Prediction of genetic Values using Neural Networks

Paulino Perez $^1$
Daniel Gianola $^2$
Jose Crossa $^1$

$^1$CIMMyT-Mexico $^2$University of Wisconsin, Madison.

September, 2014
Contents

1. Introduction
2. Non linear models and NN
3. Model fitting
4. Case study: Wheat
5. Application examples
High density marker panels enable genomic selection (GS).

Marker based models performs better than pedigree based models (e.g. de los Campos et al., 2009).

Most research done with linear additive models (see eq. 1).

It might be possible to increase accuracy using non-linear models with dominance and additive effects.

\[ y_i = \sum_{j=1}^{p} x_{ij} \beta_j + e_i \]  \hspace{1cm} (1)
Recent studies with non-additive effects:

Dagnachew et al. Genetics Selection Evolution 2011, 43:31
http://www.gsejournal.org/content/43/1/31

Casein SNP in Norwegian goats: additive and dominance effects on milk composition and quality

Binyam S Dagnachew¹*, Georg Thaller², Sigbjørn Lien¹³ and Tormod Ådnøy¹

doi:10.1017/S0016672312000018

Bayesian models with dominance effects for genomic evaluation of quantitative traits

ROBIN WELLMANN* AND JÖRN BENNEWITZ
Department of Animal Husbandry and Animal Breeding, University of Hohenheim, D-70599 Stuttgart, Germany
(Received 11 July 2011; revised 2 November 2011 and 7 December 2011; accepted 12 December 2011)
Including non-additive genetic effects in Bayesian methods for the prediction of genetic values based on genome-wide markers

Dörte Wittenburg*, Nina Melzer and Norbert Reinsch
Non linear models and neural networks

\[ y_i = \mu + f(x_i) + e_i \] (2)

- Any non linear function can be exactly represented as (Kolmogorov’s theorem):
  \[ f(x_i) = f(x_{i1}, \ldots, x_{ip}) = \sum_{q=1}^{2p+1} g \left( \sum_{r=1}^{p} \lambda_r h_q(x_{ir}) \right) \] (3)

- In **Neural Networks (NN)** non-linear functions are “approximated” as sums of finite series of smooth functions.

- Most basic and well known NN is the Single Hidden Layer Feed Forward Neural Network (SHLNN).
Figure 1: Graphical representation of a SHLNN.
Figure 2: Inputs (e.g. Markers) and output (phenotype) for a SHLNN.
Prediction has two (automated) steps:

- Inputs transformed non-linearly in the hidden layer.
- Outputs from hidden layer combined to obtain predictions.

\[ y_i = \mu + \sum_{k=1}^{S} w_k g_k \left( b_k + \sum_{j=1}^{p} x_{ij} \beta_j^{[k]} \right) + e_i \]

- \( g_k(\cdot) \) is the activation (transformation) function.
Parameters to be estimated in a NN are the weights \((w_1, ..., w_S)\), biases \((b_1, ..., b_S)\), connection strengths \((\beta_1^{[1]}, ..., \beta_p^{[1]}; ..., \beta_1^{[S]}, ..., \beta_p^{[S]})\), \(\mu\) and \(\sigma_e^2\).

When number of predictors \((p)\) and of neurons \((S)\) increase, the number of parameters to estimate grows quickly.

\[\implies\] Can cause over-fitting.

To prevent over fitting use penalized methods, via Bayesian approaches.
Empirical Bayes

- McKay (1995) developed Empirical Bayes approach framework for estimating parameters in a NN.

Let \( \theta = (w_1, ..., w_S, b_1, ..., b_S, \beta_1^{[1]}, ..., \beta_p^{[1]}, ..., \beta_1^{[S]}, ..., \beta_p^{[S]}, \mu)' \)

\[
p(\theta|\sigma_{\theta}^2) = \text{MN}(0, \sigma_{\theta}^2 I)
\]

Estimation requires **two** steps,

1) Obtain conditional posterior modes of the elements in \( \theta \) assuming \( \sigma_{\theta}^2, \sigma_e^2 \) known. These are obtained by maximizing,

\[
p(\theta|y, \sigma_{\theta}^2, \sigma_e^2) = \frac{p(y|\theta, \sigma_e^2)p(\theta|\sigma_{\theta}^2)}{p(y|\sigma_{\theta}^2, \sigma_e^2)} = \frac{\int_{\mathbb{R}^p} p(y|\theta, \sigma_e^2)p(\theta|\sigma_{\theta}^2) d\theta}{\int_{\mathbb{R}^p} p(y|\sigma_{\theta}^2, \sigma_e^2)p(\theta|\sigma_{\theta}^2) d\theta}
\]

which is equivalent to minimizing the “augmented” sum of squares:

\[
F(\theta) = \frac{1}{2\sigma_e^2} \sum_{i=1}^{n} e_i + \frac{1}{2\sigma_{\theta}^2} \sum_{j=1}^{m} \theta_j^2
\] (4)
2) Update $\sigma_\theta^2$, $\sigma_e^2$ by maximizing marginal likelihood of the data $p(y|\sigma_\theta^2, \sigma_e^2)$. The marginal log-likelihood approximated as:

$$\log p(y|\sigma_\theta^2, \sigma_e^2) \approx k + \frac{n}{2} \log \beta + \frac{m}{2} \log \alpha - \frac{1}{2} \log |\Sigma|_{\theta=\theta_{map}} - F(\theta)|_{\theta=\theta_{map}}$$

where $\Sigma = \frac{\partial^2}{\partial \theta \partial \theta'} F(\theta)$.

It can be shown that this function is maximized when:

$$\alpha = \frac{\gamma}{2 \sum_{j=1}^m \theta_j^2}, \quad \beta = \frac{n - \gamma}{\sum_{i=1}^n e_i^2}, \quad \gamma = m - 2\alpha \text{Trace}(\Sigma^{-1})$$

Iterate between 1 and 2 until convergence.

**NOTE:**

SIMILAR TO USING BLUP AND ML IN GAUSSIAN LINEAR MODELS.
Problems with the approach

- Huge number of parameters to estimate,

\[ m = 1 + S \times (1 + 1 + p) \]

where \( S \) is the number of neurons and \( p \) is the number of covariates.

- Gauss-Newton algorithm used to minimize (4) requires solving linear systems of order \( m \times m \), complexity \( O(m^3) \).

- Updating formulas for the variance components requires inverting a matrix of order \( m \times m \), complexity \( O(m^3) \).

Alternatives:

- Derivative free algorithms (may have poor performance, unstable).
- Parallel computing.
We developed an R package (brnn) that implements the Empirical Bayes approach to fitting a NN. It will be available in a few months in the R-mirrors.

Figure 3: Help page for the trainbr package.
Prediction of Grain yield (GY) and Days to heading (DTH) in wheat lines,

- 306 wheat lines from Global Wheat Program of CIMMYT.
- 1,717 binary markers (DArT).
- Two traits analyzed:
  1. GY (5 Environments).
  2. DTH (10 Environments).

Bayesian regularized neural networks fitted by using the MCMC approach.

Predictive ability of BRNN compared against standard models by generating 50 random partitions with 90% of observations in training and 10% in testing.
### Table 1: Correlations between observed and predicted phenotypes for DTH and GY (“winner” underlined).

<table>
<thead>
<tr>
<th>Trait</th>
<th>Environment</th>
<th>BL</th>
<th>Bayes A</th>
<th>Bayes B</th>
<th>RKHS</th>
<th>RBFNN</th>
<th>BRNN</th>
</tr>
</thead>
<tbody>
<tr>
<td>DTH</td>
<td>1</td>
<td>0.5877</td>
<td>0.5737</td>
<td>0.5342</td>
<td>0.6551</td>
<td>0.6637</td>
<td>0.6358</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>0.5772</td>
<td>0.5818</td>
<td>0.5150</td>
<td>0.6281</td>
<td>0.6072</td>
<td>0.6187</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>0.6001</td>
<td>0.6019</td>
<td>0.5392</td>
<td>0.6834</td>
<td>0.6855</td>
<td>0.6661</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>0.0165</td>
<td>0.0648</td>
<td>0.0698</td>
<td>0.1238</td>
<td>0.1623</td>
<td>0.0218</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>0.6476</td>
<td>0.6539</td>
<td>0.6342</td>
<td>0.6855</td>
<td>0.6819</td>
<td>0.6753</td>
</tr>
<tr>
<td></td>
<td>8</td>
<td>0.3585</td>
<td>0.3510</td>
<td>0.3094</td>
<td>0.4616</td>
<td>0.4589</td>
<td>0.3918</td>
</tr>
<tr>
<td></td>
<td>9</td>
<td>0.5929</td>
<td>0.5261</td>
<td>0.4692</td>
<td>0.6154</td>
<td>0.6282</td>
<td>0.6092</td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>0.5418</td>
<td>0.5593</td>
<td>0.5286</td>
<td>0.6114</td>
<td>0.6229</td>
<td>0.5709</td>
</tr>
<tr>
<td></td>
<td>11</td>
<td>0.5202</td>
<td>0.5126</td>
<td>0.4797</td>
<td>0.5823</td>
<td>0.5873</td>
<td>0.5490</td>
</tr>
<tr>
<td></td>
<td>12</td>
<td>0.4453</td>
<td>0.4504</td>
<td>0.4476</td>
<td>0.4673</td>
<td>0.3942</td>
<td>0.3543</td>
</tr>
<tr>
<td>GY</td>
<td>1</td>
<td>0.4768</td>
<td>0.4651</td>
<td>0.4390</td>
<td>0.5091</td>
<td>0.5135</td>
<td>0.5046</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>0.4755</td>
<td>0.3222</td>
<td>0.2860</td>
<td>0.5005</td>
<td>0.4286</td>
<td>0.4318</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>0.1963</td>
<td>0.2817</td>
<td>0.2619</td>
<td>0.3747</td>
<td>0.4230</td>
<td>0.3152</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>0.4549</td>
<td>0.4302</td>
<td>0.3957</td>
<td>0.5301</td>
<td>0.5538</td>
<td>0.4901</td>
</tr>
<tr>
<td></td>
<td>7</td>
<td>0.4629</td>
<td>0.4067</td>
<td>0.3745</td>
<td>0.5274</td>
<td>0.5367</td>
<td>0.5013</td>
</tr>
</tbody>
</table>

**NOTE:** Non-parametric methods better in 15/15 comparisons.
Figure 4: Plot of the correlation for each of 50 partitions and 10 environments for days to heading (DTH) in different combination of models.
# Example 1
# Noise triangle wave function, similar to example 1 in Foresee and Hagan (1997)

# Generating the data
x1 = seq(0, 0.23, length.out=25)
y1 = 4 * x1 + rnorm(25, sd=0.1)
x2 = seq(0.25, 0.75, length.out=50)
y2 = 2 - 4 * x2 + rnorm(50, sd=0.1)
x3 = seq(0.77, 1, length.out=25)
y3 = 4 * x3 - 4 + rnorm(25, sd=0.1)
x = c(x1, x2, x3)
y = c(y1, y2, y3)
X = as.matrix(x)

neurons = 2
out = brnn(y, X, neurons=neurons)
cat("Message: ", out$reason, \\
"
"
plot(x, y, xlim=c(0, 1), ylim=c(-1.5, 1.5),
main="Bayesian Regularization for ANN 1-2-1")

Note:

- Type library(brnn) and then demo('Example_1') to run this example in the R console.
Application examples

Continued...

[Graph showing data points and lines for Matlab and R]

- Prediction of genetic values using Neural Networks

SLU, Sweden
#2 Inputs and 1 output
#the data used in Paciorek and Schervish (2004). The data is from a two input one output function with Gaussian noise with mean zero and standard deviation 0.25.

data(twoinput)
X=normalize(as.matrix(twoinput[,1:2]))
y=as.vector(twoinput[,3])

neurons=10
out=brnn(y,X,neurons=neurons)
cat("Message: ",out$reason,"\n")

f=function(x1,x2,theta,neurons) predictions.nn(X=cbind(x1,x2),theta,neurons)

x1=seq(min(X[,1]),max(X[,1]),length.out=50)
x2=seq(min(X[,1]),max(X[,1]),length.out=50)
z=outer(x1,x2,f,theta=out$theta,neurons=neurons)  # calculating the density values

transformation_matrix=persp(x1, x2, z,
     main="Fitted model",
     sub=expression(y==italic(g)~(bold(x))+e),
     col="lightgreen",theta=30, phi=20, r=50, d=0.1,
     expand=0.5,ltheta=90, lphi=180,
     shade=0.75, ticktype="detailed",nticks=5)

points(trans3d(X[,1],X[,2], f(X[,1],X[,2],
     theta=out$theta,neurons=neurons), transformation_matrix), col = "red")
y = g(x) + e
Warning: This analysis can take a while,... We are selected only some markers. You can select markers based on p-values for example or try to reduce the dimensionality of your problem using G matrix as input or principal scores.

```r
rm(list=ls())
setwd("/tmp")
library(brnn)
library(BLR)
#Load the wheat dataset
data(wheat)

#Normalize inputs
y=normalize(Y[,1])
X=normalize(X)

p=300

#Fit the model with the FULL DATA, but some markers,
#You can select the markers based on p-values for example
out=brnn(y=y,X=X[,1:p],neurons=2)
cat("Message: ",out$reason,"\n")

#Obtain predictions
yhat_R=predictions.nn(X[,1:p],out$theta,neurons=2)
plot(y,yhat_R)
```
Notes:

- The function predictions.nn obtains \( \hat{y} \). This function takes as arguments the vector of estimated parameters and the number of neurons.
- The vector of estimated parameters can be obtained using the function brnn.
- The brnn software works faster in the R version developed by Revolution Analytics in Linux environments.
References


