I. THEORY

A. Neural network

A neural network consists of neurons, also called nodes, in a network which are inspired by the neurons in the brain (**?**). The node takes input from the neighbouring nodes and uses an activation function f to calculate the input nodes with the weights w that describes the connection between the nodes. The output y that is connected to a set of nodes x_i with weights w_i is gives an

$$y = f\left(\sum_{i=1}^{n} w_i x_i\right) = f(u) \tag{1}$$

and is illustrated in Figure 1a. A neural network consist of layers with connected neurons, or also called nodes.

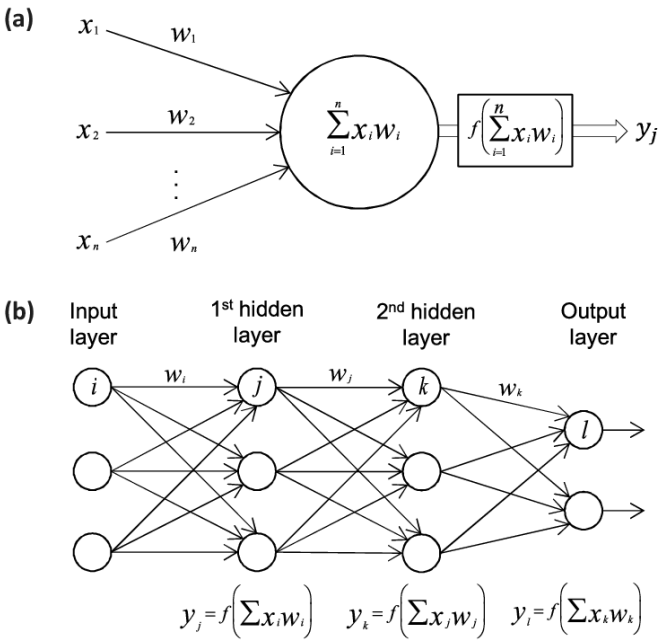

FIG. 1: Neural network, from (**?**)

The output y of a node is given via the activation function f as

$$y = f\left(\sum_{i=1}^{n} w_i x_i + b_i\right) = f(z), \tag{2}$$

where x_i is the input from the neurons in the preciding layers, and b_i is called the *bias* which is included in the case of a zero weight w_i or input x_i in the activation function. In e.g the first layer of a neural network we calculate the weighted sum z_i^1 for each node in the layer over the input coordinates x_j as

$$z_i^1 = \sum_{j=1}^{M} w_{ij}^1 x_j + b_i^1, \tag{3}$$

where M denotes all the possible inputs to a given node i in the first layer. We call the layer *fully connected* if all the nodes in the layer takes all the nodes from the previous layer as input, and M is then equal to the number of nodes in the previous layer. The activation function f_i for each node i takes the value of the activation function in Eq. (3) and gives us the output y_i^1 of all neurons in layer 1 as

$$y_i^1 = f(z_i^1) = f\left(\sum_{j=1}^{M} w_{ij}^1 x_j + b_i^1\right), \tag{4}$$

with the assumption that all the nodes in the same layer has the same activation function, and we therefore drop the subsctipt i on f_i. If the layers has different activation functions we can denote it with a superscript l for a layer l. The output of the node i for layer l therefore becomes

$$y_i^l = f(z_i^l) = f\left(\sum_{j=1}^{N_{l-1}} w_{ij}^l y_j^{l-1} + b_i^l\right), \tag{5}$$

where N_{l-1} is the number of nodes in the previous layer if the layer is fully connected. With Eq. (5) we can calculate the output y_i^l from layer l from the output y_i^{l-1} from the previous layer $l-1$. Given the input x_n from the input layer with n nodes, the output from layer $l+1$ can therefore be calculated as

$$y_i^{l+1} = f^{l+1}\left[\sum_{j=i}^{N_l} w_{ij}^{l+1} f^l\left(\sum_{j=i}^{N_{l-1}} w_{jk}\left(\right.\right.\right.$$
$$\left.\left.\left. ... f^1\left(\sum_{j=i}^{N_0} w_{mn}^1 x_n + b_m^1\right)...\right) + b_j^l\right) + b_i^{l+1}\right]. \tag{6}$$

Eq. (6) shows that the MPL is nothing more than an analytic function which is given by the real valued map $\hat{x} \in \mathbb{R}^n \to \hat{y} \in \mathbb{R}^m$. This concludes the feed forward method. Now we go over to the famous back propagation algorithm (cite) which is used to update the weights and biases in the neural network. To summarize the weights w_{ij}^l and biases b_j are updated according to

$$w_{jk}^l \leftarrow w_{jk}^l - \eta \delta_j^l y_k^{l-1} \tag{7}$$
$$b_j^l \leftarrow b_j - \eta \delta_j^l, \tag{8}$$

where δ_j^l is the error that is calculated from the derivative of the cost function $\mathcal{C}$ and η is the learning rate. A more detailed description of the back propagation algorithm can be found in (**?**).

B. Discovery statistics and the profile likelihood ratio in particle physics

In particle physics we often test a so called *background hypythis* that test the physics we know against the *signal + background hypotisis* that test unknown physics

that could possible discover by rejecting the background hypothisis. One way of doing this is to calculate the discover significance Z which counts the number of standard derivations and is the square of some test statistics q_0. In a search region where there are b background events and s signal events the discover significance is given as

$$Z = \sqrt{2(s+b)\ln\left(1+\frac{s}{b}\right) - s}, \tag{9}$$

and is often used in particle physics. Another way of using the test statistics is to use the *profile likelihood ratio* and the signal strength μ which measures how strong the signal is. To test a hypothisis value for the signal strength μ we use the *profile likelihood ratio* (**?**)

$$\lambda(\mu) = \frac{L(\mu, \hat{\hat{\boldsymbol{\theta}}})}{L(\hat{\mu}, \hat{\boldsymbol{\theta}})}, \tag{10}$$

where in the numerator the value of θ that maximizes L for a specific μ is denoted by the quantity $\hat{\hat{\boldsymbol{\theta}}}$, and is therefore the conditional maximum-likelihood (ML) estimator that estimates $\boldsymbol{\theta}$, and therefore a function of μ. The denominator is the unconditional likelihood function where $\hat{\mu}$ and $\hat{\boldsymbol{\theta}}$ are their ML estimators.

By Wilks' theorem we can use the test statistics

$$q_0 = \begin{cases} -2\ln\lambda(0), & \text{if } n > b, \\ 0, & \text{otherwise} \end{cases} \tag{11}$$

The appropiate likelihood ratio in searches for new particles is the ratio of the likelihood for the signal+background hypothisis and the likelihood for the background hypothisis and is given generally as

$$Q = \frac{L_{s+b}(\mu)}{L_b} \tag{12}$$

This likelihood ratio can be expressed in terms of the background only likelihood function L_b and signal only density function L_s and is given as (**?**)

$$Q = e^{-s_{exp}} \prod_{i=1}^{n} \left(1 + \frac{s_{exp} L_s(x_i)}{b_{exp} L_b(x_i)}\right), \tag{13}$$

where b_{exp} and s_{exp} are the expected background and signal events, and the product is taken over a given experimental result x_i for each event i over the total number of events n. We insert the signal strength parameter μ in front of the expected number of signal events s_{exp}, which gives us

$$Q(\mu) = e^{-\mu s_{exp}} \prod_{i=1}^{n} \left(1 + \frac{\mu s_{exp} L_s(x_i)}{b_{exp} L_b(x_i)}\right). \tag{14}$$

The profile log-likelihood ratio $t(\mu)$ is therefore equal to

$$\begin{aligned} t(\mu) &= -2\ln Q(\mu) \\ &= \mu s_{exp} - \sum_{i=1}^{n} \left(1 + \frac{\mu s_{exp} L_s(x_i)}{b_{exp} L_b(x_i)}\right), \end{aligned} \tag{15}$$

when expressed in terms of L_s and L_b. In the profile likelihood ratio in Eq. (10) we see that the estimator value that maximizes the signal+background likelihood function is $\hat{\mu} = 1$ in the unconditional likelihood function in the denominator in Eq. (10). Using this the derivation in Eq. .. in Appendix gives us the relation

$$\begin{aligned} -2\ln\lambda(\mu) &= -2\ln Q(\mu) + 2\ln Q(1) \\ &= t(\mu) - t(1), \end{aligned} \tag{16}$$

If the estimated value $\hat{\mu}$ that maximises an *approximated* signal+background likelihood function is not equal to the expexted estimate $\hat{\mu} = 1$, as used to derive Eq. (16), we use the estimated value $\hat{\mu}$ instead in the relation above. This gives us the same relation as above, but where we replace 1 by $\hat{\mu}$ in the last term

$$\begin{aligned} -2\ln\lambda(\mu) &= -2\ln Q(\mu) + 2\ln Q(\hat{\mu}) \\ &= t(\mu) - t(\hat{\mu}), \end{aligned} \tag{17}$$

which is the more general form. This gives us the test statistic

$$q_0 = \begin{cases} t(0) - t(\hat{\mu}), & \text{if } n > b_{exp}, \\ 0, & \text{otherwise}, \end{cases} \tag{18}$$

and the discovery significance Z is then given as

$$Z = \sqrt{q_0} = \begin{cases} \sqrt{t(0) - t(\hat{\mu})}, & \text{if } n > b_{exp}, \\ 0, & \text{otherwise}. \end{cases} \tag{19}$$

C. Likelihood estimation

The likelihood ratio function can be estimated by a neural network with a method developed by G. V. Moustakides and K. Basioti (**?**) which transforms the output of the neural network to a likelihood ratio by solving an optimalization problem. For a random vector X with the probability function $f_0(X)$ we want to optrimalisze the following loss function

$$\mathcal{J}(u) = \mathrm{E}_0[\phi(u(X)) + \mathrm{r}(X)\psi(\mathrm{u}(X))], \tag{20}$$

where $\mathrm{E}_0[\cdot]$ is the expectation value w.r.t. $f_0(X)$, and $\phi(z)$ and $\psi(z)$ are scalar functions of the scalar z, and $\mathrm{u}(X)$ and $\mathrm{r}(X)$ are scalar functions of X were $\mathrm{r}(X)$ takes values in a *known* interval $\mathcal{I}_r$. The optimalization problem we want to solve is

$$\min_{\mathrm{u}(X)} \mathcal{J}(u) = \min_{\mathrm{u}(X)} \mathrm{E}_0[\phi(u(X)) + \mathrm{r}(X)\psi(\mathrm{u}(X))], \tag{21}$$

such that when we have a known scalar function $\omega(r)$ of the scalar r, we want to determine the two functions $\phi(z)$ and $\theta(z)$ such that the global minimizer in Eq. (**??**) is equal to

$$\mathrm{u}(X) = \omega(\mathrm{r}(X)). \tag{22}$$

In (**?**) they show that the nessesary condition to achive the optimilization that satisfies this condition is that

$$\phi'(\omega(\mathrm{r})) + \mathrm{r}\psi'(\omega(\mathrm{r})) = 0. \tag{23}$$

If $\omega(r)$ is stighly monotone we also have the result

$$\phi'(\omega(\mathrm{r})) + \omega^{-1}(z)\psi'(\omega(\mathrm{r})) = 0, \tag{24}$$

for $z \in \omega(\mathcal{I}_r)$ where $\omega^{-1}(z)$ is the inverse of r. They further show that when we chose a function $\rho(z)$ with the condition $\rho(z) < 0$ for all $z \in \omega(\mathcal{I}_r)$ and define $\phi(z)$ and $\theta(z)$ as

$$\psi'(z) = \rho(z), \text{ and } \phi'(z) = \omega^{-1}(z)\rho(z), \tag{25}$$

then the minima of $\phi(z)+r\theta(z)$ in Eq. (21) for $z \in \omega(\mathcal{I}_r)$ is uniquely located at $z = \omega(r)$. This is what we wanted to achive in Eq. (22). When we chose

$$\omega(z) = \frac{z}{z+1} \text{ and } \rho(z) = -\frac{1}{z}, \tag{26}$$

and use Eq. (25) we get the following expressions for $\phi(z)$ and $\theta(z)$,

$$\phi(z) = -\log(1-z) \text{ and } \psi(z) = -\log(z). \tag{27}$$

This combination of $\phi(z)$ and $\theta(z)$ gives us the binary cross-entropy method and is among the most popular methods for classification problems. When we define the scalar function $\mathrm{r}(X)$ in Eq. (20) to be the likelihood ratio

$$\mathrm{r}(X) = \frac{f_1(X)}{f_0(X)}, \tag{28}$$

of two probability distributions $f_0(X)$ and $f_1(X)$ the loss function $\mathcal{J}(u)$ in Eq. (20) becomes

$$\begin{aligned}\mathcal{J}(u) &= \mathrm{E}_0[\phi(u(X)) + \mathrm{r}(X)\psi\big(\mathrm{u}(X)\big)] \\ &= \mathrm{E}_0[\phi(u(X))] + \mathrm{E}_1[\psi\big(\mathrm{u}(X)\big)],\end{aligned} \tag{29}$$

where the expextation values E_0 and E_1 are w.r.t the probability functions $f_0(X)$ and $f_1(X)$ respectively. We are now interested in finding a way of using a neural network to estimate the transformation $\omega(\frac{f_1(X)}{f_0(X)})$ when we have two data samples $\{X_1^0, ..., X_{n_0}^0\}$ and $\{X_1^1, ..., X_{n_1}^1\}$ that are sampled from the *unknown* probability densities $f_0(X)$ and $f_1(X)$ respectively, and when $\omega(r)$ is a *known* scalar function. Two approximations is needed to do this. For the first one we replace the statistical expectation values in Eq. (29) with the averages over the available data, while for the second we replace the function $\mathrm{u}(X)$ with the output of a neural network $\mathrm{u}(X, \theta)$, where θ summatizes the network parameters. The first approximation requires a dataset with a large sample size, while the second approximation requires that the neural network is sufficently rich such that it can approximate any nonlinear function. When applying the two approximations above, the loss function in Eq. (29) is approximated to

$$\mathcal{J}(u) \approx \hat{\mathcal{J}}(\theta) = \frac{1}{n_0}\sum_{i=1}^{n_0}\phi(\mathrm{u}(X_i^0, \theta)) + \frac{1}{n_1}\sum_{i=1}^{n_1}\phi(\mathrm{u}(X_i^1, \theta)) \tag{30}$$

The optimalization problem is now transformed into an optimalization problem over the neural network parameters θ as

$$\begin{aligned}\min_{\mathrm{u}(X)} \mathcal{J}(u) &\approx \min_{\theta} \hat{\mathcal{J}}(\theta) \\ &= \min_{\theta}\left\{\frac{1}{n_0}\sum_{i=1}^{n_0}\phi(\mathrm{u}(X_i^0, \theta)) + \frac{1}{n_1}\sum_{i=1}^{n_1}\phi(\mathrm{u}(X_i^1, \theta))\right\},\end{aligned}$$

and when optimized the optimal parameters θ_o will give a neural network that has approximated the function $\omega(r)$, that is

$$\mathrm{u}(X, \theta_o) \approx \omega(\mathrm{r}(X)), \tag{31}$$

provided that it does not reach a local minima.

D. Estimating the profile log-likelihood ratio with neural network

We can use the method of estimating a likelihood ration described in the previous section when discovering new particles by estimating the profile log-likelihood ratio in Eq. (10). When we are interested to test the signal+background hypothisis against the background only hypothisis, which is given in the likelihood ratio function in Eq. (15), the likelihood ratio L_s/L_b is optimal to estimate when we have two datasets X_s and X_b which contains the signal and background data respectively