Prototype-Based Interpretation of the Functionality of Neurons in Winner-Take-All Neural Networks

Ramin Zarei-Sabzevar, Kamaledin Ghiasi-Shirazi, and Ahad Harati

Abstract—Prototype-based learning (PbL) using a winner-take-all (WTA) network based on minimum Euclidean distance (ED-WTA) is an intuitive approach to multiclass classification. By constructing meaningful class centers, PbL provides higher interpretability and generalization than hyperplane-based learning (HbL) methods based on maximum inner product (IP-WTA) and can efficiently detect and reject samples that do not belong to any classes. In this article, we first prove the equivalence of IP-WTA and ED-WTA from a representational power perspective. Then, we show that naively using this equivalence leads to unintuitive ED-WTA networks in which the centers have high distances to data that they represent. We propose ±ED-WTA that models each neuron with two prototypes: one positive prototype, representing samples modeled by that neuron, and a negative prototype, representing the samples erroneously won by that neuron during training. We propose a novel training algorithm for the ±ED-WTA network, which cleverly switches between updating the positive and negative prototypes and is essential to the emergence of interpretable prototypes. Unexpectedly, we observed that the negative prototype of each neuron is indistinguishably similar to the positive one. The rationale behind this observation is that the training data that are mistaken for a prototype are indeed similar to it. The main finding of this article is this interpretation of the functionality of neurons as computing the difference between the distances to a positive and a negative prototype, which is in agreement with the BCM theory. Our experiments show that the proposed ±ED-WTA method constructs highly interpretable prototypes that can be successfully used for explaining the functionality of deep neural networks (DNNs), and detecting outlier and adversarial examples.

Index Terms—Adversarial examples, BCM theory, interpretability, neuron, prototype-based learning (PbL), winner-take-all.

I. INTRODUCTION

A NATURAL method for solving multiclass classification problems is prototype-based learning (PbL) in which one allocates several prototypes to each class and assigns input data to the class of the nearest prototype, as measured by a distance function. This kind of learning has several advantages over hyperplane-based learning (HbL) methods, such as Fisher linear discriminant analysis (LDA), support vector machines (SVMs), and the majority of neural networks.

The first advantage of PbL is high interpretability: the input is assigned to a class since it is similar to one of its prototypes. Interpretability of learning machines has recently become a significant subject of research [1], [2] and is of paramount importance in some areas, such as healthcare [3], criminal justice, and finance [4]. Specifically, designing inherently interpretable machine learning systems has become an active area of research. Li et al. [5] proposed a network architecture with a special prototype layer, allowing the network to explain its predictions. Neural additive models [6] combine the expressivity of DNNs with the inherent interpretability of generalized additive models by transforming each feature separately by a DNN and adding the outputs together. The second advantage of PbL is that it is suitable for few-shot learning [7], [8]. Usually, the average of a few similar samples can be used as a prototype with high generalization. The third advantage of PbL is that it provides a high-quality reject option since it is based on a distance function. Convolutional prototype networks [9], [10] use PbL for rejecting samples that are out of known classes.

Although margin-based methods such as [11], [12] provide a reject option based on the distance to margin, these methods can only reject ambiguous cases that necessarily belong to one of the classes. Since there is no reason that a clutter input, which belongs to none of the classes, would fall near the margin of a hyperplane, HbL methods cannot reject them. On the other hand, since it is unlikely that a clutter input would fall in a small distance to a prototype, PbL methods can easily reject them. Hence, measuring similarity based on a distance function, and not a general similarity measure such as inner product, is at the core of PbL. Consequently, PbL does not include hyperplane-based WTA classifiers, such as multiclass SVM [13] and neural networks with a softmax at their last layer.

Despite the aforementioned benefits of PbL, the most commonly used classification systems, such as SVM or neural networks, are HbL methods. We believe that this is because of two reasons. First, the recognition rates of the existing PbL methods are inferior to HbL methods such as deep neural networks (DNNs). Second, the relationship between these two classes of algorithms has not been studied in enough depth to offer necessary improvements from one area to the other. Few efforts have been made to explore the relationship between HbL and PbL. Graf et al. [14] demonstrated the relationship between PbL and Fisher LDA, SVM, and RVM classifiers in binary classification. Chang et al. [15] examined the PbL problem from a learning theory perspective.
Oyedotun and Khashman [16] categorized learning algorithms into two disjoint groups of prototype learning and adaptive learning. They suggested using a combination of these two learning methods in a learning system to exploit the benefits of both. In this article, we combine the benefits of PbL and HbL by introducing a prototype-based interpretation of the functionality of neurons.

We consider WTA structures based on the maximum inner product (IP) with a set of weight vectors (see 2) and the minimum Euclidean distance (ED) to a set of centers (see 4) and call them IP-WTA and ED-WTA, respectively. IP-WTA is an HbL method and ED-WTA is a PbL method. We prove that ED-WTA and IP-WTA models have the same representational power and each ED-WTA is equivalent to an IP-WTA and vice versa. We use this equivalence to find an ED-WTA network that is equivalent to a trained IP-WTA network. Besides, we propose a tailored version of the competitive cross-entropy (CCE) algorithm [17] for training ED-WTA networks from scratch. However, we found out that the centers of the obtained ED-WTA network deviate from the input data and cannot be considered as prototypes of data. The reason is that, during discriminative training of WTA networks, weights/centers not only are pulled toward the training samples but also are repelled from them. Therefore, each weight/center is the difference between two components, one representing a combination of samples that are added to it and the other representing a combination of samples that are subtracted from it.

We first propose a generalized HbL model, called ±IP-WTA, in which the weight vector of a neuron is represented as the difference of two weight vectors: one representing samples that are added to it and the other representing samples that are subtracted from it. We then propose a generalized PbL model, called ±ED-WTA, in which the weight vector of a neuron is modeled with two prototypes, a positive prototype representing the samples that belong to the neuron, and a negative prototype representing the samples that are erroneously assigned to the neuron during training. Interestingly, the computation of neurons of ±ED-WTA with positive and negative centers simplifies to IP. Modeling the weight vector of a neuron with two prototypes is of paramount importance as it gathers PbL and HbL together in artificial neural networks. Given any discriminative training algorithm for IP-WTA networks, we propose a training algorithm for ±IP-WTA and ±ED-WTA that properly shares out the training updates between the positive and negative components such that the networks are kept equivalent to the original IP-WTA during training.

The visualization of these positive and negative weights and centers revealed an interesting and somewhat unexpected fact. We observed that the negative prototype of each neuron, which had been created solely by input data from other classes, was indistinguishably similar to the positive prototype of that neuron. The rationale behind this observation is that the training data that are mistaken with a prototype are indeed similar to it. This interpretation of the functionality of IP neurons as computing the difference between the similarities to a positive and a negative prototype, which themselves are extremely similar, is one of the main findings of this article. This finding is in agreement with the BCM theory [18], which states that biological neurons discriminate between those input stimuli that excite the postsynaptic neuron strongly (here, those comprising the positive prototype) and those input stimuli that excite the postsynaptic neuron weakly (here, input patterns from other classes which are very similar to the positive prototype) [19]. Input stimuli that do not excite the postsynaptic neuron (here, those which are not similar to the positive prototype at all) do not affect the synaptic strength to that neuron. The contributions of this article are as follows.

1) We prove the equivalence of IP-WTA and ED-WTA networks after adding a nonnegative bias term to ED-WTA to compensate for the bias term in IP-WTA.
2) We explain why discriminative training of ED-WTA networks does not yield prototypes of data.
3) We propose the ±ED-WTA model along with a specialized training algorithm, which combines the benefits of HbL and PbL.
4) As reflected in the title of this article, the article’s main contribution is introducing a prototype-based interpretation of the functionality of neurons, which is in agreement with the BCM theory.

This article proceeds as follows. First, in Section II, we formally define IP-WTA and ED-WTA networks. Then, in Section III, we show that corresponding to each IP-WTA, there exists an ED-WTA with identical functionality and vice versa. However, considering that the centers should be close to the input data that they represent, in Section III-C, we propose an iterative algorithm that uses the weights of an IP-WTA network to obtain the centers of an equivalent ED-WTA whose centers are close to the input data. In Section IV, we propose a tailored version of the CCE loss [17] for training ED-WTA from scratch. In Section V, we introduce ±IP-WTA and ±ED-WTA networks in which two sets of positive and negative weights/centers are used in place of the original ones and propose specialized algorithms for their initialization and training. In Section VI, we report our experimental results on the MNIST, FERET, and CUB-200 datasets. Furthermore, we show the usefulness of the proposed ±ED-WTA model for rejecting outlier and adversarial inputs. We conclude this article in Section VII. The relations between different sections are summarized in Fig. 1.
II. INNER-PRODUCT AND EUCLIDEAN DISTANCE
WINNER-TAKE-ALL NETWORKS

Two commonly used criteria for measuring the similarity between an input sample and a prototype are Euclidean distance and inner product. Conceptually, the notion of similarity is best conveyed by the Euclidean distance (ED), while the inner product (IP) operation has benefits in terms of faster computation and biological plausibility. Most artificial neural networks use the additive neuron model that is inspired by the biological neurons in the brain. In this model, a single neuron computes a weighted sum of the activities of the presynaptic neurons plus a bias term that models the threshold potential in the biological neurons. We refer to this model as the IP model of the neuron since the output is computed as the inner product of the input vector with the weight vector.

In this article, we consider WTA networks with several neurons for each class, in which the network output is determined by the label of the neuron whose weights/centers are most similar (based on either ED or IP) to the input data. Consider a multiclass classification problem with $C$ classes and assume that the output layer of the network has $M \geq C$ neurons. For each $i \in \{1, \ldots, C\}$, let $O_i$ be the set of indices of output neurons that are assigned to class $i$. If we use IP to measure the similarity of an input vector $x$ to the weight vectors $w_1, \ldots, w_M$, then the label predicted by the network is calculated as

$$y_{\text{IP}} = \arg \max_{i \in \{1, \ldots, C\}} \max_{j \in O_i} w_j^T x + b_j.$$  

If we assume that the above-mentioned IP is applied to the augmented input $[x, 1]$ and the weights connecting the appended input to the output neurons are $b_1, \ldots, b_M$, then we obtain IP-based winner-take-all (IP-WTA), which computes the output as

$$y_{\text{IP}} = \arg \max_{i \in \{1, \ldots, C\}} \max_{j \in O_i} w_j^T x + b_j.$$  

Similarly, in the case of ED with centers $c_1, \ldots, c_M$, the predicted label is computed as

$$y_{\text{ED}} = \arg \min_{i \in \{1, \ldots, C\}} \max_{j \in O_i} \|x - c_j\|^2.$$  

If the input is augmented as $[x, 0]$ and the entries in the centers that correspond to the 0 input are $d_1, \ldots, d_M$, then we obtain an ED-based winner-take-all (ED-WTA) network, which computes the output as

$$y_{\text{ED}} = \arg \min_{i \in \{1, \ldots, C\}} \max_{j \in O_i} \|x - c_j\|^2 + d_j^2.$$  

III. EQUIVALENCE OF IP-WTA AND ED-WTA NETWORKS

In this section, we show that the modeling capabilities of IP-WTA and ED-WTA networks are the same. To this end, we show that there is an equivalent IP-WTA corresponding to each ED-WTA and vice versa.

A. From ED-WTA to IP-WTA

For WTA networks with a single neuron for each class, Martin-del-Brio [20] showed that there is an equivalent IP-WTA corresponding to each ED-WTA. The result can be trivially extended to the case of multiple neurons for each class. We will state this result for the sake of completeness. We show that for any ED-WTA network with centers $c_1, \ldots, c_M$ and biases $d_1, \ldots, d_M$, there exists an IP-WTA network with weights $w_1, \ldots, w_M$ and biases $b_1, \ldots, b_M$ such that, for any input data, the winning neurons of both models are the same. Starting from (4), we have

$$y_{\text{ED}} = \arg \min_{i \in \{1, \ldots, C\}} \max_{j \in O_i} \|c_j - x\|^2 + d_j^2$$

$$= \arg \min_{i \in \{1, \ldots, C\}} \max_{j \in O_i} (c_j - x)^T (c_j - x) + d_j^2$$

$$= \arg \min_{i \in \{1, \ldots, C\}} \max_{j \in O_i} (c_j^T c_j - 2c_j^T x + x^T x + d_j^2).$$  

Considering that for any input data $x$, the value $x^T x$ is fixed, we can eliminate it and obtain

$$y_{\text{ED}} = \arg \min_{i \in \{1, \ldots, C\}} \max_{j \in O_i} (c_j^T c_j - 2c_j^T x + d_j^2)$$

$$= \arg \max_{i \in \{1, \ldots, C\}} \left( c_j^T x - \frac{1}{2} (c_j^T c_j + d_j^2) \right).$$  

Hence, if we choose the parameters of the IP-WTA model as $w_j = c_j$ and $b_j = -(1/2)c_j^T c_j - (1/2)d_j^2$, then the two models become equivalent.

B. From IP-WTA to ED-WTA

Suppose that we have an IP-WTA network with weights $w_1, \ldots, w_M$ and biases $b_1, \ldots, b_M$. It is sufficient to show the existence of an ED-WTA network with some parameters $c_1, \ldots, c_M$ and $d_1, \ldots, d_M$ such that the winning neurons of both networks are the same for all inputs $x$, that is,

$$y_{\text{IP}} = \arg \max_{i \in \{1, \ldots, C\}} \max_{j \in O_i} w_j^T x + b_j$$

$$= \arg \max_{i \in \{1, \ldots, C\}} \max_{j \in O_i} \|c_j - x\|^2 + d_j^2.$$  

Using (6), it is enough to prove that

$$y_{\text{IP}} = \arg \max_{i \in \{1, \ldots, C\}} \max_{j \in O_i} w_j^T x + b_j = \arg \max_{i \in \{1, \ldots, C\}} \max_{j \in O_i} c_j^T x + e_j$$

where $e_j = -(1/2)(c_j^T c_j + d_j^2)$, for $j = 1, \ldots, M$. For (8) to be an identity, it suffices to exist a positive coefficient $\alpha > 0$ and another constant $\gamma$ such that for every $j \in \{1, \ldots, M\}$ and any input $x$

$$\alpha (w_j^T x + b_j) - \gamma = c_j^T x + e_j.$$  

For $j \in \{1, \ldots, M\}$, the values $w_j$ and $b_j$ are given from IP-WTA and the values of $c_j$, $d_j$, $\alpha$, and $\gamma$ must be chosen in such a way that the above equation holds. This choice is
not unique and there are different solutions for each choice of \(\alpha > 0\). The above equation holds if and only if for every \(j \in \{1, \ldots, M\}\), we have \(aw_j = c_j\) and \(ab_j - \gamma = e_j\), which leads to

\[
ab_j - \gamma = -\frac{1}{2}(c_j^T c_j + d_j^2) = -\frac{1}{2}(\alpha^2 w_j^T w_j + d_j^2).
\]

Therefore, for each choice of \(\alpha > 0\) and \(\gamma\), \(d_j\) is given by

\[
d_j = \pm \sqrt{-\alpha^2 w_j^T w_j - 2ab_j + 2\gamma}.
\]

In order to ensure that the expression under radical is not negative, it suffices to choose \(\gamma\) greater than \(\gamma_0 = (1/2)\max_j(\alpha^2 w_j^T w_j + 2ab_j)\). Therefore, any choice \(\alpha > 0\) and \(\gamma \geq \gamma_0\) yields an ED-WTA with \(c_j = aw_j\) and \(d_j\) as given by (11) that is equivalent to an IP-WTA with weights \(w_j\) and biases \(b_j\), where \(j = 1, \ldots, M\).

### C. Finding a Natural Equivalent ED-WTA

Assume that the weights \(w_1, \ldots, w_M\) are obtained by training an IP-WTA network. We want to find the centers \(c_1, \ldots, c_M\) in such a way that, first, the ED-WTA network has the same performance as the original IP-WTA network and, second, the centers \(c_1, \ldots, c_M\) are close to the input data that they represent. Let the centers \(c_j\) and the weights \(w_j\) be related by \(c_j = aw_j + u\), where \(u\) is fixed and does not depend on \(j\). For each input data \(x\), let \(q(x)\) be the index of the winning neuron in IP-WTA with the mathematical definition

\[
q(x) = \arg\max_{1 \leq j \leq M} w_j^T x + b_j.
\]

We consider the following error function, measuring the sum of the squared distances between the input data and the nearest center:

\[
E(\alpha, u) = \sum_{x \in \mathcal{D}} \|x - c_{q(x)}\|^2 = \sum_{x \in \mathcal{D}} \|x - (\alpha w_{q(x)} + u)\|^2 = \sum_{x \in \mathcal{D}} \left[\alpha^2 \|w_{q(x)}\|^2 + \|u\|^2 - 2\alpha w_{q(x)}^T x - 2u^T w_{q(x)} + \text{const}\right]
\]

where \(\mathcal{D}\) is the set of training samples. To minimize \(E(\alpha, u)\), we set the partial derivatives with respect to the parameters \(\alpha\) and \(u\) equal to zero

\[
\frac{\partial E}{\partial \alpha} = \sum_{x \in T} 2\alpha \|w_{q(x)}\|^2 - 2w_{q(x)}^T x + 2u^T w_{q(x)} = 0
\]

\[
\frac{\partial E}{\partial u} = \sum_{x \in T} 2u - 2x + 2aw_{q(x)} = 0
\]

and obtain the following fixed-point equations:

\[
\alpha = \frac{\sum_{x \in \mathcal{D}} (x - u)^T w_{q(x)}}{\sum_{x \in \mathcal{D}} \|w_{q(x)}\|^2}
\]

\[
u = \frac{1}{|\mathcal{D}|} \left(\sum_{x \in \mathcal{D}} x - \alpha w_{q(x)}\right).
\]

We start with \(\alpha = 0\) and alternate between updating \(u\) and \(\alpha\) using (14). We prove, in the Appendix, that this process converges to a global optimum.

Note that for all choices of \(\alpha\) and \(u\), the ED-WTA network with centers \(c_j = \alpha w_j + u\) and ED biases \(d_j = \pm(c_j^T c_j - 2ab_j + 2\gamma)^{1/2}\) is equivalent to the original IP-WTA network. Since our goal is to obtain prototypes that are similar to input data, the natural choice for the ED biases \(d_1, \ldots, d_M\) is zero, as they correspond to a feature with value zero in the augmented input vectors \([x, 0]\). Thus, at the end of the above-mentioned fixed-point optimization, we remove the parameters \(d_1, \ldots, d_M\) from ED-WTA, which breaks the equivalence between the IP-WTA and ED-WTA networks and considerably deteriorates the accuracy of the ED-WTA network. To take back the lost accuracy, in the Section IV, we introduce a variant of the CCE loss for optimizing ED-WTA networks and call the optimized network NAT-IP-EQUIV-ED-WTA.

### IV. Training ED-WTA with CCE Loss

The cross-entropy (CE) loss is ubiquitously used for training neural networks in multiclass classification problems. CCE is a generalization for the case of more than one neuron per class [17]. Here, we tailor CCE loss for training ED-WTA networks. In contrast to IP-WTA in which the winning neuron is the neuron with maximum activity, in ED-WTA networks, the winning neuron is the least active one, the one with the lowest distance to the learned prototype. To be able to use the machinery (like softmax) of maximum-activity WTA networks, we modify the functionality of ED neurons slightly with formula

\[
z_j = -\frac{1}{2}\|x - c_j\|^2 \quad \text{for} \quad j = 1, \ldots, M
\]

which leads to a maximum-activity formulation for ED-WTA. Now, consider an ED-WTA network with \(M\) output neurons and, for \(i = 1, \ldots, C\), let \(O_i\) denote the set of indices of output neurons that are assigned to class \(i\). For \(j = 1, \ldots, M\), let \(z_j\) be the value of the output ED neuron \(j\). To convert the output of the network to a probability distribution, we apply a softmax function with some parameter \(\beta\) to \(z\) and obtain the probability distribution \(y\) with formula

\[
y_j = \frac{e^{\beta z_j}}{\sum_{i=1}^{M} e^{\beta z_i}}, \quad \text{for} \quad j = 1, \ldots, M.
\]

As suggested in [17], the target distribution \(\tau\) for each sample input is constructed by competition among the neurons that belong to the true class (say \(k\))

\[
\tau_j = \begin{cases} 
\frac{e^{\beta z_j}}{\sum_{i\in O_k} e^{\beta z_i}}, & j \in O_k \\
0, & \text{otherwise}.
\end{cases}
\]

The instantaneous CCE loss for a single training sample is defined as the CE between the target distribution and the output distribution

\[
E_{\text{CCE}} = -\sum_{j=1}^{M} \tau_j \log y_j.
\]

\(^1\)For a generalization of IP and ED neurons, see L2 family of generalized convolution operators in [21], which is a family of similarity measures that smoothly transform from IP to ED.
The inclusion of the parameter $\beta$ in the softmax operation for ED-WTA is crucial. The reason is that unless the input values $z$ to the softmax layer are in an appropriate range, the gradient backpropagated by the softmax layer would vanish. In IP-WTA, the weights themselves can be adjusted to bring $z$ to an appropriate range. However, in ED-WTA, the centers should be kept close to the input data, and therefore, an extra parameter $\beta$ is needed to put the values of $z$ in an appropriate range for the softmax operation. The parameter $\beta$ is trained, like all the parameters of the network, with gradient descent. The gradient of the error function $E^{CCE}$ with respect to the centers $c_1, \ldots, c_M$ and the parameter $\beta$ is

$$
\frac{\partial E^{CCE}}{\partial c_j} = \frac{\partial E^{CCE}}{\partial z_j} \frac{\partial z_j}{\partial c_j} = \beta(y_j - \tau_j)(x - c_j)
$$

$$
\frac{\partial E^{CCE}}{\partial \beta} = -\frac{1}{2} \sum_{j=1}^{M} (y_j - \tau_j)\|x - c_j\|^2.
$$

Using a learning rate of $\mu$, the formulas for updating the centers $c_1, \ldots, c_M$ and the parameter $\beta$ with stochastic gradient descent are

$$
c_j^{\text{new}} = c_j^{\text{old}} - \mu \beta(y_j - \tau_j)(x - c_j)
$$

$$
\beta^{\text{new}} = \beta^{\text{old}} + \frac{1}{2} \mu \sum_{j=1}^{M} (y_j - \tau_j)\|x - c_j\|^2.
$$

Remember that we assumed that the true label is $k$. For output neurons of the true class, i.e., when $j \in \mathcal{O}_k$, we have

$$
y_j = \frac{e^{\beta z_j}}{\sum_{i \in \mathcal{O}_k} e^{\beta z_i} + \sum_{i \notin \mathcal{O}_k} e^{\beta z_i}} \leq \frac{e^{\beta z_j}}{\sum_{i \in \mathcal{O}_k} e^{\beta z_i}} = \tau_j.
$$

On the other hand, for neurons of other classes, i.e., when $j \notin \mathcal{O}_k$, we have $\tau_j = 0$ and therefore

$$
y_j = \frac{e^{\beta z_j}}{\sum_{i=1}^{M} e^{\beta z_i}} > \tau_j.
$$

From (22) and (23), it follows that in (20), the centers $c_j$ belonging to the true class move toward the input data, while the centers $c_j$, which belong to other classes, are repelled from it. Stated another way, for each input sample, some positive multiples of it are added to the centers belonging to the true class, and some positive multiples are subtracted from the centers of other classes. Consequently, instead of being prototypes for input data, the centers of ED-WTA are differences of two prototypes of input data: one protoyping the data that belong to that center and another protoyping the data that are mistakenly won by that center.

V. EXPLICIT MODELING OF POSITIVE AND NEGATIVE PROTOTYPES

As we saw in the previous section, the centers learned by CCE for ED-WTA were not prototypes for the data they represented. The reason is that, during the discriminative training process, while data belonging to a center appear with a positive sign in the update relation, however, data belonging to other classes will also contribute to the value of the center with a negative sign. While we desire that the centers become approximately equal to the mean of the data that they represent, nonetheless, the discriminative training of the parameters causes this not to be the case and the centers are a combination of the data with positive and negative coefficients.

Based on this observation, first of all, we split the parameters of each neuron into a positive parameter and a negative one. Then, we propose a new learning rule and apply all updates in which the input data appear with a positive sign to the positive parameter and those with a negative sign to the negative parameter. We apply this method to IP-WTA and ED-WTA models and call them $\pm$IP-WTA and $\pm$ED-WTA models, respectively.

A. $\pm$IP-WTA Model

Suppose that for each $j \in \{1, \ldots, M\}$, we split the weight vector $w_j$ in IP-WTA to $w_j = w_j^+ - w_j^-$, where $w_j^+$ and $w_j^-$, respectively, represent the contributions of data with positive and negative coefficients to the weight vector $w_j$. We call this new model $\pm$IP-WTA and call $w_j^+$ and $w_j^-$ the positive and the negative weights, respectively. For training the parameters, in the new formulation, always a positive coefficient of each input sample is accumulated with either the positive or the negative weights. Thus, whenever the update rule of IP-WTA adds a positive coefficient of a training sample to $w_j$, we apply it to $w_j^+$, and whenever the update rule of IP-WTA adds a negative coefficient of a training sample to $w_j$, we apply this change with a positive coefficient to $w_j^-$. Clearly, at each step, only one of the parameters $w_j^+$ or $w_j^-$ would be updated. The updating rule of weights in the CCE algorithm is

$$
w_j^{\text{new}} = w_j^{\text{old}} - \mu (y_j - \tau_j)x \quad \forall j \in \mathcal{O}_k
$$

$$
w_j^{\text{new}} = w_j^{\text{old}} - \mu (y_j - 0)x \quad \forall j \notin \mathcal{O}_k.
$$

We have already shown in (22) and (23) that the coefficient of $x$ is positive for $j \in \mathcal{O}_k$ and negative for $j \notin \mathcal{O}_k$. Therefore, updating weights in $\pm$IP-WTA is defined by

$$
w_j^{+\text{new}} = w_j^{+\text{old}} + \mu (\tau_j - y_j)x \quad \forall j \notin \mathcal{O}_k
$$

$$
w_j^{-\text{new}} = w_j^{-\text{old}} + \mu y_jx \quad \forall j \notin \mathcal{O}_k.
$$

The above formulation guarantees that both the positive and the negative weights will be combinations of data with positive coefficients. Although, in $\pm$IP-WTA, we obtain weights that are visually similar to the input data, the positive/negative weights are not prototypes of data since their magnitude is arbitrary. To solve this drawback, in the next section, we apply this method to the ED-WTA model and obtain positive and negative prototypes.

B. $\pm$ED-WTA Model

Naive splitting of the centers of ED-WTA results in the following new model:

$$
z_j = -\frac{1}{2}\|x - (c_j^+ - c_j^-)\|^2
$$

2Please note that in general, when data take negative values, $w_j^+$ and $w_j^-$ may also take negative values. Thus, do not confuse the contribution of data with a positive coefficient with the positivity of $w_j^+$. 

Authorized licensed use limited to the terms of the applicable license agreement with IEEE. Restrictions apply.
where both $c_{j}^{+}$ and $c_{j}^{-}$ are obtained by accumulating training samples with nonnegative coefficients. The problem with this model is that it favors a low distance between the input data and the difference of the positive center $c_{j}^{+}$ and the negative center $c_{j}^{-}$, not a low distance to the centers themselves. Therefore, we propose the following model for neurons and call it ±ED-WTA:

$$z_{j} = -\frac{1}{2} \left( \|x - c_{j}^{+}\|^2 - \|x - c_{j}^{-}\|^2 \right).$$

(27)

Considering that the new model is based on the distances between the centers $c_{j}^{+}$ and $c_{j}^{-}$ with the input data $x$, we expect that, by a proper training method, $c_{j}^{+}$ and $c_{j}^{-}$ would become prototypes of data. Interestingly, the ±ED-WTA model (27) simplifies to the IP-WTA model

$$z_{j} = \left( c_{j}^{+} - c_{j}^{-} \right)^{T} x - \frac{\|c_{j}^{+}\|^2 - \|c_{j}^{-}\|^2}{2}.$$  

(28)

Please note that while the computation of ±ED-WTA simplifies to IP-WTA, from the learning viewpoint, ±ED-WTA differs considerably from IP-WTA. Since the bias term in (28) is written in terms of positive and negative centers, the gradient of the centers of ±ED-WTA differs from the gradient of the weights in IP-WTA. Besides, since $c_{j}^{+}$ and $c_{j}^{-}$ should be prototypes of the data, the magnitude of $c_{j}^{+} - c_{j}^{-}$ cannot be freely chosen and there is nothing to prevent the values $z_{1}, \ldots, z_{M}$ in (28) from falling into the saturated region of the softmax function in which the backpropagated error becomes approximately zero. Therefore, for ±ED-WTA, there should be another parameter, say $\beta$, which adjusts the input’s range of the softmax function.

The structure of a ±ED-WTA network with explicit positive and negative centers is shown in Fig. 2. Note that according to (27) and (28), after training, ±ED-WTA can be substituted with an ordinary IP-WTA network. Training the parameters of ±ED-WTA can be based on any optimization algorithms such as stochastic gradient descent, NAG, or Adam. However, to achieve proper positive and negative prototypes, we should update the parameters more cleverly. If we apply an optimization algorithm to the new formulation (28) based on the positive centers $c_{j}^{+}$ and the negative centers $c_{j}^{-}$, then both the positive and the negative centers of all neurons are changed for each input data, violating the philosophy behind them. Therefore, at each step, the positive centers $c_{j}^{+}$ belonging to the class of the input sample and the negative centers $c_{j}^{-}$ of other classes are updated. Considering the CCE algorithm and assuming that the label of the current training data is $k$, we have

$$\tau_{j} = \begin{cases} e^{\beta z_{j}}, & j \in O_{k} \\ 0, & j \notin O_{k} \end{cases}$$

(29)

The rules for updating the centers of ±ED-WTA are

$$c_{j}^{+}_{\text{new}} = c_{j}^{+}_{\text{old}} + \mu \beta (\tau_{j} - y_{j}) (x - c_{j}^{+}_{\text{old}}) \quad \forall j \in O_{k}$$

$$c_{j}^{-}_{\text{new}} = c_{j}^{-}_{\text{old}} + \mu \beta y_{j} (x - c_{j}^{-}_{\text{old}}) \quad \forall j \notin O_{k}$$

(30)

where $y$ is the probability distribution obtained by the softmax function according to (16). These rules always update the positive/negative centers toward the input data, making these centers prototypes of the input data. In contrast to ±IP-WTA, in which the weights are updated with multiple input data, in ±ED-WTA, the centers are updated based on their distance to the input data. Consequently, in contrast to ±IP-WTA, in ±ED-WTA, the centers will remain close to the data that they represent. Therefore, we obtain positive and negative prototypes that are close to the input data in the Euclidean norm. Our training algorithm for ±ED-WTA is shown in Algorithm 1.

**Algorithm 1 ±ED-WTA Training Algorithm**

**Require**: Dataset $D = \{(x_{1}, l_{1}), (x_{2}, l_{2}), \ldots, (x_{|D|}, l_{|D|})\}$

**Require**: Learning rate $\mu$

1: Initialize centers $c_{j}^{+}$ and $c_{j}^{-}$ for all $j = 1, \ldots, M$

2: Initialize parameter $\beta$

3: for iter = 1 : number of epochs do

4: for $i = 1 : |D|$ do

5: Let $k = l_{i}$ denote the label of the current sample

6: for $j = 1, \ldots, M$ do

7: $z_{j} = (c_{j}^{+} - c_{j}^{-})^{T} x_{i} - \frac{\|c_{j}^{+}\|^2 - \|c_{j}^{-}\|^2}{2}$

8: $y_{j} = \frac{\exp(\beta z_{j})}{\sum_{k=1}^{M} \exp(\beta z_{k})}$

9: end for

10: $\tau_{j} = \frac{\exp(\beta z_{j})}{\sum_{k=1}^{M} \exp(\beta z_{k})}, \quad \forall j \in O_{k}$

11: $c_{j}^{+} \leftarrow c_{j}^{+} + \mu \beta (\tau_{j} - y_{j}) (x_{i} - c_{j}^{+}) \quad \forall j \in O_{k}$

12: $c_{j}^{-} \leftarrow c_{j}^{-} + \mu \beta y_{j} (x_{i} - c_{j}^{-}) \quad \forall j \notin O_{k}$

13: $\beta \leftarrow \beta - \mu \sum_{j=1}^{M} (y_{j} - \tau_{j}) z_{j}$

14: end for

15: end for

**C. Initialization of Centers in ±ED-WTA**

Positive and negative centers in a ±ED-WTA can be initialized by random values (RAND-INIT). In our initial experiments, we would use this method to ensure that the emerged positive and negative centers, especially the similarity...
between the positive and the negative centers, are not due to a special initialization method. However, the learning process would speed up if the centers were initialized to values close to their final values. One choice is the mean of all training data or a small perturbation of it to break symmetry (MEAN-INIT). For positive centers, a better choice is to initialize them by the centers obtained by a clustering algorithm such as k-means (K-MEANS-INIT). We also devised another initialization method which we call CLEVER-INIT, in which the positive and negative centers are initialized with K-MEANS-INIT and MEAN-INIT, respectively.

VI. EXPERIMENTS

In this section, we experimentally evaluate different WTA models and show the usefulness of ±ED-WTA on a set of tasks. First, in Section VI-A, we compare the recognition rate and the interpretability of LVQ. IP-WTA, ED-WTA, ±IP-WTA, and ±ED-WTA on the MNIST digit recognition dataset and show that ±ED-WTA has the best combination of high recognition rate and prototype-based interpretability. Considering ±ED-WTA as our method of choice, we then focus on showing the usefulness of ±ED-WTA on various tasks. In Section VI-B, we perform a gender recognition experiment on FERET and show that ±ED-WTA yields excellent prototypes of typical faces of men and women. In Section VI-C, we apply ±ED-WTA to DenseNet-121 and show its usefulness in obtaining a highly interpretable viewpoint about the functionality of a DNN on the challenging dataset of Caltech-UCSD Birds-200-2011. In Section VI-D, we show the usefulness of the proposed ±ED-WTA model in detecting outlier samples. Finally, in Section VI-E, we show the robustness of ±ED-WTA against adversarial examples. The code used for performing these experiments is available at https://github.com/raminzs/pmED-WTA.

A. Comparing WTA Models on MNIST

In this section, we compare various WTA models in terms of recognition accuracy and prototype-based interpretability. We perform this comparison on the MNIST dataset, which is a standard dataset to evaluate neural networks on multiclass classification problems. This dataset contains 60,000 training images and 10,000 testing images from English handwriting digits with a size of 28 × 28. Table I summarizes the recognition rates of different WTA methods on MNIST. The results obtained by IP-WTA, ED-WTA, ±IP-WTA, and ±ED-WTA are almost identical and statistically insignificant. However, LVQ algorithms obtain significantly lower recognition rates.

1) Experimental Settings: In all experiments, we set the number of output neurons for each class to 6 and the number of training epochs to 50. All experiments were repeated five times with five different random seeds. We selected the initial learning rate by fivefold cross validation from the set of values \{10^{-2}, \ldots, 10^{2}\}. During training, we multiplied the learning rate by 0.95 after completing each epoch. For training LVQ, the number of epochs and neurons is the same as other algorithms, and the LVQ-PAK package automatically selected other parameters. In all color images, positive values are shown in green and negative values are shown in red.

2) IP-WTA and ED-WTA: Fig. 3(a) shows the weights of an IP-WTA network trained on MNIST. As desired, the weights did not emerge as prototypes of the data. The weights’ values of an IP-WTA network can be best described as a vote given by each pixel in favor/against the activity of each neuron. The centers obtained by the algorithm of Section III-C applied to the MNIST dataset are shown in Fig. 3(b). As can be seen, in comparison to the weights of IP-WTA, the centers of ED-WTA have become slightly more similar to the data of their associated classes. However, the obtained centers are still far from the prototypes of the data as can be understood by looking at the centers obtained by the k-means algorithm, as shown in Fig. 3(d). Besides, the accuracy decreased from 96.42% in IP-WTA to 88.14% when the bias terms of ED neurons had been dropped. By dropping the biases in ED-WTA and continuing training with CCE, the accuracy went up to 96.42%, but the resulting weights, as shown in Fig. 3(c), deviated even more from being prototypes.

3) LVQ Algorithms: Fig. 4 shows the weights obtained by different versions of the LVQ algorithm. Although the centers in LVQ1 can be regarded as prototypes of data, LVQ1 has a considerably lower recognition rate of 91.11%. On the other hand, the centers of LVQ2.1 and LVQ3 are not prototypical of data, while these algorithms obtain higher recognition rates of 93.91% and 94.03%, respectively. Fig. 4(d) shows the average of samples belonging to each neuron after training with LVQ2.1. It can be seen that the prototypes of data, as shown in Fig. 4(d), significantly differ from the centers obtained by LVQ2.1, as shown in Fig. 4(b). The centers of Fig. 4(b) are created by a combination of training samples with positive and negative coefficients and even some pixels of the trained centers took negative values. Thus, the centers obtained by LVQ2.1 do not properly represent the corresponding classes.

4) ±ED-WTA and ±IP-WTA: Now, we report the results of the main proposal of this article: ±ED-WTA. Fig. 5(a) and (b) shows the positive and the negative prototypes obtained by a ±ED-WTA network trained on MNIST. As desired, the positive prototypes for each class are prototypical for the corresponding data. Nevertheless, the astonishing result is that the negative centers have also developed to be representative of data of that class. This is particularly strange as these prototypes are merely obtained by samples from other classes.

| Model | Initialization method | Accuracy    |
|-------|-----------------------|-------------|
| LVQ1  | LVQ-PAK initialization program | 91.11 ± 0.21% |
| LVQ2.1| LVQ-PAK initialization program | 93.91 ± 0.10% |
| LVQ3  | LVQ-PAK initialization program | 94.03 ± 0.21% |
| IP-WTA| RAND-INIT              | 96.42 ± 0.21% |
| ED-WTA| RAND-INIT              | 96.04 ± 0.15% |
|       | K-MEANS-INIT           | 96.41 ± 0.13% |
|       | NAT-IP-EQUIV-ED-WTA    | 96.42 ± 0.21% |
| ±IP-WTA| RAND-INIT             | 96.33 ± 0.16% |
| ±ED-WTA| RAND-INIT             | 96.42 ± 0.08% |
|       | CLEVER-INIT            | 96.58 ± 0.11% |
However, this result becomes understandable by noticing that a negative prototype is formed by samples that are so similar to the positive prototype that they are mistaken with it and wrongly classified. Thus, it can be said that the negative prototypes are perturbed versions of their corresponding positive prototypes. Fig. 5(c) shows the difference between the positive and negative prototypes, a visualization that is very similar to the weights of IP-WTA. Furthermore, ±ED-WTA obtained a recognition rate of 96.58%. As desired, ±ED-WTA combines the high accuracy of IP-based networks with the interpretability of ED-based models. Pictures of the positive and negative weights for ±IP-WTA (not shown) are similar to the positive and negative centers of ±ED-WTA with the difference that their actual magnitude is not within the range of data.

B. Experiment on FERET

In this section, we provide a prototype-based interpretation for the functionality of neurons in an IP-WTA network obtained after training a ±ED-WTA network on FERET for the task of gender classification. The color FERET dataset [22], [23] contains 11338 facial images of size 512 × 768 and is provided in two DVDs. From the first DVD, we extracted the frontal facial images in which the positions of the left and right eyes were marked and the subject did not wear glasses, resulting in 822 male faces and 654 female faces. We aligned faces using the imutils python package such that the left and right eyes have fixed positions in the image. We then cropped and resized the images to make them 182 × 182 pixels.

We first clustered each of the sets of faces of men and women into ten clusters using k-means. We initialized the positive centers of a ±ED-WTA network with the centers of these clusters. Also, we initialized the negative centers with an image whose single intensity was the average of all pixels of all facial images. We trained the network for ten iterations. Fig. 6 shows the positive and negative centers at the end of training. The figure shows that the network has got expertise in differentiating very similar men’s and women’s faces. Although the final ±ED-WTA network can be expressed as an ordinary IP-WTA network, it yields a very clear interpretation of the functionality of each neuron.

C. Experiment on Caltech-UCSD Birds-200-2011

In this section, we show how the use of ±ED-WTA at the last layer of a deep classification network provides an interpretation of the functionality of a DNN on a challenging dataset. We use the Caltech-UCSD Birds-200-2011 (CUB-200) dataset that consists of 5994 training images and 5794 testing images of 200 different bird species. We obtained cropped images of the birds using the location annotation given in the dataset.

We devise a method for obtaining an interpretable visualization of the positive prototypes learned by a ±ED-WTA layer in a DNN. Motivated by [24], we desire to associate each feature at the representation layer of the network with a region in the input image. We use the DenseNet-121 architecture [25] since it first obtains a representation consisting of 1024 planes of size 7 × 7 and then simplifies it to 1024 features using global average pooling. The use of global pooling suggests that all neurons of a 7 × 7 plane detect the same pattern but at different regions of the input image. We replace the global average pooling with a global max pooling so that each one of the 1024 features can be traced back to a specific neuron in the preceding 7 × 7 plane. We also split the 224 × 224 pixels of the input image area into 7 × 7 nonoverlapping 32 × 32 regions and associate each of the 7 × 7 output neurons on the DenseNet-121 output plane with one of these regions.

As a baseline, we trained an ordinary classifier (using softmax with the CE loss) on top of the features extracted from a pretrained (on ImageNet) DenseNet-121 network. After training the softmax layer for 30 epochs and fine-tuning the whole network for ten epochs, we obtained a recognition rate of 72.59% on the test set. After using global max pooling and training the softmax layer for ten epochs and fine-tuning the whole network for ten epochs, the recognition rate reached 72.92%. We used the features extracted by this fine-tuned DenseNet-121 network as input data for training a ±ED-WTA network.

We decided to use two neurons for each class to model two variations for each bird (e.g., one variation in which the bird is sitting and another in which the bird is flying). We initialized the ±ED-WTA network with CLEVER-INIT and trained it for 20 epochs. The trained ±ED-WTA network obtained a recognition rate of 74.83% and yielded 400 positive prototypes and 400 negative prototypes for the two variants of the 200 classes of the birds. Given that the DenseNet-121 features have passed through a ReLU activation function, they are nonnegative, and we can consider the magnitude of each feature as a measure of its importance in the prototype.

However, it is necessary to normalize the value of each feature with respect to the distribution of that feature in the training data. For a prototype c, we define the importance score of the ith feature, with mean μ_i and standard deviation σ_i over the training dataset, as max((c_i − μ_i)/σ_i, 0). Consequently, for each positive prototype learned in ±ED-WTA, we obtain an importance score for each of the 1024 features.

Since we use a global max pooling, we know the location of each of the 1024 features within a 7 × 7 grid in the input image. However, this is not the case for the prototypes since their feature values are obtained by averaging over different training samples, each sample corresponding to a different location in the input image. For this reason, instead of visualizing a positive prototype itself, we visualize data won by the neuron associated with that prototype. For each input image won by a neuron, we first associate each feature to a location within the 7 × 7 grid and then add up the importance scores of the features of the positive prototype for each region.

To avoid producing a jagged visualization, we represent each of the 32 × 32 input regions with a Gaussian filter centered at that region with a standard deviation of 16. Using the sum of the importance scores for each region as weight, we compute the weighted sum of these Gaussian filters, obtaining a filter on the input image, which shows the importance of each region. By multiplying the input image with this filter, we get an image in which the areas
corresponding to the positive prototype are shown brighter, and the other areas are shown darker. By looking at a set of images produced in this way for a positive prototype, one can infer the projection of the positive prototype in the input domain. Due to the depth of the network and the fact that a feature may be observed in different input areas, the location...
Fig. 6. Positive and negative centers obtained after training a ±ED-WTA network on FERET for gender recognition (a) Positive (above) and negative (below) centers obtained for men. (b) Positive (above) and negative (below) centers obtained for women.

corresponding to each feature is different for each input data.

Fig. 7 visualizes the positive prototypes for five classes of birds. Each positive prototype is represented by two sample input images, highlighted based on important features in the positive prototype. We chose these two samples in a way that best reflects the concept learned by the positive prototype. We observe that the obtained visualization clearly describes the pattern learned by each neuron. For instance, one of the neurons allotted to the black-footed Albatross class models a flying bird and the other models a bird with closed wings. In addition, highlighted regions show that the neural network truly attends to the relevant areas of the input image for the task of recognition.

D. Robustness Against Outlier Inputs

Although neural networks have achieved great success in pattern classification, they usually suffer from vulnerability against out-of-class samples. Thus, when they are fed a sample from an unseen class, they still associate it to a known class, possibly with high confidence. In this section, we show the superiority of the proposed ±ED-WTA model against IP-WTA in detecting outlier data, which belongs to none of the classes. For this purpose, we train ±ED-WTA and IP-WTA on the MNIST digits and evaluate their robustness by testing on face samples from the ORL dataset [26]. ORL face dataset contains 400 images captured from 40 distinct subjects. The images are grayscale of size $92 \times 112$, which we resized to $28 \times 28$ for compatibility with the size of MNIST images. We consider two distinct test sets: 1) the test set of MNIST that consists of 10,000 digits and 2) all 400 faces from the ORL dataset.

We evaluate the classifiers by two criteria: the true acceptance rate and the true rejection rate. The true acceptance rate is the ratio of the MNIST test samples that got accepted and the true rejection rate is the ratio of ORL samples that got rejected. Suppose that $k \in \{1, \ldots, C\}$ is the predicted label (i.e., winning class) and that $m \in O_k$ is the index of the winning neuron in a ±ED-WTA network. Since, by (28), ±ED-WTA simplifies to IP-WTA, the probability of the winning class $k$ according to the IP neuron model is

$$P_{IP}(x) = \frac{\sum_{j \in O_k} e^{\beta z_j}}{\sum_{i=1}^M e^{\beta c_i}} (31)$$

where $z_1, \ldots, z_M$ are calculated by (28).

To distinguish between outlier samples and input data that indeed belong to one of the classes, we consider the positive center of the winning neuron in the ±ED-WTA model and propose an additional confidence measure

$$P_{+ED}(x) = \frac{\sum_{j \in O_k} e^{-\frac{1}{2} \|x-c_j\|^2}}{\sum_{i=1}^M e^{-\frac{1}{2} \|x-c_i\|^2}} (32)$$

which is the probability that, based on the distance to positive centers, sample $x$ belongs to the winning class $k$. Fig. 8 shows several samples of the ORL dataset along with their confidence measures. While these samples have a high probability of $P_{IP}$ (in most cases, above 99%), the ±ED-WTA model assigns a low probability of $P_{+ED}$ to them and easily detects them as outliers.

Fig. 9 shows the acceptance rate on the MNIST test set and the rejection rate on the ORL dataset of the

![Fig. 6. Positive and negative centers obtained after training a ±ED-WTA network on FERET for gender recognition (a) Positive (above) and negative (below) centers obtained for men. (b) Positive (above) and negative (below) centers obtained for women.](image)

![Fig. 7. Visualization of the positive prototypes learned by a ±ED-WTA network over the representation provided by a denseNet-121 network on the CUB-200 dataset. Each row of the figure corresponds to a class of birds. The rows from top to bottom correspond to the following classes of birds. 1: Black-footed Albatross. 2: Laysan Albatross. 3: Sooty Albatross. 4. Pelagic Cormorant. 5: Frigatebird. Variations within each class are modeled by two neurons. Each pair of images in a row represents the pattern learned by a neuron. Full results for the first 25 classes can be obtained from http://profsite.um.ac.ir/~k.ghiasi/publications/TNNLS2022.](image)

![Fig. 8. Shows several samples of the ORL dataset along with their confidence measures.](image)

![Fig. 9. Shows the acceptance rate on the MNIST test set and the rejection rate on the ORL dataset of the ±ED-WTA model.](image)
Fig. 8. Confidence measures of some outlier samples from the ORL dataset, computed by a ±ED-WTA model trained on the MNIST dataset. The confidence measures $P_{IP}$ and $P^{+ED}$ are drawn below each sample from left to right.

Fig. 9. Acceptance rate on the MNIST test set and the rejection rate on the ORL dataset for different threshold values of $P^{+ED}$.

±ED-WTA model for different threshold values on $P^{+ED}$. For instance, by choosing the threshold value 0.19 for $P^{+ED}$, the acceptance and rejection rates will be 96.7% and 98.75%, respectively. However, the equivalent IP-WTA, in which only the probability $P_{IP}$ can be used for rejection, achieves an acceptance rate of 77.95% and a rejection rate of 64.5% when we empirically set the threshold value for $P_{IP}$ to maximize the multiplication of the acceptance and rejection rates.

E. Robustness Against Adversarial Data

Recent studies have shown that neural networks are vulnerable to adversarial examples [27]. Adversarial examples are generated by applying some small perturbations to real samples and can easily fool a neural network to predict wrong classes with high confidence. In this section, we show the robustness of ±ED-WTA against adversarial examples. For this purpose, we generate two types of adversarial data.

1) Type-1: Data that do not belong to any classes, while the model recognizes and assigns them to a class with a high probability.

2) Type-2: Data that are related to a certain class but are classified incorrectly by the model with a high probability.

To this end, similar to the previous experiment, we first train a ±ED-WTA model on the MNIST training data. By starting from 1000 pure noise images, we generate the Type-1 adversarial data in which each sample corresponds to one of the ten classes. For each sample, the values of pixels are updated by backpropagating errors from the equivalent IP-WTA network of the trained ±ED-WTA model.

Fig. 10. Process of generating adversarial examples. (a) Some samples from the MNIST test set along with a pure-noise image. (b) Positive centers of the neuron in ±ED-WTA chosen as the target for generating an adversarial example. (c) Resulting adversarial examples. In (a) and (c), the probabilities $P_{IP}$ and $P^{+ED}$ are drawn from left to right below each sample. While $P_{IP}$ is high for both of the original and adversarial examples, $P^{+ED}$ is only high for the original digit images. The values of $P_{IP}$ and $P^{+ED}$ for the pure-noise image in (a) are very interesting. Since the pure-noise image is far from all hyperplanes, $P_{IP}$ confidentially accepts it as a digit. On the other hand, since the pure-noise image is far from all positive centers of ±ED-WTA, $P^{+ED}$ confidentially rejects it.

Fig. 11. Acceptance rate on the MNIST test set and the rejection rate on 100k adversarial data for different threshold values of $P^{+ED}$.

To generate Type-2 adversarial data, we use 10000 MNIST test images and update their pixel values using backpropagation while setting the target to one of the other nine classes. Thus, 10000 Type-1 data and 90000 Type-2 data are generated. Overall, we have 110000 testing data composed of 100000 adversarial data and the original 10000 MNIST test data.

Fig. 10 shows the process of generating one Type-1 and ten Type-2 adversarial examples using the IP-WTA network obtained after training a ±ED-WTA network on the MNIST dataset. While the generated adversarial examples have high $P_{IP}$ probabilities, these samples can be easily rejected based on their low $P^{+ED}$ probabilities. Similar to the previous section, we obtained a detector for accepting the MNIST testing data and rejecting the adversarial examples by choosing a threshold on output confidences.

Fig. 11 shows the acceptance rate on the MNIST test set and the rejection rate on the adversarial data for the ±ED-WTA model when choosing different thresholds for $P^{+ED}$.
The $\pm$ED-WTA model achieved a 91.34% acceptance rate and a 90.39% rejection rate by choosing the threshold value 0.3 for $P_{\pm}$ED. However, we could not reject even 1% of adversarial samples by using the probability $P_{\pm}$IP, irrespective of the threshold value.

**VII. CONCLUSION**

Winner-takes-all (WTA) networks based on Euclidean distance (ED-WTA) have been used for prototype-based multiclass classification in LVQ. Nowadays, WTA networks based on the inner product (IP-WTA) are ubiquitously used at the last layer of neural networks for multiclass classification and are trained using softmax with CE loss.

In this article, we reconsidered PbL in WTA networks. We first showed that IP-WTA and ED-WTA have the same representational power. Then, we proposed a novel algorithm for discriminative training of ED-WTA networks using a CCE loss. We noticed that during discriminative training, the centers deviate considerably from prototypes of data and tend to a difference between two prototypes: a positive prototype representing data from the true class and a negative prototype for data from other classes. We proposed the $\pm$ED-WTA network in which each neuron is modeled by two prototypes based on the Euclidean distance: a positive prototype for modeling the data that are truly won by this neuron and a negative prototype for those that are erroneously won. We also proposed $\pm$IP-WTA in which each weight is decomposed as the difference between a positive and a negative weight.

We devised a novel algorithm for training $\pm$IP-WTA and $\pm$ED-WTA. While training of $\pm$ED-WTA networks differs from ordinary neural networks, the resulting model of a trained $\pm$ED-WTA can be expressed as an ordinary IP-WTA network. This interpretation of the functionality of neurons as differentiating between those stimuli that are strongly similar to a prototype and those that are weakly similar to it is in agreement with the BCM theory. Besides, we experimentally showed that the prototypes learned by $\pm$ED-WTA provide interpretability and can be used for detecting outlier and adversarial examples. All of these benefits are obtained at the expense of slightly higher computation at the training phase.

**APPENDIX**

**Convergence Proof of Algorithm of Section III-C**

In this section, we prove that the fixed-point equation (14) globally converges from all initial values. Let us first simplify things by defining the following quantities:

$$\bar{x} = \frac{1}{|D|} \sum_{x \in D} x, \quad \bar{p} = \frac{1}{|D|} \sum_{x \in D} x^T w_{q(x)}$$

$$\bar{\omega} = \frac{1}{|D|} \sum_{x \in D} w_{q(x)}, \quad \bar{x} = \frac{1}{|D|} \sum_{x \in D} \| w_{q(x)} \|^2.$$ (33)

Then, the iterative algorithm that updates $\alpha$ and $u$ according to the fixed-point equation (14) can be succinctly written as

$$\alpha \leftarrow \frac{\bar{p} - u^T \bar{\omega}}{\bar{s}}$$

$$u \leftarrow \bar{x} - \alpha \bar{\omega}.$$ (34)

Now, we prove a useful lemma.

**Lemma 1:** $\| \bar{\omega} \|^2 < \bar{x}$.  

**Proof:** Since the weights $\omega$ are initialized from a trained IP-WTA network, training data are won by different neurons and the following property certainly holds:

$$0 < \frac{1}{|D|} \sum_{x \in D} \| w_{q(x)} - \bar{\omega} \|^2.$$ (35)

The conclusion follows from the following sequence of equalities:

$$0 < \frac{1}{|D|} \sum_{x \in D} \| w_{q(x)} - \bar{\omega} \|^2 = \frac{1}{|D|} \sum_{x \in D} (w_{q(x)} - \bar{\omega})^T (w_{q(x)} - \bar{\omega}) = \frac{1}{|D|} \sum_{x \in D} \| w_{q(x)} \|^2 - 2 \bar{\omega}^T \left( \frac{1}{|D|} \sum_{x \in D} w_{q(x)} \right) + \bar{\omega}^T \bar{\omega} = \bar{x} - \| \bar{\omega} \|^2.$$ (36)

Finally, we state the main result of this section.

**Theorem 1:** The iterative algorithm (34), which updates $\alpha$ and $u$ according to the fixed-point equation (14), globally converges from all initial values for $\alpha$ and $u$.

**Proof:** Writing (34) solely based on $u$, we obtain

$$u^{(k+1)} = \bar{x} - \frac{\bar{p} - \bar{\omega}^T u^{(k)}}{\bar{s}} \bar{\omega} = A u^{(k)}$$ (37)

where we have defined the nonlinear operator $A$ as

$$A u = \frac{\bar{\omega}^T u}{\bar{s}} u + \bar{x} - \frac{\bar{p}}{\bar{s}}.$$ (38)

To prove the global convergence of (34), it suffices to show that $A$ is a contraction mapping and then apply the contraction mapping principle (also known as Banach fixed-point theorem). We have

$$\| A u - A v \| = \| \frac{\bar{\omega}^T u}{\bar{s}} (u - v) \| \leq \| \frac{\bar{\omega}^T}{\bar{s}} \| \| u - v \|.$$ (39)

Appealing to the definition of operator norm, we have

$$\| \bar{\omega} \| \| u \| = \sup_{x: \| x \|=1} \| \bar{\omega} \| x \| \| \bar{\omega}^T x \| = \| \bar{\omega} \| \| \bar{\omega}^T \| \| x \| \| x \| = \| \bar{\omega} \|^2.$$ (40)

From this equation and Lemma 1 it follows that:

$$\| \frac{\bar{\omega}^T \bar{\omega}}{\bar{s}} - \frac{\| \bar{\omega} \|^2}{\bar{s}} < 1.$$ (41)

This shows that $A$ is a contraction mapping and the conclusion of the theorem follows from the contraction mapping principle (See [28, Th. 4.3.4]).

**REFERENCES**

[1] M. Du, N. Liu, and X. Hu, “Techniques for interpretable machine learning,” Commun. ACM, vol. 63, no. 1, pp. 68–77, Dec. 2019.

[2] C. Rudin, “Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead,” Nature Mach. Intell., vol. 1, no. 5, pp. 206–215, 2019.

[3] J. de la Torre, A. Valls, and D. Puig, “A deep learning interpretable classifier for diabetic retinopathy disease grading,” Neurocomputing, vol. 396, pp. 465–476, Jul. 2020.
[4] Y. Li, L. Yang, B. Yang, N. Wang, and T. Wu, “Application of interpretable machine learning models for the intelligent decision,” Neurocomputing, vol. 333, pp. 273–283, Mar. 2019.

[5] O. Li, H. Liu, C. Chen, and C. Rudin, “Deep learning for case-based reasoning through prototypes: A neural network that explains its predictions,” in Proc. AAAI Conf. Artif. Intell., 2018, vol. 32, no. 1, pp. 3530–3537.

[6] R. Agarwal et al., “Neural additive models: Interpretable machine learning with neural nets,” in Proc. 35th Conf. Neural Inf. Process. Syst., 2021, pp. 1–13.

[7] S. Blaes and T. Burwick, “Few-shot learning in deep networks through global prototyping,” Neural Netw., vol. 94, pp. 159–172, Mar. 2017.

[8] J. Snell, K. Swersky, and R. Zemel, “Prototypical networks for few-shot learning,” in Proc. Adv. Neural Inf. Process. Syst., 2017, pp. 4077–4087.

[9] H.-M. Yang, X.-Y. Zhang, F. Yin, and C.-L. Liu, “Robust classification with convolutional prototype learning,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit., Jun. 2018, pp. 3474–3482.

[10] H.-M. Yang, X.-Y. Zhang, F. Yin, Q. Yang, and C.-L. Liu, “Convolutional prototype network for open set recognition,” IEEE Trans. Pattern Anal. Mach. Intell., early access, Dec. 16, 2020, doi: 10.1109/TPAMI.2020.3045079.

[11] M. Wegkamp and M. Yuan, “Support vector machines with a reject option,” Bernoulli, vol. 17, no. 4, pp. 537–544, Nov. 2011.

[12] J. C. Platt, “Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods,” Adv. Large Margin Classifiers, vol. 10, no. 3, pp. 61–74, 1999.

[13] F. Aiolli and A. Sperduti, “Multiclass classification with multi-prototype support vector machines,” J. Mach. Learn. Res., vol. 6, pp. 817–850, May 2005.

[14] A. B. A. Graf, O. Bousquet, G. Rätsch, and B. Schölkopf, “Prototype classification: Insights from machine learning,” Neural Comput., vol. 21, no. 1, pp. 272–300, 2009.

[15] F. Chang, C.-C. Lin, and C.-J. Lu, “Adaptive prototype learning algorithms: Theoretical and experimental studies,” J. Mach. Learn. Res., vol. 7, pp. 2125–2148, Oct. 2006.

[16] O. K. Oyedotun and A. Khashman, “Prototype-incorporated emotional learning,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit., Jun. 2018, pp. 3474–3482.

[17] M. Wegkamp and M. Yuan, “Support vector machines with a reject option,” Bernoulli, vol. 17, no. 4, pp. 537–544, Nov. 2011.

[18] J. C. Platt, “Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods,” Adv. Large Margin Classifiers, vol. 10, no. 3, pp. 61–74, 1999.

[19] F. Aiolli and A. Sperduti, “Multiclass classification with multi-prototype support vector machines,” J. Mach. Learn. Res., vol. 6, pp. 817–850, May 2005.

[20] A. B. A. Graf, O. Bousquet, G. Rätsch, and B. Schölkopf, “Prototype classification: Insights from machine learning,” Neural Comput., vol. 21, no. 1, pp. 272–300, 2009.

[21] F. Chang, C.-C. Lin, and C.-J. Lu, “Adaptive prototype learning algorithms: Theoretical and experimental studies,” J. Mach. Learn. Res., vol. 7, pp. 2125–2148, Oct. 2006.

[22] O. K. Oyedotun and A. Khashman, “Prototype-incorporated emotional neural network,” IEEE Trans. Neural Netw. Learn. Syst., vol. 29, no. 8, pp. 3560–3572, Aug. 2018.

[23] K. Ghiasi-Shirazi, “Competitive cross-entropy loss: A study on training single-layer neural networks for solving nonlinearly separable classification problems,” Neural Process. Lett., vol. 50, no. 2, pp. 1115–1122, Oct. 2019.

[24] E. L. Bienenstock, L. N. Cooper, and P. W. Munro, “Theory for the development of neuron selectivity: Orientation specificity and binocular interaction in visual cortex,” J. Neurosci., vol. 2, no. 1, pp. 32–48, Jan. 1982.

[25] D. Krotov and J. J. Hopfield, “Unsupervised learning by competing hidden units,” Proc. Nat. Acad. Sci. USA, vol. 116, no. 16, pp. 7723–7731, Apr. 2019.

[26] B. Martin-del-Brío, “A dot product neuron for hardware implementation of competitive networks,” IEEE Trans. Neural Netw., vol. 7, no. 2, pp. 529–532, Mar. 1996.

[27] K. Ghiasi-Shirazi, “Generalizing the convolution operator in convolutional neural networks,” Neural Process. Lett., vol. 50, no. 3, pp. 2627–2646, Dec. 2019.

[28] P. J. Phillips, H. Wechsler, J. Huang, and P. J. Rauss, “The FERET database and evaluation procedure for face-recognition algorithms,” Image Vis. Comput., vol. 16, no. 5, pp. 295–306, Apr. 1998.

[29] P. J. Phillips, H. Moon, S. A. Rizvi, and P. J. Rauss, “The FERET evaluation methodology for face-recognition algorithms,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 22, no. 10, pp. 1090–1104, Oct. 2000.

[30] C. Chen, O. Li, D. Tao, A. Barnett, C. Rudin, and J. K. Su, “This looks like that: Deep learning for interpretable image recognition,” in Proc. Neural Inf. Process. Syst. (NeurIPS), vol. 32, 2019, pp. 8930–8941.