mathbf{x}_\pi$	LOS
$f_3(\mathbf{x}; \mathbf{w}) = \mathbf{w}^t\mathbf{x}_\pi + w_{N+1}$	LOSN

Herein, we focus on the following questions. First, we know that it is possible to take the derivative of an OWA and that an OWA can be used in a NN [5]. However, what is it doing and how does that compare to the perceptron? Second, is it possible to integrate an OWA into a context like deep learning?

This paper makes the following specific contributions. First, it explores the class of problems involving real-valued inputs and real-valued weights and the resultant *linear order statistic neuron* (LOSN). Second, this paper provides a geometric interpretation for the LOSN, and explores a comparison between it and the perceptron. Third, this paper shows how to optimize the LOSN and its corresponding implementation through open source python code (which is made freely available at https://github.com/charlieveal/LOSN). Ultimately, the aim of this paper is to advance NNs using fuzzy set theory and to advance fuzzy sets by looking at data/information aggregation from a different perspective.

The remainder of the article is organized as follows. First, the perceptron is reviewed in section II. Next, the OWA and LOS are reviewed in section III. Subsequently, a generalization of the LOS is put forth, in section IV, called the LOSN. Implementation details and experiments follow. Table I summarizes acronyms and notation.

II. Perceptron

The famous linear discriminate model of pattern recognition is the perceptron. Formally introduced by Warren S. McCulloch and Walter Pitts [14], the perceptron was originally purposed to solve arithmetic and logical functions. However, Frank Rosenblatt [15] advanced this model by optimizing it with a training algorithm. Formally, the perceptron is

$$f_1(\mathbf{x}; \mathbf{w}) = w_{N+1} + \sum_{i=1}^{N} w_i x_i = \mathbf{w}^t\mathbf{x} + w_{N+1}, \qquad (1)$$

where $f_1 = 0$ is a decision boundary optimized usually by gradient descent to invoke a binary classification; $f_1(\mathbf{x}; \mathbf{w}) \geq$

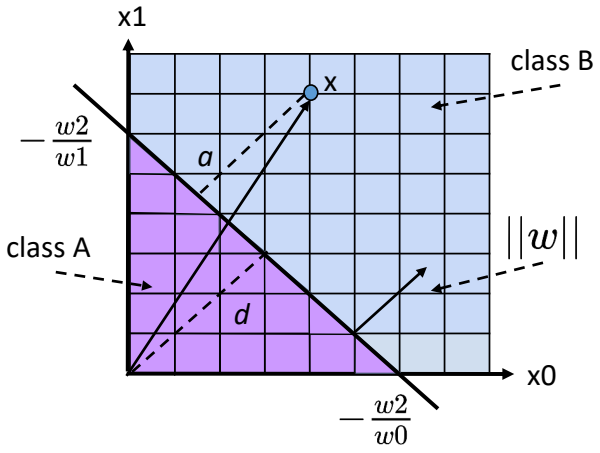


Fig. 1: Geometrical interpretation of perceptron for $N = 2$. The y and x intercepts are $-\frac{w_2}{w_1}, -\frac{w_2}{w_0}$. The color blue shows class B ($f_1 \geq 0$) and purple is class A. Orthogonal to the decision boundary is its normal, $\mathbf{w}$, which points towards class B. Lastly, a and d are the Euclidean distance from a sample point in space, x, to the decision boundary and the origin to the decision boundary, respectively.

0 means class 1, class 2 otherwise. When trying to solve an N-dimensional space problem, the perceptron creates an N-1 dimensional hyper plane that separates the problem space into two sub-planes. These sub-planes represent the binary classes of interest (see Figure 1).

III. Ordered Weighted Average

There are countless challenges that require the intelligent combining of information to formulate a decision; smart cars, drones, remote sensing, etc. To this end, one such mechanism is aggregation. An aggregation is a mapping $f(\mathbf{x};\theta) \in \mathbf{y}$, where θ are function parameters and $\mathbf{y}$ is usually a real valued number. A famous information aggregation function in the fuzzy set community is the OWA. This tool was introduced by Yager in the context of *multi-criteria decision making* (MCDM) [6],

$$f_2(\mathbf{x};\mathbf{w}) = \sum_{i=1}^{N} w_i x_{\pi(i)} = \mathbf{w}^t \mathbf{x}_\pi, \tag{2}$$

where π is an ordering such that $x_{\pi(1)} \geq x_{\pi(2)} \geq ... \geq x_{\pi(N)}$, $\mathbf{x}_\pi$ is the sorted input, $w_i \geq 0$, and $\sum_{i=1}^{N} w_i = 1$.

Example 1. Let $N = 3$, $\mathbf{x} = (4, 1, 3)^t$ and $\mathbf{w} = (0.7, 0.3, 0)^t$ (a soft max). The OWA is $0.7(4)+0.3(3)+0(1) = 3.7$, which is very close to 4, the largest input.

In this paper, a constraint is imposed on the OWA such that both the weights and inputs are real valued instead of fuzzy set valued. In the community, this often goes by the name *linear order statistic* (LOS). While the LOS is a simple operator, it provides a range of useful aggregations, from t-norms (intersection like) to t-conorms (union like), see Figure 2. Examples include the max ($w_1 = 1$, $w_k = 0$ for $k > 1$), min ($w_N = 1$, $w_k = 0$ for $k < N$), mean ($w_k = \frac{1}{N}$), median, soft versions, trimmed statistics, and much more.

Whereas the context of this section is information aggregation, in the next section we consider the geometric interpretation of a LOS.

IV. Linear Order Statistic Neuron (LOSN)

The purpose of this section is to explore the ideology of replacing the perceptron with an OWA, or more specifically the LOS for NNs. Traditionally, an OWAs weights are constrained such that $w_i \in [0, 1]$ and $\sum_{i=1}^{N} w_i = 1$. Semantically, this make sense on domains like MDCM, whose inputs are also in $[0, 1]$. However, this ends up becoming a hindrance; a restriction with respect to the number of problems that one can solve regarding NNs. As such, the following is a subtle, yet useful, extension of the LOS, referred to hereafter as the *LOS neuron* (LOSN),

$$f_3(\mathbf{x};\mathbf{w}) = \mathbf{w}^t \mathbf{x}_\pi + w_{N+1}. \tag{3}$$

Geometrically, the LOSN can be visualized (and characterized) in $\Re^N$ space. In N dimensions the LOSN has $N!$ "regions", one for each permutation (π_i). Above, this paper introduced the notation $\mathbf{w}^t\mathbf{x}_\pi$. This means the weights stay fixed but the input is sorted. Alternatively, one can sort the weights and keep the inputs fixed, i.e., $\mathbf{x}^t\mathbf{w}_\pi$. Thus, a LOSN can be thought of as $N!$ "shared weight" perceptrons. This means there are $N!$ actual perceptrons, but they all share the same weights.

Example 2. Consider $N = 2$, see Figure 3. The left-to-right axis is x_1 and the up-to-down axis is x_2. There are $N!$ sorts, $x_1 > x_2$ (green area in Figure 3) and $x_2 > x_1$ (orange area in Figure 3). In Figure 3, the red line is a possible linear decision line (aka perceptron) for $x_1 > x_2$, i.e., $\mathbf{x}^t\mathbf{w}_{\pi_1}$. Respectively, the green line is $\mathbf{x}^t\mathbf{w}_{\pi_2}$ for $x_2 > x_1$. Note, in both cases the dashed arrow, perpendicular to the decision lines, indicates the surface normal direction. In the case of a perceptron, this denotes the positive valued region, which often is used to pick a class in classification. For two sort regions and two perceptrons, there are four total regions, R_1 to R_4 in Figure 3. "Class 1" is the union of R_1 and R_3, and "Class 2" is R_2 and R_4. Last, the dashed black line denotes $x_1 = x_2$.

In the next few subsections we explore the factors (LOSN parameters) that drive important geometric properties.

A. Constraints: ($w_i \geq 0$, $\sum_{i=1}^{N} w_i = 1$, $w_{N+1} = 0$)

This subsection explores the consequence of constraining a LOSN to act like a conventional OWA; aka, positive weights that sum to one and no bias. To avoid loss of generality, $N = 2$ will be considered in order to facilitate graphical illustration. The first example will be for the sake of understanding.

Example 3. Let $\mathbf{w} = (0.5, 0.5)^t$. As such, the $N! = 2$ subspaces ($x_1 > x_2$ and $x_2 > x_1$) have the perceptron $y = 0.5x_1 + 0.5x_2$. This results in an x-axis intercept of $-\frac{w_3}{w_1} = 0$ and an y-axis intercept of $-\frac{w_3}{w_2} = 0$; a.k.a the perceptron passes through the origin. In both regions the normal direction is $(0.5, 0.5)$. The corresponding LOS

Max	1	0	0	0	0	0	0	0	0	0
Soft Max	0.7	0.2	0.1	0	0	0	0	0	0	0
Mean	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1
Trimmed Mean	0	0	0.16	0.16	0.16	0.16	0.16	0.16	0	0
Median	0	0	0	0	0.5	0.5	0	0	0	0
Min	0	0	0	0	0	0	0	0	0	1
Weights	w_1	w_2	...	...	...	...	...	...	...	w_N

Fig. 2: Illustration of a few common OWAs/LOSs.

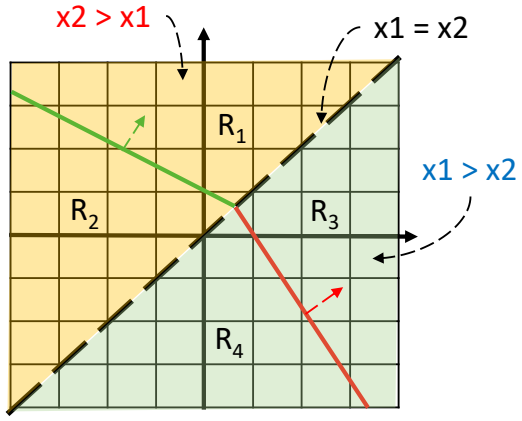


Fig. 3: LOSN for $N = 2$. See Example 2 for details.

decision boundary is illustrated in Figure 4(a). Blue indicates areas where the peceptrons return a positive value. Similar story holds for negative perceptron-valued regions. What is interesting about this example is that we can see that the LOSN reduces into a perceptron.

In general, interesting observations can be made for ($w_i \geq 0$, $\sum_{i=1}^{N} w_i = 1$, $w_{N+1} = 0$). First, due to the constraint $w_{N+1} = 0$, the intercepts are $-\frac{w_{N+1}}{w_i}$, are always zero. This means that the $N!$ shared weight perceptrons will always pass through the origin. Second, the constraint $w_i \geq 0$ restricts the "angular range" of perceptron surface normals. For example, we could not produce the same line with flipped class labels, i.e., make the blue region purple and vice versa. As stated earlier in this article, these are a few of the overly restrictive reasons why a LOSN was an ideal extension from the LOS.

B. Constraints: ($w_i \geq 0$, $\sum_{i=1}^{N} w_i = 1$, $w_{N+1} \neq 0$)

This subsection allows for non-zero bias terms. As before, a numeric example is provided first.

Example 4. Let $N = 2$ and $\mathbf{w} = (0.9, 0.1, 3)^t$, where w_3 is the bias. As such, $x_1 \geq x_2$ leads to $y = 0.9x_1 + 0.1x_2 + 3$ and $x_2 > x_1$ leads to $y = 0.9x_2 + 0.1x_1 + 3$. In $x_1 \geq x_2$, this leads to $y_i = -\frac{3}{0.1} = -30$ and $x_i = -\frac{3}{0.9} = -\frac{30}{9}$. The normal for $x_1 \geq x_2$ is $(0.9, 0.1)$. In $x_2 \geq x_1$, we get the reversal of intercepts, i.e., $y_i = -\frac{3}{0.9}$ and $x_i = -30$, and normal, i.e., $(0.1, 0.9)$. The decision boundary is shown in Figure 4(b).

Example 4 reveals an interesting geometric property of the LOSN. Namely, the perceptrons intersect on the equality line ($x_1 = x_2$). This is always the case due to the calculation of the intercepts and the fact that the sort just flips these values. In general, the perceptrons are $N - 1$ dimensional hyperplanes; aka lines in $N = 2$. Thus, one obtains nonlinear "wedges" for the LOSN due to the sort. This means that fewer LOSNs can be used, relative to perceptrons to achieve as strong as a result, which ultimately means more freedom.

C. Unconstrained LOSN

This subsection explores the unconstrained LOSN. As before, we begin with an example.

Example 5. Let $N = 2$ and $\mathbf{w} = (2, -4, -6)^t$. Note, the big difference here is that some weights are negative-valued. For $x_1 \geq x_2$ we get $y = 2x_1 - 4x_2 - 6$ and $x_2 > x_1$ leads to $y = 2x_2 + -4x_1 - 6$. In $x_1 \geq x_2$, we get $y_i = -\frac{-6}{-4} = -1.5$ and $x_i = -\frac{-6}{2} = 3$. The normal for $x_1 \geq x_2$ is $(2, -4)$. In $x_2 \geq x_1$, we get $y_i = 3$, $x_i = -1.5$, and a normal of $(-4, 2)$. The decision boundary is shown in Figure 4(c).

Example 5 is interesting for a few reasons. First, one is able to see the hyperwedge decision boundary. Second, it can be observed that the negative weights allow the shared weight perceptrons to "point" in any direction, i.e., full range for the hyperplane surface normals. Figure 4(c) shows this as the hyperwedge "pointing direction" has flipped.

Next, we look at how to learn the LOSN relative to gradient descent-based optimization.

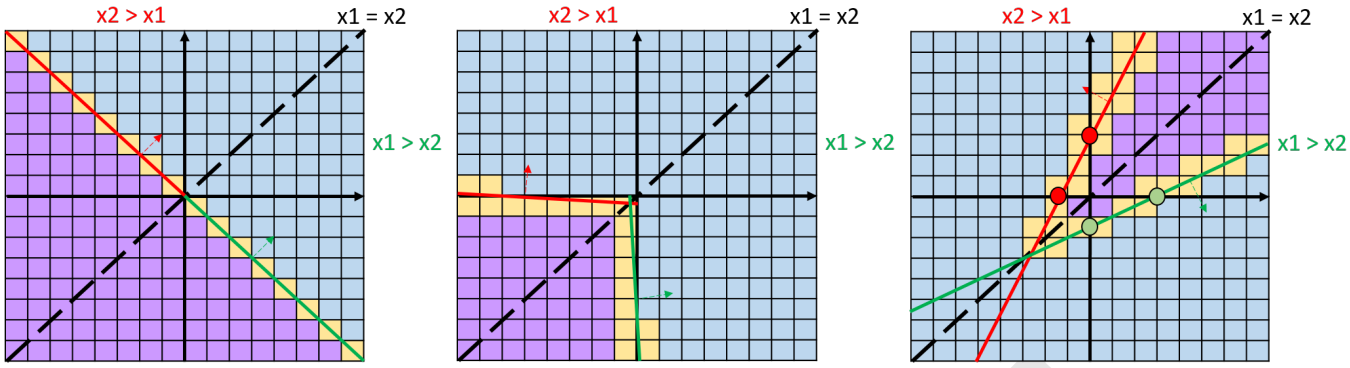
D. LOSN Optimization

The *sum of squared error* (SSE) and gradient descent-based optimization process for the LOSN is similar to that of the perceptron. Let ($\mathbf{x}_k \in O, l_k \in L$) be a labeled training dataset, where $M = |O|$. The subsequent SSE is

$$J = \frac{1}{2}\sum_{k=1}^{M}(f_3(\mathbf{x}_k; \mathbf{w}) - l_k)^2 \quad = \quad \frac{1}{2}\sum_{k=1}^{M} e_k^2 \tag{4}$$

where l_k is the true label, $f_3(\mathbf{x}_k; \mathbf{w})$ is the LOSN predicted value, $\mathbf{w}$ is the parameters to optimize, and e_k is the error between the two respectively. From here, one can calculate the gradient with respect to the coefficient of interest,

$$\frac{\partial J}{\partial w_i} = \frac{\partial J}{\partial e} * \frac{\partial e}{\partial f_3(\mathbf{x}; \mathbf{w})} * \frac{\partial f_3(\mathbf{x}; \mathbf{w})}{\partial w_i} \tag{5}$$



(a) Case 1 (b) Case 2 (c) Case 3

Fig. 4: Illustration of different decision boundaries for the LOSN and $N = 2$ (see text for full details). Black dotted line is the sort order line ($x_1 = x_2$), red line is the perceptron for $x_2 > x_1$ and the green line is the perceptron for $x_1 \geq x_2$. Light blue cells are "region/class one" (meaning LOSN > 0) and purple cells are for "region/class two" (LOSN < 0). Light yellow cells indicate need further division; since this figure is showing a continuous space using discrete cells. Lines in case (a) are at increments of $\Delta = 1$, case (b) is $\Delta = 5$, and case (c) is $\Delta = 1$. See Section IV for full details.

Nothing has truly changed between optimizing a perceptron. The only term of interest is $\frac{\partial f_3(\mathbf{x};\mathbf{w})}{\partial w_i}$. In particular, this term is interesting only because the LOSN does a sort. This is trivial to solve, i.e.,

$$\frac{\partial f_3(\mathbf{x};\mathbf{w})}{\partial w_i} = x_{\pi(i)}. \tag{6}$$

Remark 1. In implementation, it is simpler to pre-sort the data and keep the LOSN weights fixed (see Figure 5).

V. Implementation

In an effort to facilitate reproducible research, we are making all code available; see https://github.com/charlieveal/LOSN. Python has become the modern language of artificial intelligence due in part to the popularity of deep learning. We created two implementations: (1) LOSN using mainstream libraries like NumPy and MatPlotLib and (2) in the state-of-the-art NN library PyTorch.

VI. Experiments and Results

The goal of this section is to explore the LOSN via validation on a synthetic dataset and the famous XOR dataset. Synthetic data is ideal for experimentation because one is not only able to know the underlying target solution, but we also can control factors such as the number of features (N) and type of aggregation (labels).

A. XOR Dataset

The purpose of this subsection is to demonstrate that the LOSN can tackle a nonlinear classification task, namely the classical XOR dataset (see Figure 6). The XOR problem is one that has two points in class 1, $(1, 1)$ and $(-1, -1)$, and two points in class 2, $(1, -1)$ and $(-1, 1)$. Naturally, the perceptron cannot solve this problem; the XOR is not linearly separable. Instead, the perceptron needs to be extended, e.g., a multi-layer perceptron network. However, the LOSN has inherit nonlinear capabilities due to its sorting operation.

Figure 6(a) illustrates the perceptron and Figure 6(b) shows the LOSN solution. Referencing these results, one can see that the LOSN converges quickly and smoothly to an acceptable solution based on an arbitrarily selected initialization. On the other hand, the perceptron struggles in its attempt to solve a problem that it cannot. In summary, the LOSN for $N = 2$ has the capability of two perceptrons with shared weights. This results in a wedge shape whose nonlinearity is adequate to tackle the XOR.

B. Single LOSN and Synthetic Data

The purpose of this experiment is twofold. First, it shows that the LOSN can learn an aggregation from data. Second, it shows LOSN behavior with respect to dimensionality (N) and number of observations (aka samples).

Herein, four datasets of size 100, 500, 1000, and 10000 were randomly initialized from a uniform distribution over $[0, 1)$ with four different Ns; *small* = 100, *medium* = 500, *large* = 1000, and *massive* = 10000. Afterwards, the following LOSN aggregations were used to generate a corresponding set of training labels; max, soft min, and mean. For each set of labels, a LOSN is trained for each N. For training, the learning rate (λ) was fixed to $1e^{-4}$ to handle larger N. Each training cycle of the LOSN was run for 1000 iterations, where an iteration is giving the LOSN an observation of all samples from the entire training dataset. The method of measuring the models error is the sum squared error loss function. Results can be found in Table 1.

Table 1 shows that as N increases the problem quickly becomes more complex. Therefore, more samples are needed for the LOSN to tackle the problem. Table 1 also shows that the minimum samples to get a desirable result share a linear relationship with the N; e.g., N = 1000 requires 1000 samples minimum for low error. Lastly, only one table was shown

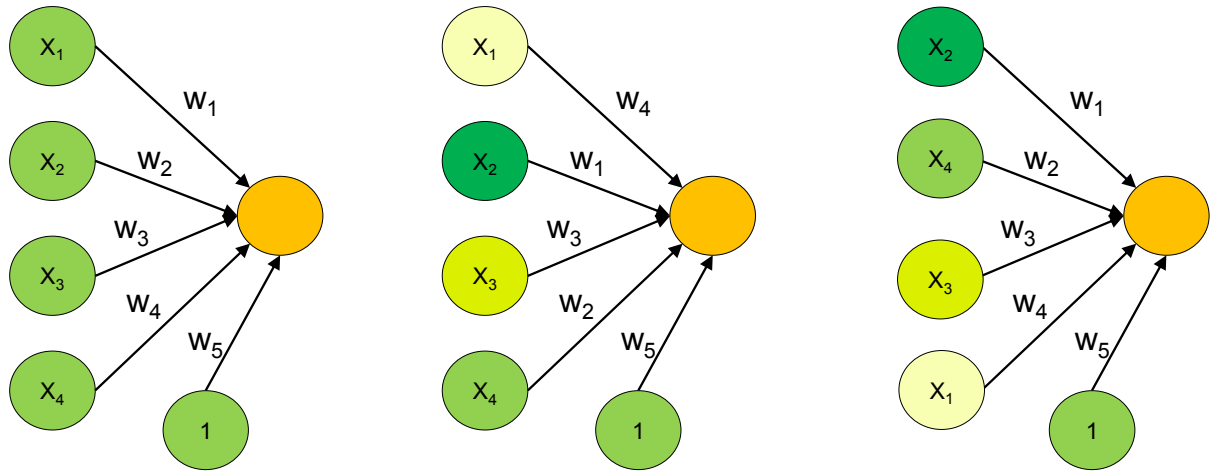
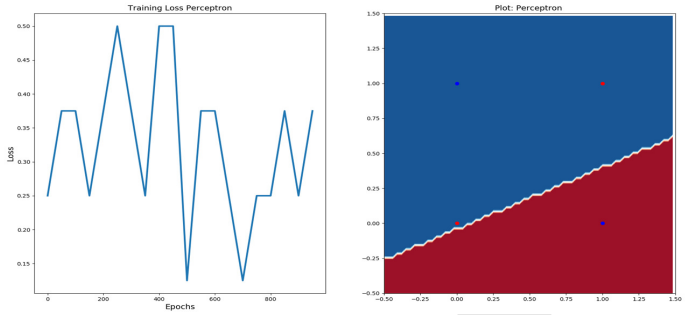
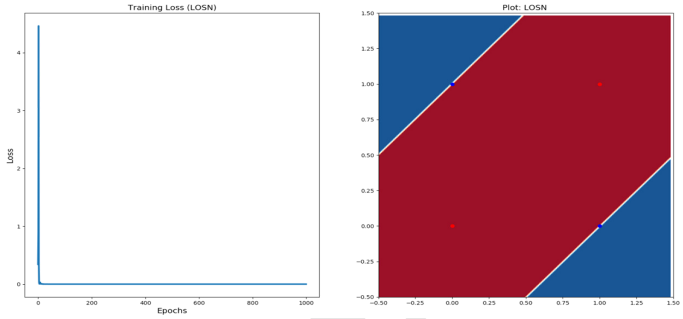


Fig. 5: Example LOSN for $N = 4$. (left) Four inputs and bias term. (middle) Case where $x_2 \geq x_4 \geq x_3 \geq x_1$ and the weights ($\mathbf{w}$) are reordered. (right) Simplification where inputs are reordered and weights (and subsequent gradients) can remain fixed.



(a) Perceptron



(b) LOSN

Fig. 6: (a) Perceptron versus (b) LOSN for the XOR dataset. The left images are error convergence plots and the right blue and red colored plots are the final decision boundaries.

TABLE II: SSE for Ground Truth of Mean

	Samples			
Dimensions	100	500	1000	10000
N = 100	0.05	0.005	0.002	0.0003
N = 500	0.27	0.016	0.006	0.0006
N = 1000	0.37	0.032	0.014	0.0006
N = 10000	5.28	0.44	0.17	0.005

out of the three aggregations because the results across the other synthetic experiments were more-or-less the same. This is not surprising because the labels are drawn from the same random generated data distribution the LOSN does not care which operator is needed, they are all the same "complexity". Overall, this experiment has shown that the LOSN can solve high dimensionality problems when given enough data. This is important because it means that the LOSN can fuse a large number of inputs, which is important in modern deep learning.

VII. Conclusions and Future Work

Herein, this paper introduced the *linear order statistic neuron* (LOSN), a generalization of the *ordered weighted average* (OWA) from the fuzzy set community. Specifically, the LOSN is $N!$ shared weight perceptrons. In terms of *neural networks* (NN), this means that one should be able to achieve more with less architectural overhaul, i.e shallower networks. This paper then illustrated that a LOSN is associated, geometrically, with a "hyper wedge" decision boundary. The experiments show that the LOSN can learn an aggregation, or solve some underlying decision boundary problem in the context of classification, over a range of sizes. The experiments also illustrated the LOSNs ability to solve non-linear problems.

In future work, we will explore the following. The purpose of this article was to understand and validate the LOSN. Next, we will extend the LOSN to deep learning for purposes like realizing shallower nets. Other goals include exploiting our knowledge about the OWA for *explainable artificial intelligence* (XAI). Furthermore, this article shows a "geometric take" on aggregation. Future work will focus on if there are any theoretical advantages in thinking about aggregation or NNs in terms of the other. Next, the LOSN requires a sort and future work will focus on a way to reduce its computational cost, either via an approximation or possible an alternative way of expressing the LOSN. Last, it is not clear where the biggest advantage is with respect to a LOSN in the context of NNs. Future work will explore the application of a LOSN at different levels in a NN, e.g., signal, feature, or decision.

REFERENCES

[1] François Chollet, "Xception: Deep learning with depthwise separable convolutions," *CoRR*, vol. abs/1610.02357, 2016.

[2] Barret Zoph, Vijay Vasudevan, Jonathon Shlens, and Quoc V. Le, "Learning transferable architectures for scalable image recognition," *CoRR*, vol. abs/1707.07012, 2017.

[3] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun, "Deep residual learning for image recognition," *CoRR*, vol. abs/1512.03385, 2015.

[4] J. M. Keller and D. J. Hunt, "Incorporating fuzzy membership functions into the perceptron algorithm," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. PAMI-7, no. 6, pp. 693–699, Nov 1985.

[5] Ronald R Yager, "Applications and extensions of owa aggregations," *International Journal of Man-Machine Studies*, vol. 37, no. 1, pp. 103–122, 1992.

[6] Ronald R Yager, "On ordered weighted averaging aggregation operators in multicriteria decisionmaking," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 18, no. 1, pp. 183–190, 1988.

[7] S. Rajurkar and N. K. Verma, "Developing deep fuzzy network with takagi sugeno fuzzy inference system," in *2017 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)*, July 2017, pp. 1–6.

[8] Cho Sung-Bae, "Fuzzy aggregation of modular neural networks with ordered weighted averaging operators," *International Journal of Approximate Reasoning*, vol. 13, no. 4, pp. 359–375, 1995.

[9] Sung Bae Cho and Jin H Kim, "Combining multiple neural networks by fuzzy integral for robust classification," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 25, no. 2, pp. 380–384, 1995.

[10] Grant J Scott, Richard A Marcum, Curt H Davis, and Tyler W Nivin, "Fusion of deep convolutional neural networks for land cover classification of high-resolution imagery," *IEEE Geoscience and Remote Sensing Letters*, 2017.

[11] G. J. Scott, K. C. Hagan, R. A. Marcum, J. A. Hurt, D. T. Anderson, and C. H. Davis, "Enhanced fusion of deep neural networks for classification of benchmark high-resolution image data sets," *IEEE Geoscience and Remote Sensing Letters*, vol. 15, no. 9, pp. 1451–1455, Sep. 2018.

[12] Sankar K Pal and Sushmita Mitra, *Neuro-fuzzy pattern recognition: methods in soft computing*, John Wiley & Sons, Inc., 1999.

[13] Yonggwan Won, P. D. Gader, and P. C. Coffield, "Morphological shared-weight networks with applications to automatic target recognition," *IEEE Transactions on Neural Networks*, vol. 8, no. 5, pp. 1195–1203, Sep. 1997.

[14] Warren McCulloch and Walter Pitts, "A logical calculus of the ideas immanent in nervous activity," *Bulletin of Mathematical Biophysics*.

[15] Frank Rosenblatt, "The perceptron: A probabilistic model for information storage and organization in the brain," *Psychological Review*.