k-WTA Activation Function

Marco Costa, Andrea Luca Antonini

24 giugno 2020

1 Introduction

The purpose of this project is to implement a Neural Network in which we use k-WTA as activation function.
The use of k-WTA acivation is motivated for defending against gradient-based adversarial attacks. In fact, provided a labeled data item (**x**,y), the attacker finds a perturbation **x'** imperceptibly similar to **x** but misleading enough to cause the network to output a label different from y. The most effective way to find such a perturbation, i.e. adversarial example, is by exploiting the gradient information of the network w.r.t. its input **x**.
k-WTA activation function takes as input the entire output of a layer, retains its k largest values and deactivates all others to zero. If we use $f(\mathbf{x};\mathbf{w})$ to denote a k-WTA network taking an input **x** and parameterized by weights **w**, then the gradient $\nabla_{\mathbf{x}} f(\mathbf{x};\mathbf{w})$ at certain **x** is undefined. Therefore, $f(\mathbf{x};\mathbf{w})$ is C^0 discontinous. The discontinuities in the activation function also renders $f(\mathbf{x};\mathbf{w})$ discontinous w.r.t. **w**, but these discontinuities are rather sparse in the space of **w**, thus the network can be trained succesfully.
Formally, k-WTA retains the k largest values of a $Nx1$ input vector and sets all others to be zero before feeding the vector to the next network layer:

$$\phi_k(\mathbf{y})_j = \begin{cases} y_j \text{ , if } y_j \in \{ \text{ k largest elements of } \mathbf{y} \text{ } \} \\ 0 \text{ , otherwise} \end{cases} \tag{1}$$

where $\phi_k : R^N \to R^N$ is the k-WTA function (parameterized by an integer k), $\mathbf{y} \in R^N$ is the input to the activation, and $\phi_k(\mathbf{y})_j$ denotes the j-th element of the output $\phi_k(\mathbf{y})$.
Instead of specifying k, we introduce a parameter $\gamma \in \{0, 1\}$ called *sparsity ratio*. If a layer has an ouput dimension N, then its k-WTA activation has $k = \lfloor \gamma \cdot N \rfloor$. Even though the sparsity ratio can be set differently for different layers, in practice there is no gain from introducing such a variation, so we use a fixed γ.
In a Convolutional Neural Network (CNN), the output of a layer is a C x H x W tensor: in this case we will treat the tensor as a C $\cdot$ H $\cdot$ W x 1 vector input to the k-WTA activation function.
The runtime of computing a k-WTA activation is asymptotically $O(N)$, because finding k largest value in a list corresponds to finding the k-th largest value, which has $O(N)$ complexity.
Concerning the training of k-WTA networks, we know that when the sparsity ratio γ is relatively small (≤ 0.2), the network training converges slowly. In fact,

a smaller γ activates fewer neurons, effectively reducing more of the layer width and, therefore, the network size. Nevertheless, we prefer a smaller γ because it usually leads to better robustness against finding adversarial examples.
Theoretically speaking, consider one layer outputting values $\mathbf{x}$, passed through a k-WTA activation, and followed by the next layer whose linear weight matrix is W. We define the k-WTA *activation pattern* under the input $\mathbf{x}$ as:

$$\mathcal{A}(\mathbf{x}) := \{i \in [l] \mid x_i \text{ is one of the } k \text{ largest values in } \mathbf{x}\} \subseteq [l]$$

where we use $[l]$ to indicate integer set $\{1, ..., l\}$.
Even an infinitesimal perturbation of $\mathbf{x}$ can change $\mathcal{A}(\mathbf{x})$: some element i is removed from $\mathcal{A}(\mathbf{x})$, while another element j is added in. Corresponding to this change, in the evaluation of $W\phi_k(\mathbf{x})$, the contribution of W's column vector W_i is useless, while another column vector W_j suddenly takes effect. It's this change that makes the result of $W\phi_k(\mathbf{x})$ C^0 discontinous.
Once W is determined, the discontinuity jump depends on the value of x_i and x_j, that are respectively the input value that does no more affect the result and the new value that actually has effect. We note from previous experiments that the smaller γ is, the larger the discontinuity jump will be: for this reason, we prefer a smaller γ, that will make the search of adversarial examples harder.
If the activation patterns of all layers are fixed, then f($\mathbf{x}$) is a linear function, but when the activation pattern changes, f($\mathbf{x}$) switches from one linear function to another linear function. Over the entire space of $\mathbf{x}$, f($\mathbf{x}$) is *piecewise linear*. The specific activation patterns of all layers define a specific linear piece of the function, i.e. a *linear region*: in k-WTA, these linear regions are disconnected. If the linear regions are densely distributed, a small perturbation $\Delta\mathbf{x}$ from any data point will likely cross the boundary of the linear region where $\mathbf{x}$ is. Whenever boundary crossing occurs, the gradient becomes undefined.
The rationale behind k-WTA activation is to destroy network gradients, that are information needed in *white box attacks*.
k-WTA is able to universally improve the white-box robustness, regardless of the training method. The k-WTA robustness under *black box attacks* is not always significantly better than other activation function networks (e.g. ReLU).