IOSR Journal of Computer Engineering (IOSR-JCE)
e-ISSN: 2278-0661, p- ISSN: 2278-8727Volume 16, Issue 3, Ver. II (May-Jun. 2014), PP 36-41
www.iosrjournals.org

Controlling the Behavior of a Neural Network Weights Using Variables Correlation and Posterior Probabilities Estimation

Hazem Migdady, Yousef Talafha, Hussam Alrabaiah*
Department of Mathematics and Computer Science, Tafila Technical University, P.O. Box 179, Tafila 66110

ABSTRACT: *In this article, a number of posterior probabilistic based equations were introduced to detect the effect of controlling the correlation between variables on the behavior of feed forward neural network weights. In this paper it was proofed that, under certain assumptions, in a feed forward neural network with backprobagation learning algorithm, the correlation between the input variables on one side and the target variable on the other, is directly proportional to the values of the connection weights from the input layer to the output layer through the hidden layer.*

Keywords. *Correlation Coefficients, Features Selection, Data Mining, Logistic Regression, Machine Learning, Neural Networks, Posterior Probability.*

I. INTRODUCTION

Arribas, Cid-Sueiro and Alberola-Lopez [1] argue that "the neural network architectures that compute soft decisions can be used to estimate posterior class probabilities". They believe that such approach "provides a confidence measure of the classifier decisions". Rojas [5] introduced a short proof shows that "neural classifiers can learn to compute posterior probabilities of classes in input space, under the condition that the neural network must be perfectly trained and have enough plasticity". Actually several proofs were provided to show that neural networks can learn to compute the posterior probability that an input x belongs to each possible class. Such proofs can be found in Bourland and Morgan [2], also in Richard and Lippmann [4]. In this article, those proofs are used to show how controlling the correlation between the input variables and the target variable would control the behavior of the connection weights in the neural classifier. In this context we provide a proof shows that, under certain assumptions, in a feed forward neural network the connection weights that connect an input variable to the target variable are directly proportional to the correlation between that input variable and the target variable. Hence; increasing the correlation between the input variable and the target variable, increases the connection weights that connect the input variable to the target variable and vice versa.

This paper is organized as follows: In section 2 we introduce some basics to be used in the later parts. Section 3 contains two subsections to show the effect of correlation controlling on the input and hidden weights in a neural network. The paper ends with the conclusion.

II. PRELIMINARIES

In this section some basics of probability will be introduced to show how a logistic regression function can be used to compute the posterior probability between two variables. Moreover, we will show how a multilayer feed forward neural network learns the posterior probability between two variables. moreover we will show how a multilayer feed forward neural network learns the posterior estimation and satisfies the concept of Chain Rule.

Multilayer feed forward neural networks with backprobagation learning algorithm uses the logistic regression function which has convenient mathematical properties for learning tasks. Russell and Norvig [8] believe that "the output of the logistic regression function, being a number between 0 and 1, can be interpreted as a probability" of belonging to a particular class. According to Duin and Tax [3] it is possible to write the logistic regression function with respect to the posterior probability as:

$$p(Q|Z) = \frac{1}{1+e^{-Z}} \qquad (1)$$

Where Z is a multiple regression function. Duin and Tax [3] argue that equation (1) "maximizes the likelihood of the classified training objects" by optimizing the weights in Z. But equation (1) expresses the case of single perceptron neural network. Hence; in the case of multilayer feed forward neural network there is a combination of a group of single perceptron. Multilayer feed forward neural network is based on the logistic regression to learn the posterior probability between the input variable and the target variable, taking into account that the input variable is connected to the target variable indirectly through the hidden variable (hidden layer). In order to illustrate the process of multilayer feed forward neural network we will assume a multilayer neural network with single input variable x, single hidden output h, and single actual output y (See Figure 1 Below). Also we

will assume that there is a single input weight w_x and a single hidden weight w_h. Hence; the input combination to produce the hidden output is: $A = w_x x$, and the hidden combination to produce the actual output is: $B = w_h h$.

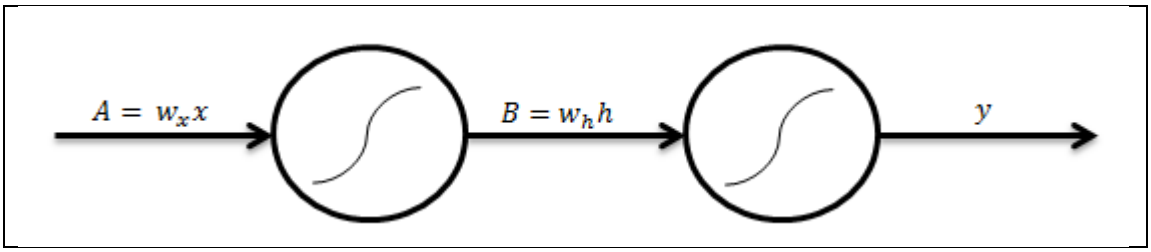

Figure 1: Single Input/Single Output Neural Network

In this context, what happens in the multilayer neural network is that: the input combination A is fed into the logistic regression process in the hidden layer to produce the hidden output h, which in turn is used to produce the hidden combination B to be used later to produce the actual output y of the neural network. However, in the sense of posterior probability concept, the variables A, B and y are considered as three nested variables, which satisfy the definition of chain rule.

Russell and Norvig (2003) found that, in probability, it is possible to use the chain rule to calculate the value of any member of a joint distribution. Thus:

$$p(V_1, \dots, V_n) = p(V_n | V_{n-1}, \dots, V_1) \cdot p(V_{n-1}, \dots, V_1)$$
$$p(V_1, \dots, V_n) = p(\cap_{k=1}^{n} V_k) = \prod_{k=1}^{n} p\left(V_k \middle| \cap_{j=1}^{k-1} V_j\right)$$

Applying the chain rule concept over the former variables: A, B and y would make it easier to understand the relation between those variables and the weights of the neural network.
Hence

$$p(y, B, A) = p(y|B, A)p(B|A)p(A) \qquad (2)$$

or:

$$p(y, B, A) = p(y|B, A)p(A|B)p(B) \qquad (3)$$

Both equations (2) and (3) are equivalent. What we need to show here is that: controlling the amount of $p(y|x)$ would affect the amount $p(y, B, A)$ and consequently affects the weights of the neural network. But the amount $p(y|x)$ is not available explicitly in equations (2)and (3) above. Instead we have the amount $p(y|A)$, which is exactly the same as $p(y|x)$. In this context, we believe that: $p(y|x) = p(y|A)$, and this is true as long as each value in variable x has only one corresponding value in A. Actually this condition holds because the function that maps feature x into the combination A ($f: x \to A$) is one-to-one function. It is possible to proof that by the following: $w_i x_1 = w_i x_2 \implies x_1 = x_2 : w_i \neq 0$.

In the next section, equations (2) and (3) will be used to show how the amount $p(y|A)$ affects the amount $p(y, B, A)$ and consequently the weights of the neural network.

III. THE EFFECT OF CONTROLLING VARIABLES CORRELATION ON NEURAL NETWORK WEIGHTS

As mentioned earlier in the introduction, Rojas [5] provided a short proof shows that a feed forward neural network learns the posterior probabilities of the classes in the target variable.

In the next two subsections we will show how controlling the posterior probabilities affects the behavior of the weights in a neural classifier.

(I) THE EFFECT OF CONTROLLING VARIABLES CORRELATION ON INPUT WEIGHTS

In the analysis below we will show how the amount $p(y, B, A)$ is affected by controlling the amount $p(y|A)$ and how this is reflected on the behavior of the input weights in the neural classifier.
In this sense, by recalling equation (2)

$$p(y, B, A) = p(y|B, A)p(B|A)p(A)$$

But, by using the general conditional probability [6], it is possible to rewrite the amount $p(y|B, A)$ as the following:

$$p(y|B, A) = \frac{p(B|y, A)p(y|A)}{p(B|A)} \qquad (4)$$

Hence

$$p(y,B,A) = \frac{p(B|y,A)p(y|A)}{p(B|A)}p(B|A)p(A)$$

Therefore, we have

$$p(y,B,A) = p(y|A)\cdot C \qquad (5)$$

where $C = p(B|y,A)p(A)$. As equation (5) illustrates, the amounts $p(y,B,A)$ and $p(y|A)$ are directly proportional to each other.

In equation (5) it is obvious that the probability of the nested events in a multilayer feed forward neural network depends mainly on two terms: $p(y|A)$ and C. Hence; controlling those two terms implies controlling the amount $p(y,B,A)$. Even though it is possible to control the term $p(y|A)$because the values of event y and input variable x $(A = w_x x)$ are known in advance, this is not possible for amount C which depends on the value of the hidden combination B, because the values of the hidden output h $(B = w_h h)$ are unknown in advance (i.e. before starting the training of the neural network), and they also change after each epoch. In this sense and because it is unknown for certain whether the amount C will decrease, increase or stay as is, we will assume that the amount C is constant for the remainder of this analysis.

Now we need to show how the amount $p(y,B,A)$ affects the weights of the neural network. We will start by rewriting equation (2) in the form of odds, as the following:

$$\frac{p(y,B,A)}{1-p(y,B,A)} = \frac{p(y|B,A)p(B|A)p(A)}{1-p(y|B,A)p(B|A)p(A)}$$

By taking the natural log of the odds, we will end up with the logit form:

$$\log\left(\frac{p(y,B,A)}{1-p(y,B,A)}\right) = \log\left(\frac{p(y|B,A)p(B|A)p(A)}{1-p(y|B,A)p(B|A)p(A)}\right)$$

$$= \log\left(\frac{p(y|B,A)p(B|A)p(A)}{1-p(y|B,A)p(B|A)p(A)}\right) = \log(p(y|B,A)p(B|A)p(A)) - \log(1-p(y|B,A)p(B|A)p(A)) \qquad (6)$$

By using the elementary properties of logarithmic function together with the use of the inequality $\log(1+x) \leq 1+\log x$ for all $x > 0$, we obtain

$$\begin{aligned}
\log(p(y|B,A)p(B|A)p(A)) &= \log p(y|B,A) + \log p(B|A) + \log p(A)\\
&= \log p(B|A) + \log p(y|B,A) + \log p(A)\\
&= \log p(B|A) + u\\
&= \log\frac{1}{1+e^{-A}} + u\\
&= -\log(1+e^{-A}) + u\\
&\geq -1 - \log e^{-A} + u\\
&= -1 + A + u
\end{aligned}$$

Where $u = \log(y|B,A) + \log(A)$. Therefore, we have

$$\log(p(y|B,A)p(B|A)p(A)) \geq w_x x - 1 + u \qquad (7)$$

On the other hand, By using the inequality $\log(1-x) \leq 1 - \log x$ for $x \in (0,1)$ we obtain

$$\begin{aligned}
\log(1-p(y|B,A)p(B|A)p(A)) &\leq 1 - \log(p(y|B,A)p(B|A)p(A))\\
&= 1 - (\log p(y|B,A) + \log p(B|A) + \log p(A))\\
&= 1 - (\log p(B|A) + k)\\
&= 1 - \left(\log\frac{1}{1+e^{-A}} + k\right)\\
&= 1 - (-\log(1+e^{-A}) + k)
\end{aligned}$$

$$\leq 2 - A - k$$

where $k = \log(p(y|B,A) + \log p(A))$. Thus, it follows that

$$\log(1 - p(y|B,A)p(B|A)p(A)) \leq 2 - w_x x - k \qquad (8)$$

Combining (6)- (8), it can be seen that

$$\log\left(\frac{p(y,B,A)}{1-p(y,B,A)}\right) \geq 2w_x x + k + u - 3 \qquad (9)$$

Hence, Inequality (9) contains two cases

Case (1): $\log\left(\frac{p(y,B,A)}{1-p(y,B,A)}\right) = 2w_x x + k + u - 3$. This formula shows the logit in the form of linear combination which in turn shows that the input weights are directly proportional to the logit.

Case (2): $\log\left(\frac{p(y,B,A)}{1-p(y,B,A)}\right) > 2w_x x + k + u - 3$. This formula does not provide enough or certain information about the relation between the logit and the weights (*i.e.* whether the weights are directly or inversely proportional to the logit).

In the analysis above, we showed how the amount $p(y|A)$ affects the amount $p(y,B,A)$ and consequently affects the amounts of the input weights.

(II) THE EFFECT OF CONTROLLING VARIABLES CORRELATION ON HIDDEN WEIGHTS

In the next analysis we will show how the amount $p(y|A)$ affects the amount $p(y,B,A)$ and consequently affects the amounts of the hidden weights.

Recalling equation (3) to show how the amount $p(y|A)$ affects the amount $p(y,B,A)$.

$$p(y,B,A) = p(y|B,A)p(A|B)p(B)$$

Hence, By using the general conditional probability Ross [6], it is possible to rewrite the amount $p(y|B,A)$ as the following:

$$p(y|B,A) = \frac{p(B|y,A)p(y|A)}{p(B|A)} \qquad (10)$$

Hence

$$p(y,B,A) = \frac{p(B|y,A)p(y|A)}{p(B|A)} p(A|B)p(B)$$

$$= p(y|A)\frac{p(B|y,A)p(A|B)p(B)}{p(B|A)}$$

Then we have

$$p(y,B,A) = p(y|A) \cdot C \qquad (11)$$

Where $C = \frac{p(B|y,A)p(A|B)p(B)}{p(B|A)}$.

As equation (11) illustrates, the amounts $p(y,B,A)$ and $p(y|A)$ are directly proportional to each other. Hence, controlling the amount $p(y|A)$ controls the amount $p(y,B,A)$ as well.

Equation (11) illustrates that the probability of the nested events in a multilayer feed forward neural network depends mainly on two terms: $p(y|A)$ and C. Hence, controlling those two terms implies controlling the amount $p(y,B,A)$. As mentioned earlier in this article, even though it is possible to control the term $p(y|A)$ because the values of event y and candidate feature x ($A = w_x x$) are known in advance, this is not possible for the amount C which depends on the value of the hidden combination B, because the values of the hidden output h ($B = w_h h$) are unknown in advance (i.e. before starting the training of the neural network), and they also change after each epoch. In this sense and because it is unknown for certain whether the amount C will decrease, increase or stay as is, we will assume that amount C is constant for the remainder of this analysis.

Now we need to show how the amount $p(y, B, A)$ affects the hidden weights of the neural network. We will start by rewriting equation (3) in the form of odds:

$$\frac{p(y,B,A)}{1-p(y,B,A)} = \frac{p(y|B,A)p(A|B)p(B)}{1-p(y|B,A)p(A|B)p(B)}$$

By taking the natural log of the odds, the logit will be produced:

$$\log\left(\frac{p(y,B,A)}{1-p(y,B,A)}\right) = \log\left(\frac{p(y|B,A)p(A|B)p(B)}{1-p(y|B,A)p(A|B)p(B)}\right)$$

$$log\left(\frac{p(y|B,A)p(A|B)p(B)}{1-p(y|B,A)p(A|B)p(B)}\right) = log(p(y|B,A)p(A|B)p(B)) - log(1-p(y|B,A)p(A|B)p(B)) \quad (12)$$

Using the same argument, we have

$$\begin{aligned}\log(p(y|B,A)p(A|B)p(B)) &= \log p(y|B,A) + \log p(A|B) + \log p(B)\\ &= \log p(A|B) + \log p(y|B,A) + \log p(B)\\ &= \log p(A|B) + u\\ &= \log\tfrac{1}{1+e^{-B}} + u\\ &= -\log(1+e^{-B}) + u\\ &= -\log(1+e^{-B}) + u\\ &\geq -1 - \log e^{-B} + u\\ &\geq B - 1 + u\end{aligned}$$

Where $u = \log p(y|B,A) + \log(B)$. Then we obtain

$$\log(p(y|B,A)p(A|B)p(B)) \geq w_h h - 1 + u \qquad (13)$$

On the other hand, by using inequality $\log(1-x) \leq 1 - \log x$ for $x \in (0,1)$, we have

$$\begin{aligned}\log(1-p(y|B,A)p(A|B)p(B)) &\leq 1 - \log(p(y|B,A)p(A|B)p(B))\\ &= 1 - (\log p(y|B,A) + \log p(A|B) + \log p(B))\\ &= 1 - (\log p(A|B) + k)\\ &= 1 - \left(\log\tfrac{1}{1+e^{-B}} + k\right)\\ &= 1 - (-\log(1+e^{-B}) + k)\\ &\leq 2 - B - k\end{aligned}$$

Where $k = (\log p(y|B,A) + \log p(B))$. Moreover, we obtain:

$$\log(1-p(y|B,A)\,p(A|B)\,p(B)) \leq 2 - B - k \qquad (14)$$

Combining the estimates obtained thus far, (12), (13) and (13) we have verified the following

$$\log\left(\frac{p(y,B,A)}{1-p(y,B,A)}\right) \geq 2w_h h + k + u - 3 \qquad (15)$$

Hence, the feed forward neural network is supposed to learn inequality (15), which contains two cases

Case (1): $\log\left(\frac{p(y,B,A)}{1-p(y,B,A)}\right) = 2w_h h + k + u - 3$. The previous formula shows the logit in the form of linear combination which in turn shows that the hidden weights are directly proportional to the logit.

Case (2): $\log\left(\frac{p(y,B,A)}{1-p(y,B,A)}\right) > 2w_h h + k + u - 3$. This formula has the form of inequality, which does not provide enough or certain information about the relation between the logit and the weights (*i.e.* whether the weights are directly or inversely proportional to the logit).

IV. CONCLUSION

According to the analysis above and under certain assumptions, in a feed forward neural network with backprobagation learning algorithm, the correlation between the input variables on one side and the target variable on the other, is directly proportional to the values of the connection weights from the input layer to the output layer through the hidden layer. Hence; controlling the correlation between the input variables and the output variable affects the weights of the neural classifier.

REFERENCES

[1] Arribas, I., J., Cid-Sueiro, J., & Alberola-Lopez, C., *Handbook of Neural Engineering.* The Institute of Electrical and Electronic Engineers, Inc. 41. 2007.

[2] Bourland, H., & Morgan, N., *Connectionist Speech Recognition* . Kluwer Academic, Boston. 1998.

[3] Duin, R., B. W., & Tax, M. J., D., *Classi_er Conditional Posterior Priority* .SSPR 98/SPR 98 Proceedings of the Joint IAPR International Workshop on Advances in Pattern Recognition.Volume 1451. pp. 611-619. ISBN: 3-540-64858-5, 1998.

[4] Richard, M. D., & Lippmann, R. P., *Neural Network Classi_ers Estimate a Posteriori Probabilities. Neural Comp.* 3 (4), 461-483, 1991.

[5] Rojas, R., *A Short Proof of the Posterior Probability Property of Classi_er Neural Networks* .Neural Computation.Vol. 8. 41-43, 1996.

[6] Ross. S., *A First Course in Probability* (9th ed.), Pearson Prentice Hall, 2012.

[7] Russell, S. J.; P., *Arti_cial Intelligence: A Modern Approach* (2nd ed.). Upper Saddle River, New Jersey: Prentice Hall, ISBN 0-13-790395-2, 2003.

[8] Russell, S. J., & Norvig, P., *Arti_cial intelligence: a modern approach* (3rd ed.). USA: Prentice Hall. 725-737, 2010.