NeuroSNP: Tool to Filter SNPs in Whole Genomic DNA

B. Zonovelli[1], **C. C. H. Borges**[1], **and W. A. Arbex**[2]
[1]Federal University of Juiz de Fora, Juiz de Fora, MG, Brazil
[2]Brazilian Agricultural Research Corporation, Juiz de Fora, MG, Brazil

Abstract— *The classification task the punctual differences found when comparing the genomes of two distinct individuals is complex. In draft genome this difference can be 7-8x higher than a final version itself genome. This difference is a noise, this can be in the input or output label. The presence this noise difficult the process of the classification the differences how being single nucleotide polymorphisms (SNPs) or not. The SNPs are specific differences between different pairs of aligned sequences and consist of the most common type of genetic variation. To task of the classification was used the technique of the neural network with excellent results for whole genome the different genomes..*

Keywords: Bioinformatics, Genomic DNA, SNP Filtering, Machine Learning, Computational Intelligence, Neural Network

1. Introduction

In this work we discuss the capacity the neural network to identify correctly the single nucleotide polymorphisms (SNPs), distinguishing them from mismatch that can be error. The SNP that are specific differences between different pairs of aligned sequences and consist of the most common type of genetic variation [5]. SNPs can cause functional or phenotypical alterations, which, in turn, can result in evolution or biochemical consequences on individuals where the SNPs are manifested.

It is known that, at each stage destined to DNA sequencing an error can be introduced, even in small portions, introducing noise which could lead to erroneous identification of an SNP. To solve this problem, filters have been constructed, linked or not of software the alignment and mapping of sequences, which are used in the assembly of the genome of a particular organism. In this scenario, there is the software MAQ (Mapping and Assembly with Quality) [13]. This program aims at mapping and assembly of complete genomes sequenced by NGS platforms, in addition to having a SNPs filter coupled.

2. Background

With the growing advancement of genome sequencing platforms, the need arises for computational models capable of analyzing, effectively the large volume of available data. The correct identification of SNPs is an important step for their use in other studies, however, for better identifying can be necessary a filtering process. The filtering of SNPs in data from next generation platforms presents as an area of research where there is a need for new developments. Specifically, filters based on strategies at machine learning and computational intelligence, which basically are not exploited. The main difficulty in the correct identification of SNPs reside in the noise present in the input and the output, ie, the values used for the training of machine learning tools contains noise, same after its application.

2.1 next-generation sequencing

In the late 70's, was developed two classical DNA sequencing methods, the chemical degradation method or procedure of [18] and the method enzymatic degradation or Sanger procedure [24]. Such techniques employ chemist process to identify and determine the order of nitrogenous bases in the DNA of an organism. But because of the ease of interpretation of the data derived from the method developed by Frederick Sanger, his technique was widely used by stakeholders in DNA sequencing. However, the high cost and low efficiency inherent in this method has become a limiting factor for projects aimed at large-scale genomic sequencing [4].

From 2005, the sequencing technologies have undergone a considerable advance, reducing costs and increasing capacity for sequencing. Today, the new sequencing platforms known as next-generation sequencing (NGS), became effective options for routine use in sequencing projects and ressequencing individual genomes [25], [10]. These platforms are a powerful alternative to the detection of variations between the target genome and the reference, to the studies of structural and functional genomics [17], [19]. Are able to generate information of millions of reads in a single run [27], [4].

The difficulty in defining the presence of noise in the filtering process of the SNPs is inherent in the generation process of data, from sequencing until assembly process. Draft genome assembly, in general, provide versions, known as drafts before the final version, as with the human genome project [6]. The draft genome contain more errors than the final version, so that the use of a filter may be of great importance. To assess the impact of noise, and the ability of machine learning tools for circumvents this work was being undertaken on the draft genome of an animal of the Fleckvieh breed, using as reference the bovine genome bosTau4.0 [12].

2.2 Noise Identification

The task of completing assembly of a genome, pass by the process of sequencing, alignment and assembly of reads. In each of these stages is there the insertion of noise, which in the discovery phase hinder the correct identification of a SNP. At this stage any difference between the sequences is an mismatches, however some of these differences are SNPs and the other no. Despite being computationally a difference, define when mismatches is or not a SNP is a complex task, this definition is at the discretion of the filtering step. Regarding the NGS platforms is known that errors are introduced in the range 0.1% of 1% [9].

By aligning two sequences with the reference genome to generate consensus, the software alignment and assembly can identifies a mismatches in the first positions of the fragment. However, this may be the best alignment for the fragment data. This situation typically occurs when used reads are short, which is useful when is used the NGS platforms. The mismatches generated by this alignment may be the result of an error in the sequencing step, or a SNP. The mismatches can be generated by an alignment error resulting from sequencing error [16].

3. Methods

3.1 Reassembly

The processes of discovery and filtering are executed always after assembly of the genome, so the need to perform this phase of the project. A project of assembling a genome can be extensive. So, to make it possible to follow the steps of discovery and filtering of SNPs, the genomes of two distinct species were reassembled using the MAQ software. The stage of discovery generates the necessary file for the study of false positives. The SNPs filter of the MAQ software, SNPfilter, and the filter implemented in this article use this file to perform the filtering step.

The genome is the an animal of the species Bos taurus, Fleckvieh breed, which was sequenced using NGS [8]. The reassembled genome was sequenced using the Solexa Genome Analyzer II platform, generating 24 giga bases of sequence with size 36bp mate-pair after trimming, resulting in an assembly with 7.4x average depth. Was used as reference the assembly bosTau4.0 the bovine genome, sequenced by Baylor College of Medicine and made available by the University of California at Santa Cruz [12]. Being found 6,599,143 SNPs in the discovery and 2,162,709 after filtering step.

3.2 Neural Network

For the filtering work was chosen the technique of artificial neural networks that comprise a computational resource often used for the solution of various problems, including bioinformatics and biological problems [26], [11], [7], [22], [15], [3]. However, in research carried no were references found to the use of neural networks to filter SNPs in complete genomic DNA, sequenced in NGS platforms. The artificial neural network was the technique of computational intelligence chosen, because her classification ability is one of its main characteristics and can therefore be used in the assembly of a filter, which is nothing more than a classifier.

3.2.1 Resilient Network

The concept of resilience, or resilient, can be defined as something or someone with the ability to adapt to an unexpected situation, so flexible. Applying this concept to the network, a resilient network has the ability to adapt in the best way, the data presented. Resilient networks do not require that the rate of learning to be informed, because it is updated by the learning algorithm developed. Thus solving one of the main difficulties presented, which is the definition of the learning rate [1]. The algorithm used to build a resilient network was (**R**esiliente back**prop**agation) [23]. The parameters used by the algorithm, by definition of the authors, are started with: $\Delta_0 = 0.1$; $\Delta_{max} = 50.0$; $\Delta_{min} = 1e^{-6}$; $\eta_+ = 1.2$; $\eta_- = 0.5$.

3.3 Filter Implementation Using Neural Networks

Classification models supervised for filtering SNPs are not yet explored in the literature. Possible reasons are the difficulty of having a reliable database for both false positives how to for SNPs proven to obtain the assumption of generalization. Thus, any attempt to use supervised classification for filtering SNPs must necessarily pass through the definition of a strategy for the construction of training base and / or determination of the class of instances, these strategies aim to minimize the noise present in the training set and / or testing. To mount the databases three strategies were defined: i)Using a pre-filter for determining the classes; ii) construction of specific bases to maximize the power of generalization of supervised classification tool; iii) construction of specific bases using some rules of pre-filtering. In order that each of the strategy generated a different model.

It is noteworthy that both strategies suffer from problems relating to noise in determining the class (or false positive SNPs) of each candidate. Thus, we seek to evaluate the potential of one of the models of neural networks to circumvent this feature of the problem of detecting SNPs. The library Fast Artificial Neural Network (FANN) [20] was used for the encoding of the model. The library allows the creation of a network using a number of different programming languages, the C language was chosen for this work. The process of assembling a computational model for filtering SNPs can be described in the following steps:

1) Riding a dataset for training and testing;
2) Training several networks with the given set of data;
3) Analysis of the results obtained with the training and choose the best network;

4) The filter program, reads the selected network and filters SNPs;
5) Analysis of the results provided by the filter;
6) When necessary, redo the process.

The definition of the structure of the network, passes by the choice of activation function that best adapts to the problem. Among the various activation functions available, four were chosen. The logistic sigmoid due to its extensive use. The Gaussian function to be of general use. The sigmoid Elliot function can have a lower mathematical complexity, so that it is expected to be faster than the sigmoid logistic, and the Elliot symmetrical.

The topology used consisted of a network with ten neurons in the input layer, a hidden layer with twenty neurons, this value was chosen randomly. The output layer with one neuron, initially binary, classifying SNPs in 0 or 1, simulating the behavior of the MAQ software filter. For the training algorithm was used resilient backpropagation, making it not necessary to define several learning rates. The data set used was reassembled genome of the Bos Taurus. The network was implemented following the code patterns informed at the manual FANN [20]. Is presented, then the first model.

3.3.1 First Model

The first model based on machine learning to be presented for filtering SNPs is part of of the line the classification with noise, where basically the framework of the classification of SNPs is presented polluted by noise. This noise is introduced by a pre-classification needed to be used for classification of candidate SNPs. Logically, this pre-classification is failures, otherwise would not require an adjustment by means of a strategy of machine learning.

In this work, it is natural to use the filtering achieved by the logical expressions of the MAQ filter as a first assessment of SNPs and false positives. Thus, one can form a training base with the output of the instances being defined by MAQ output filter. The noise of the instances that are misclassified by the filter and thus hinder the process of learning the strategy used in this case neural networks.

The expectation is that the use of additional variables not used in MAQ filter as well as the potential of a neural network with one or more inner layers represent nonlinear functions can generate a classification hypothesis that can be properly generalize the filtering minimizing the effect classroom noise levels.

The dataset used was extracted from the output files of the MAQ software. The first file is derived from the discovery stage and the second stage of the filter. In these steps the reassembled genome Bos Taurus was used, then the first file had $\approx$ 7 millions the SNPs and the second $\approx$ 2 millions. The file used for automatic training of the neural network by FANN library has a specific format. Therefore, we developed a PHP script that scans the file first randomly selecting 4,000 SNPs per chromosome for the training set and 2,000 for the test set. If the SNPs are present in the second file it was indicated as 1 or how 0 otherwise. Getting this set so as SNPs 1 and 0 as error.

The setting values of 4,000 and 2,000 for samples SNPs training and testing. Were chosen after initial tests with several different values. The values tested were from 2/3 and 1/3 of the total, up to only 100 and 50 SNPs for chromosome, and the value of 4,000 and 2,000 were very close to the results obtained by the values of 2/3 and 1/3 total.

The first model was the basis for defining the best network structure for the development of the work, the resilient backpropagation algorithm is efficient, and the activation function ELLIOT was the one that achieved the best results, with errors next to 0.6%. However, this first strategy considered MAQ as correct, and suffered from the presence of noise in the data set used for training. The expectation is that the generation of new datasets, based on knowledge gained on the functioning of alignment and assembly programs, reducing this noise occurs.

The filter the MAQ software is binary, classifying SNPs as 0 or 1, ie, true or false positive. Thus the characteristic function is a threshold function, However, the implemented and trained network achieved better results with the sigmoid function, that ranked SNPs in the range [0, 1], this characteristic has been exploited generating an important feature of the filter, defined here as a restriction. Were defined three constraints, LOW, MEDIUM and HIGH. The restriction LOW allows any SNPs is classified by the network, if it has a different value than 0. MEDIUM constraint behaves like the threshold function, ie all SNPs rated value above 0.5 is considered true. HIGH constraint, only ranked SNPs with a value of 1.

3.4 Second Model

Although the first model has shown interesting results with respect to the classification process using polluted classes, initial experiments indicated that the neural network did not provide an generalization ability to obtain a better determination of SNPs and false positives by regarding determined MAQ reference filter.

However, it has been the target of a machine learning tool with proper training can be competitive compared to traditional filters. In this second model, we intend to enhance the generalization capability of machine learning model by attempting to diminish the influence of the noise arising from the pre-filtering used.

Thus, we seek to replace the use of MAQ filter on pre-filtering for “rules” more stringent for determining SNPs and especially false positives. The new form of filtering, in this way, will be less sensitive to borderline cases. Thus, these rules shall establish a training base with greater definition mainly of false positives. It is hoped that in this way, has

become benefits in the process generalization with greater ease in learning and discrimination of new instances.

The determination of the rules for generating the class of basic training, like any filter, will be subject to noise. The expectation is that they will become stricter in the detection of false positives that, for this particular problem, is the most data. Thus, we expect a better reflection of the results, with the generalization by providing a more accurate filtering.

The data sets used were designed based on two rules. The first rule defines a group of SNPs with high confidence, and the second group with low trust, where trust is understood by how much an mismatches could be considered a SNP. The objective is to define a SNP with high confidence as being true, and a SNP with low confidence as a mistake. SNPs that are not in either group will be classified by the network. Based on the above tests, the same topology has been defined for the second model, using the sigmoid activation function Elliot, with time constant of 0.5. Thus, as in the previous step, the genome reassembled Bos Taurus was used.

3.4.1 Generation of Data Sets

The first rule for the generation of the data set used in training is the choice of SNPs with high confidence. The rule was set after parameter analysis and genome study. The parameters are the twelve columns of the output file MAQ software. The first two columns identify the SNPs, therefore, are not used in the filters. The other 10 have diverse information, four of which inform the nucleotides present in the reference genome and consensus genome, so these columns are not considered when selecting high confidence SNPs. The mean value between the second and third best call was not used because it is a feature of the MAQ software information. The value of hit was not used because, according [13], this variable can create doubt when the filter therefore is among the parameters presented to the neural network, however, but was not considered in the selection of SNPs for mounting data set.

The choice of SNPs with high trust followed the following criteria: depth greater than or equal to 6 (the bovine genome has an average depth of 6.98 so the choice of SNPs that are near to or above); Phred-like greater than or equal to 20; quality and mapping quality on the flank of 6 greater than or equal to 50, this value was also used by [14] in their work. Therefore, 429.078 SNPs were found in the total pool, originating from the discovery of the software before MAQ filter which satisfy these criteria file.

The construction of the group of SNPs with low confidence used the same parameters in the group with high confidence. The criterion for the determination of SNPs low confidence consists in the withdrawal of the total data set, SNPs having at least one parameter equals 0, which satisfy the criterion 1,821,527 SNPs.

The data set consists of a mounted file workout with 116,000 entries and a test file with 58,000, consisting in a balanced manner, ie, deriving half of the dataset with high trust and the other half of the set with low confidence. Moreover, we used the same number of SNPs for each of the 29 chromosomes present in bovine genome studied.

3.5 Third Model

In this third model, it is intended, as in the second, improve the generalization of the machine learning model. The difference between the second and third model is the rule for selecting SNPs with high confidence that this model is less restrictive. The difference is also in the non-consideration of SNPs that have a null parameter. The data set was constructed based on two rules that are described below. As in previous cases, the same topology defined for the first and second models will be used. Also as in the previous step, the genome reassembled Bos Taurus was used.

3.5.1 Generation of Data Sets

The choice of SNPs with high trust followed the following criteria: greater than or equal to 6 deep; Phred-like greater than or equal to 20; mapping quality greater than or equal to 40 and greater quality in the flank or equal to 20. We notice that the values used are the same as the MAQ filter, except for the depth value.

SNPs with low trust, do not have different criteria for the second and third model. That is, the same SNPs were considered to be of low confidence used in the assembly of the sets of data from both models, the second and third.

The two new sets of data have different contents, however, were set identically. Each dataset consists of a training file with 116,000 entries and a test file with 58,000. As the second model bases are balanced with half the originating data set with high reliability and half of the assembly with low confidence. Remains the same representation bases for the 29 chromosomes present in the bovine genome studied.

3.6 Training Second and Third Models

Both models were trained, each with its particular set of data. Ten held trainings, selecting the best for the construction of training and testing graph. Unlike the first model graph, you can see a small increase in test curve, while the curve of workout keeps reducing. This phenomenon is indicative of stopping the training process. Elliot function, converged quickly to a great result, as expected.

The algorithm implemented for training saves the network with their synaptic weights, when the error in the test is less than the previous error, this time indicated by the blue arrow. Thus, the parameters of the neural network with the best performance is stored for later use. Even if the algorithm does not stop training, the network is stored one that had the lowest error on the test.

The results in the training phase of the three models using different databases, indicate a very different behavior between them. It is not trivial to identify which model

provides higher quality results. Intends to apply neural networks generated for each model in complete genomes can thus get a better indication regarding the quality of the bases used in the generation of neural networks. The following shows the construction of the filter that uses neural networks trained for subsequent application to whole genomes.

3.6.1 Implementing NeuroSNP

After completion of the training step, the next step was the development of the filter itself. The filter consists of an algorithm that reads one trained network and filter the SNPs of the source file, generating a new file with the SNPs that passed the filter. With all the initial tests completed, the filter based on computational intelligence techniques, called NeuroSNP, was completed and the call itself now requires four parameters, explained in the table 1

Table 1: Parameters NeuroSNP

Parameter	Description
-n	Output file of training the network. This file contains the structure of the trained network
-o	Output file, the SNPs considered positive are saved in this file.
-d	Source file of the SNP by default and the output file MAQ Software.
-r	Restriction (0 - LOW, 1 - HIGH, 2 - MEDIUM).

The NeuroSNP gets the table parameters 1 at the time of your call. The first action of the filter is to reassemble the neural network with its weights. The parameter $-n$, contains the path to the output of the network training phase, and this file used to reassemble the network. Then the filter starts reading the file with SNPs, parameter $-d$. Each SNPs contained in the file has 12 columns with their identification and characteristics of assembly and alignment, then the filter reads the columns and informs from data contained in the network. The network returns an output value if the returned value satisfies the constraint informed, parameter $-r$, then the SNPs with your data and stored in the output file, parameter $-o$.

4. Results

When these tests were performed, contained in NCBI 13,704,221 SNPs submitted and 3,003 valid for animals breed Bos Taurus, last accessed 02/2013. are extracted the nucleotide sequences of the dbSNP, used for locally building the database used by software BLAST. The database contains all SNPs from NCBI to breed animals Bos Taurus.

To build the database, the BLAST uses, were assembled reads with 120bp in size, where the polymorphic variation present in each of SNPs generated a new reads, with the SNP at position 60. To analyze the results, only the alignments without gaps or mismatechs were accepted with 100% similarity, and always in the direction $5' \rightarrow 3'$ and size of 120bp. To compare different filters was used statistical measure known as odds ratio (OR) [2].

4.1 Results Obtained by the First Model

The first model used four different activation functions with three time constants. However, only the best structure of the first model was chosen. The structure chosen uses the Elliot activation function and time constant equal to 0.5. The results of the first model are shown in Table 2. The selected networks in this model, the NeuroSNP1.A and NeuroSNP1.B, obtained the following training errors: 0.003560 e 0.011363.

When analyzing the table 2 it is possible to see that the value 5.3746 obtained by calculating the OR for SNPfilter was only exceeded by the value of 6.0669 NeuroSNP1.B with HIGH restriction. However, the MEDIUM and LOW restrictions, have lower values OR (4.8398 and 3.4714), indicating that the NeuroSNP1.B was less efficient. For the increasing number of filtered SNPs or sample, did not generate an equal increase in the number of valid alignments. The same behavior is observed for the NeuroSNP1.A, with a value of 5.0302 OR with HIGH restricted, and lower values for the MEDIUM and LOW restrictions (4.3026 and 3.5126).

Table 2: Comparison between Model First and SNPfilter.

	SNPs	alignments	OR	CI
MAQ	6,599,143	2,162,709	-	-
SNPfilter	2,174,341 (32.95%)	1,573,706 (72.77%)	5.3746	5.3565 - 5.3929
		NeuroSNP1.A		
HIGH	1,878,258 (28.46%)	1,334,174 (61.69%)	5.0302	5.0124 - 5.0480
MEDIUM	2,243,455 (34.00%)	1,519,172 (70.24%)	4.3026	4.2887 - 4.3166
LOW	2,725,354 (41.30%)	1,720,551 (79.56%)	3.5126	3.5022 - 3.5229
		NeuroSNP1.B		
HIGH	1,557,915 (23.61%)	1,164,256 (53.83%)	6.0669	6.0429 - 6.0910
MEDIUM	2,001,787 (30.33%)	1,405,903 (65.01%)	4.8398	4.8232 - 4.8565
LOW	2,809,366 (42.57%)	1,765,863 (81.65%)	3.4714	3.4613 - 3.4815

Another factor to be observed is the CI, that in all cases remained low, almost zero if the value is rounded to one decimal place only. The small CI indicates that the OR calculated is accurate, being extremely significant.

Thus, using this first model only was superior to SNPfilter when used HIGH restriction, the value obtained by OR NeuroSNP1.A is close to the SNPfilter, and the NeuroSNP1.B is superior. Therefore, the network trained using the class of each instance as the output SNPfilter not shows promise for the classification of mismatches. However, the expectation is that neural networks are able to perform this task satisfactorily, training with the most promising bases being only necessary.

4.2 Results Obtained by the Second Model

The second model was trained with a set of data assembled from the rules given in section 3.4.1. The table 3 shows the comparative results between the second model and SNPfilter. The selected networks obtained the following errors: 0.000646 for NeuroSNP2.A and 0.000791 for the NeuroSNP2.B. As you can see the two values are very close. Analyzing table 3, is possible to see that both networks can classify the mismatches very efficiently, obtaining higher

ORs then of SNPfilter values in the three restrictions, and primarily values near ORs between the restrictions, indicating that the second model is stable.

As can be observed, the correct classification of mismatches is a difficult task because two networks with next errors have very different final results. The search space traversed by the network in the optimization of the error may have many minimum local , possibly close to the global minimum, thus generating networks with low errors, but with variations in the classification stage. Another hypothesis is that the proximity between the candidates is large, making two networks, with errors very close, may have different behaviors for the same dataset. For this analysis it suffices to observe the sample size, which has a moderate variation in NeuroSNP2.A, and A larger variation for NeuroSNP2.B. As in the first model, the second model networks have small CIs, showing that OR calculated is accurately.

Table 3: Comparison between SNPfilter and the Second Model.

	SNPs	alignments	OR	CI
MAQ	6,599,143	2,162,709	-	-
SNPfilter	2,174,341 (32.95%)	1,573,706 (72.77%)	5.3746	5.3565 - 5.3929
		NeuroSNP2.A		
HIGH	209,875 (03.18%)	164,320 (07.60%)	7.3993	7.3220 - 7.4774
MEDIUM	398,005 (06.03%)	308,975 (14.29%)	7.1191	7.0649 - 7.1736
LOW	658,551 (09.98%)	507,243 (23.45%)	6.8769	6.8359 - 6.9180
		NeuroSNP2.B		
HIGH	81,797 (01.24%)	61,480 (02.84%)	6.2074	6.1092 - 6.3072
MEDIUM	408,590 (06.19%)	314,781 (14.55%)	6.8834	6.8321 - 6.9350
LOW	1,143,865 (17.33%)	853,942 (39.48%)	6.0420	6.0148 - 6.0694

The use of the second model proved to be more promising than the first. It is noteworthy, also, that the second model features superior results to those obtained using the SNPfilter . However, as the error observed in the two networks is next, determining which network is best for classification stage is not a trivial task. Case the need has a sample more controlled and with larger information NeuroSNP2.A is better, and in the case of a smaller sample that keep the amount of information present at NeuroSNP2.B appears more promising.

4.3 Results Obtained by the Third Model

The third model, was trained with a base mounted on the rules presented in section 3.5.1. The table 4 shows the comparative results between the neural networks in relation to SNPfilter. After training, the following error values were obtained: 0.002003 for NeuroSNP3.A and 0.002167 for a NeuroSNP3.B. As you can see, again the two values are very close.

When analyzing the table 4 notices a very behavior close between this model and the first, regarding classification ability of SNPs. Both models are little informative as indicated by the value of OR that oscillates with the increase in sample size. It is important to note that despite having a higher OR SNPfilter that the restriction on the HIGH the NeuroSNP3.A (6.9419), and in the restriction MEDIUM the NeuroSNP3.B (5.8454), the values of the ORs do not have a pattern, showing that the network is not being efficient in the classification process. However, it has behavior similar to the second model in relation to variation in the size of the sample. The CIs obtained in the networks of the third model, as in previous models, were small, showing that the calculated ORs are extremely accurate.

Table 4: Comparison between SNPfilter and networks of the Third Model.

	SNPs	alignments	OR	CI
MAQ	6,599,143	2,162,709	-	-
SNPfilter	2,174,341 (32.95%)	1,573,706 (72.77%)	5.3746	5.3565 - 5.3929
		NeuroSNP3.A		
HIGH	545,227 (08.26%)	420,862 (19.46%)	6.9419	6.8967 - 6.9874
MEDIUM	952,373 (14.43%)	660,245 (30.53%)	4.6363	4.6148 - 4.6579
LOW	2,740,161 (41.52%)	1,715,832 (79.34%)	3.4361	3.4261 - 3.4463
		NeuroSNP3.B		
HIGH	75,242 (01.14%)	46,722 (02.16%)	3.3605	3.3111 - 3.4107
MEDIUM	680,492 (10.31%)	503,720 (23.29%)	5.8454	5.8124 - 5.8785
LOW	1,938,537 (29.38%)	1,362,622 (63.01%)	4.8535	4.8366 - 4.8704

Either way, the supervised classification proved to be a useful tool in the complex task of detecting SNPs. Expectations regarding the universalization of its use in different genomes ie, for which the neural network was not specifically trained, will be evaluated in the experiments following.

5. Conclusion

Computational experiments clearly indicated the potential of the presented learning tool for the detection of SNPs. Their use alone or in conjunction with traditional filters is presented as an alternative for robust determination of SNPs in different genomes. The use of OR showed that the application of the filter increases the chance of finding a positive alignment of SNPs within the sample, and the expectation that this increase reflects the reduction of false positives.

Logically, the construction of training base can be improved mainly in two directions: by defining more specific rules for determining priority of false positives, and the use of SNPs biologically proven for building class of true positives. In any case, the supervised classification proved to be a useful tool in the complex task of detecting SNPs.

This work was presented and developed a computational strategy based on computational intelligence and machine learning, with ability to filter SNPs from whole genomic DNA (NeuroSNP). In the construction process NeuroSNP, Three different models did analyzed each one compared with the reference filter MAQ software, namely SNPfilter. In the genomes evaluated, the NeuroSNP managed to similar or superior results to the MAQ filter.

Acknowledgement

The authors thanks to reviewers who gave useful comments, and would like to express thanks to the National

Research Center of Dairy Cattle (Embrapa Dairy Cattle) of Brazilian Agricultural Research Corporation (Embrapa) for providing database and the provision of the necessary infraestructure to conduct this work; to the Graduate Program in Computational Modeling of Federal University of Juiz de Fora (UFJF) for the academic support; and to the Coordination for the improvement of Higher Level Personnel (CAPES) and the State of Minas Gerais Research Support Agency (FAPEMIG) for the financial support for the accomplishment of this paper.

References

[1] Basheer, I., Hajmeer, M., 2000. Artificial neural networks: fundamentals, computing, design, and application. Journal of Microbiological Methods 43 (1), 3 – 31, neural Computting in Micrbiology.

[2] Bland, J. M., Altman, D. G., 2000. The odds ratio. British Medical Journal 320 (7247), 1468.

[3] Bridges, M., Heron, E. A., OĎushlaine, C., Segurado, R., Morris, D., Corvin, A., Gill, M., Pinto, C., (ISC), T. I. S. C., 05 2011. Genetic classification of populations using supervised learning. PLoS ONE 6 (5), e14802.

[4] Chen, F., Dong, M., Ge, M., Zhu, L., Ren, L., Liu, G., Mu, R., 2013. The history and advances of reversible terminators used in new generations of sequencing technology. Genomics, Proteomics & Bioinformatics 11 (1), 34 – 40.

[5] Consortium, I. H., 2003. The international hapmap project. Nature 426 (6968), 789 – 96.

[6] Consortium, I. H. G. S., 2001. Initial sequencing and analysis of the human genome. Nature 409 (6822), 860–921.

[7] Curtis, D., 2007. Comparison of artificial neural network analysis with other multimarker methods for detecting genetic association. BMC Genetics 8 (1), 49.

[8] Eck, S., Benet-Pages, A., Flisikowski, K., Meitinger, T., Fries, R., Strom, T., 2009. Whole genome sequencing of a single bos taurus animal for single nucleotide polymorphism discovery. Genome Biology 10 (8), R82.

[9] GLENN, T. C., 2011. Field guide to next-generation dna sequencers. Molecular Ecology Resources 11 (5), 759–769.

[10] Gupta, P. K., 2008. Single-molecule DNA sequencing technologies for future genomics research. Trends in Biotechnology 26 (11), 602 – 611.

[11] Heidema, A. G., Boer, J., Nagelkerke, N., Mariman, E., van der A, D., Feskens, E., 2006. The challenge for genetic epidemiologists: how to analyze large numbers of snps in relation to complex diseases. BMC Genetics 7 (1), 23.

[12] HGSC, B. C. o. M., 2007. Ucsc, genome bioinformatics.

[13] Li, H., Ruan, J., Durbin, R., 2008. Mapping short dna sequencing reads and calling variants using mapping quality scores. Genome Research 18 (11), 1851–1858.

[14] Liu, Q., Guo, Y., Li, J., Long, J., Zhang, B., Shyr, Y., 2012. Steps to ensure accuracy in genotype and snp calling from illumina sequencing data. BMC Genomics 13 (Suppl 8), S8.

[15] Long, N., Gianola, D., Rosa, G., Weigel, K., Avendano, S., 2009. Comparison of classification methods for detecting associations between snps and chick mortality. Genetics Selection Evolution 41 (1), 18.

[16] Malhis, N., Jones, S. J. M., 2010. High quality snp calling using illumina data at shallow coverage. Bioinformatics 26 (8), 1029–1035.

[17] Mardis, E. R., 2008. The impact of next-generation sequencing technology on genetics. Trends in Genetics 24 (3), 133 – 141.

[18] Maxam, A. M., Gilbert, W., 1977. A new method for sequencing dna. Proceedings of the National Academy of Sciences of the United States of America 74 (2), 560–4.

[19] Morozova, O., Marra, M. A., 2008. Applications of next-generation sequencing technologies in functional genomics. Genomics 92 (5), 255–264.

[20] Nissen, S., 2005. Neural networks made simple. Software 2.0 magazine (2), 14–19.

[21] Ossowski, S., Schneeberger, K., Clark, R. M., Lanz, C., Warthmann, N., Weigel, D., 2008. Sequencing of natural strains of arabidopsis thaliana with short reads. Genome Research 18 (12), 2024–2033.

[22] Ren, L., Wang, W.-P., Gao, Y.-Z., Yu, X.-W., Xie, H.-P., 2009. Typing snp based on the near-infrared spectroscopy and artificial neural network. Spectrochimica Acta Part A: Molecular and Biomolecular Spectroscopy 73 (1), 106 – 111.

[23] Riedmiller, M., Braun, H., 1993. A direct adaptive method for faster backpropagation learning: the rprop algorithm. IEEE International Conference on Neural Networks 1 (3), 586–591.

[24] Sanger, F., Nicklen, S., Coulson, A. R., 1977. Dna sequencing with chain-terminating inhibitors. Proceedings of the National Academy of Sciences of the United States of America 74 (12), 5463–5467.

[25] Service, R. F., 2006. The race for the $1000 genome. Science 311 (5767), 1544–1546.

[26] Tomita, Y., Tomida, S., Hasegawa, Y., Suzuki, Y., Shirakawa, T., Kobayashi, T., Honda, H., 2004. Artificial neural network approach for selection of susceptible single nucleotide polymorphisms and construction of prediction model on childhood allergic asthma. BMC Bioinformatics 5 (1), 120.

[27] Zhang, J., Chiodini, R., Badr, A., Zhang, G., 2011. The impact of next-generation sequencing on genomics. Journal of Genetics and Genomics 38 (3), 95–109.