26th Euromicro International Conference on Parallel, Distributed, and Network-Based Processing

Combining Parallel Genetic Algorithms and Machine Learning to improve the research of optimal vaccination protocols

Marzio Pennisi
dept. of Mathematics and Computer Science
University of Catania
Catania, Italy
mpennisi@dmi.unict.it

Giulia Russo
dept. of Biomedical Sciences
University of Catania
Catania, Italy
giulia.russo@unict.it

Francesco Pappalardo
dept. of Drug Sciences
University of Catania
Catania, Italy
francesco.pappalardo@unict.it

***Abstract*—The developing of novel prophylactic and therapeutic vaccine candidates in the field of cancer immunology brought to very promising results against tumors, entitling full protection with reduced amount of the typical side effects of the actual conventional treatments. However, such treatments required a constant, life-long, administration procedure to keep protection. As both the period of protection and the relative number of administrations grow, the problem of finding the best administration protocol, in time and dosage, becomes more and more complex. Such a problem cannot be usually solved in *in vivo* experiments, as the costs in terms of time, money, and people would be prohibitive. We propose a hybrid approach that integrates machine learning and parallel genetic algorithms to enhance the research *in silico* of optimal administration protocols for a cancer vaccine. A neural network is used to improve both crossover and mutation operators. Preliminary results suggest that the use of such could bring to better administration protocols using a similar computational effort.**

***Index Terms*—Genetic algorithms, machine learning, neural networks, agent based models, optimal vaccine protocol, cancer**

I. Introduction

The goal of any administration protocol that relates to a given medical compound is to maximize the beneficial effects of the compound, with the minimal required dosage. In order to reach this goal, the dosage should be distributed in time appropriately. In some cases a single administration is sufficient. In others, multiple administrations are required to obtain the desired effects. Vaccines belong to this second class, and usually require two or more administrations to train our immune system against pathogens.

Vaccines consist of biological preparations that stimulate the body's immune system immunity towards a specific disease [7]. This is possible due to an agent included into the vaccine formulation that looks like the microorganism that causes the disease and that is then able to instruct the immune system to recognize, kill and remember the pathogen, especially for future encounters and infections. Vaccines are made of killed or weakened forms of the pathogen, and are usually coupled with specific components like "adjuvants" in order to achieve a prophylactic effect and to obtain an increase of the immune response. There are essentially three types of vaccines: i) vaccines composed by killed pathogens, ii) vaccines composed by modified or attenuated live pathogens and iii) vaccines composed by fragments of the pathogen [11].

A critical aspect that it is worth mentioning is related to the safety and effectiveness of vaccine itself that should be guaranteed by the vaccination protocol. Typical vaccination schedules are usually composed by more than a vaccine administration. However, the design of the schedule it usually done on the basis of theoretical considerations, clinical trials and practical medical experience that have previously shown the safety and the efficacy at the individual level. Nevertheless, the opportunity to obtain the best administration schedule and the maximum health benefit for a specific vaccine in terms of time and dosage still remains an outstanding unsolved problem. Indeed, the research process would spend too many resources and time to optimize the schedule of administration for each vaccine.

To this aim, *in silico* computational models [3] could help to a refinement of vaccine administration protocol [21], a re-testing of hypotheses and hence, an optimization of the schedule itself through the employment of several optimal research strategies like Genetic Algorithms (GA) [25] and simulated annealing (SA) [23], [24].

However, since these computational models need to be executed thousands of times inside the optimization research strategies on different virtual individuals to evaluate the efficacy of a protocol, the use of parallel computing has been taken into account in order to speed-up the research procedure. Even in this scenario, the computational effort remains high, and novel strategies that may reduce the execution time, or at least, provide better results in the same time-window are welcome.

We propose a preliminary hybrid approach that integrates machine learning and parallel genetic algorithms to improve the research *in silico* of optimal administration protocols for a cancer vaccine. Neural networks (NN) are used to improve both crossover and mutation operators. First results suggest their adoption could lead to better administration protocols in the same computational time. Such approaches can lead to

0-7695-6380-5/18/$31.00 ©2018 IEEE
DOI 10.1109/PDP2018.2018.00070

a new evaluation of possible novel candidate administration protocols that could maximize efficacy and reduce the risk of possible adverse reactions.

II. Background

A. Application of Agent Based Models to predict Cancer

Without any presumption to have a comprehensive overview of tumors and tumor immunology, here we would like to highlight that cancer is elicited by a combination of exogenous and endogenous factors. Tumor immunology is a hot topic that deals with the exploration of these factors and the related effects on the immune system with the aim to cure (immunotherapy) or prevent (immunoprevention) human tumors.

Vaccine development is a complex and relatively verbose process. It involves the testing of ideas in *in vitro* and *in vivo* experiments experiments in order to maximize effectiveness and efficacy, without forgetting safety. The cost and administration issues of vaccinations also need to be considered for the vaccine development. The Triplex vaccine [8], [16] was designed to improve the efficacy of existing immunopreventive treatments against mammary carcinoma. A vaccination schedule is effective if it prevents the formation of carcinoma in situ (CIS) in the host in the period under consideration. Triplex has been proved effective in preventing CIS when it is administered with a chronic schedule. *In vivo* experiments show that the immune response to Triplex vaccination follows the law of biological variability.

SimTriplex [19] is a computational model able to simulate the competition among the Triplex immunopreventive vaccine, the mammary carcinoma and the immune system interactions. It mimics at the cellular level the behavior of immune cells of vaccinated as well as naive mice. The *in silico* experiments have shown excellent agreements with *in vivo* experiments on HER-2/neu mice for the Triplex vaccination schedules.

To obtain the maximum efficacy from a vaccination strategy, it is necessary to identify an optimal schedule i.e., one that maintains its efficacy with a minimum number of vaccine administrations. Moreover, like in standard drug administration, the vaccine must be effective for a high percentage of the treated patients. At the time of writing, the state of the art in developing a vaccine schedule is based on a consensus technique that consists in a public statement that is agreed upon as the evidence-based knowledge by a representative group of experts in that area.

B. The SimTriplex Model

The SimTriplex model is a specialized multi agent based model developed to study and predict the effects of the Triplex vaccine on the immune system - mammary carcinoma competition. SimTriplex represents, at the cellular scale, both the immune system cells and molecules stimulated by the Triplex vaccine to fight against the tumor.

The simulator uses a 16x16 lattice with hexagonal geometry to simulate a small fraction of a mammary gland (1mm3). The time is discrete and divided in time-steps of 8 hours for a total simulation time of 1200 time-steps. On each lattice site, at each time-step, all entities that are in the same site can stochastically interact each others according to pre-defined biologically driven rules. Any successful interaction causes the state-change of the involved entities (i.e., entity maturation, death, differentiation, duplication, etc.). Then, all the entities can move from the lattice site where they are in another one in the neighborhood.

The following cellular and molecular entities have been taken into account: i) the cancer cells (CC) that encode their tumor associated antigens (TAA); they interact with antibodies (Ab), T-cells (TC) and natural killer cells (NK), ii) the vaccine cells (VC) that include TAAs, IL-12 and allogenic major histocompatibility molecules class I (MHCI); VC interact like CC, with Ab, TC and NK, but the affinity function is modified by the presence of two adjuvants, iii) NK cells which, in the model, express only CD16a receptor for ADCC mediated cytotoxicity; they interact with Ab coated VC and CC.

One of the most important features of the SimTriplex model is represented by the use of binary strings to mimic cellular receptors. The occurring of cellular interactions is driven by receptor binding and affinity, and this is simulated through the use of bit-strings that represent entity receptors and the hamming distance (number of mismatching bits) between them to measure the receptors affinity. While this is a rough approximation of the real biological scenario, millions of interactions can be simulated easily with a very small computational effort. The results obtained by the SimTriplex model have shown an excellent agreement with the in vivo experiments both for the time of formation of solid tumor and for the role of antibody responses in controlling tumor growth [18].

C. Genetic Algorithms

Genetic algorithms (GAs), invented by Holland in 1960, are a particular class of evolutionary algorithms [5] that use techniques inspired by evolutionary biology. Common applications of GAs are discrete optimization problems of multivariate functions, $f(\mathbf{x})$, where $\mathbf{x}$ is the vector of variables $x_1, x_2, \ldots, x_n$. A GA is a method for moving from one set $\{\mathbf{x}^k\}$ to a new one by using a kind of natural selection together with the genetic inspired operators of crossover, mutation and so on.

In the standard GA language the set $\{\mathbf{x}^k, k = 1, \ldots, m\}$ is referred as a population and each x_i as a chromosome. A chromosome x_i is an n-dimensional string, usually a binary string; each element of this string is referred as a gene; each gene can have different instances, i.e. allele.

GAs are widely used in different areas of bioinformatics like: analysis of gene expression data [17]; cancer gene search [27]; biological pathway modeling [2] (for a review, see [10] and [14]).

Specifically, in immunoinformatics, a common optimization problem is the search of optimal vaccination schedules. Searching for an optimal dosage and timing schedule is notably important in cancer immunopreventive approaches because one has the requirement to maintain a high level of

protective immunity for the entire lifetime of the host. From this consideration, it clearly appears that the use of a GA to search for optimal vaccination strategies could be of great usefulness. The main issue is to define, in mathematical terms, the function that have to be optimized by the GA.

D. The optimization problem

The problem of study and optimize vaccination protocols using stochastic optimization techniques and/or machine learning methods is quite new. In [26] Sanders showed the application of GA to find optimal vaccination of a general population network, extending the work from Patel et al. [22]. Choi et al. presented a framework that integrates antibody feature and function data with machine learning to provide an approach for discovering and assessing multivariate immune correlates [4]. We already described in mathematical terms the problem to optimize the administration schedule of a cancer vaccine in [25], and we reassume here the presented results.

The first thing one has to face with in the process of translating the biological optimization problem into mathematical therms is represented by the concept of vaccine effectiveness. For this purpose, we consider a time interval $[0, T]$ in which it is studied the action of the vaccine on a set of virtual mice S. This can be either the lifetime of the mouse or the time length of the *in vivo* experiment. Time slice is then divided into $N-1$ equally spaced subintervals of width Δt, i.e. $\{t_1 = 0,\ t_2,\ \ldots, t_i,\ \ldots,\ t_N = T\}$. The time portion Δt represents the possible vaccine administrations, e.g. every 8 hours.

Let $\mathbf{x} = \{x_1, x_2, \ldots, x_i, \ldots x_N\}$ be a binary vector that matches the sequence of vaccine schedule. $x_i = 0/1$ stands for administration/no administration of the vaccine. i is the time t_i of the vaccine administration and t_N the end of the vaccination schedule. The number of vaccine administrations is given by $n = \sum_{i=1}^{N} x_i$.

Let $\tau(\mathbf{x}, \lambda_j)$ be the time of CIS appearance. This is a function of the vaccination schedule $\mathbf{x}$ administered to the mouse $j \in S$ and a parameter λ_j which describes the biological diversity. As an example, in the SimTriplex model, λ_j is represented by the random seed that determines the initial immunological repertoire. The vaccine will be effective if $\tau \geq T$. We then can formulate the optimization problem in an unconstrained form as the finding of $\overline{\mathbf{x}}$ such that:

$$\tau(\overline{\mathbf{x}}, \lambda_j) = max\{\tau(\mathbf{x}, \lambda_j)\} \quad (1)$$

$$(\overline{\mathbf{x}}) = min\{n(\mathbf{x})\} \quad (2)$$

This clearly represents a multi-objective discrete optimization problem. Looking at the search space width, D, one can notice that the cardinality is equal to 2^N. For $T = 360$ days, and $\Delta t = 24$ hours the cardinality is $2^{360} \sim 10^{108}$. This enormous number does not allow any computational form of exhaustive search.

Generally speaking when facing to a classical optimization problem, the fitness function can be expressed as a mathematical formula, independently from its complexity. In our case, the function τ that measures the action of the stimulated immune system, cannot be written in any analytical form. Moreover, $\tau(\mathbf{x}, \lambda_j)$ can only be computed through an ad hoc built in simulator. In our case we used SimTriplex simulator.

To write the problem in mathematical terms, we have:

$$f_j(n(\mathbf{x}), \tau(\mathbf{x}, \lambda_j)) = \frac{n^2(\mathbf{x})}{\tau(\mathbf{x}, \lambda_j)} \quad (3)$$

and problem (1) and (2) reduce to: find $\overline{\mathbf{x}}$ such that

$$f_j(n(\overline{\mathbf{x}}), \tau(\overline{\mathbf{x}}, \lambda_j)) = min\ \{f_j(n(\mathbf{x}), \tau(\mathbf{x}, \lambda_j))\} \quad (4)$$

Finding a suitable fitness function to solve this problem is one of the important step that must be accomplished to apply a genetic algorithm. This problem has been solved in [20]. However, two important issues that affected that solution are represented by *i)* the optimal schedule $\overline{x}$ showed very high peaks in cancer cells, and *ii)* the schedule found was not able to protect a statistically significant percentage of mice.

To overcome these issues we introduced some constraints on the maximum allowed number of cancer cells and reformulated the optimization problem as follows. Let $\{j_1, j_2, \ldots, j_m\} \subset S$, with $m = 8$, a random chosen subset of *in silico* mice. The optimization problem is to find $\overline{\mathbf{x}}$ such that:

$$\begin{cases} f_{j_1}(n(\overline{\mathbf{x}}), \tau(\overline{\mathbf{x}}, \lambda_{j_1})) = min\ \{f_{j_1}(n(\mathbf{x}), \tau(\mathbf{x}, \lambda_{j_1}))\} \\ f_{j_2}(n(\overline{\mathbf{x}}), \tau(\overline{\mathbf{x}}, \lambda_{j_2})) = min\ \{f_{j_2}(n(\mathbf{x}), \tau(\mathbf{x}, \lambda_{j_2}))\} \\ \vdots \\ f_{j_8}(n(\overline{\mathbf{x}}), \tau(\overline{\mathbf{x}}, \lambda_{j_8})) = min\ \{f_{j_8}(n(\mathbf{x}), \tau(\mathbf{x}, \lambda_{j_8}))\} \\ \text{subject to:} \\ M_1(\overline{\mathbf{x}}) \leq \gamma_1,\ t \in [0, T_{in}] \\ M_2(\overline{\mathbf{x}}) \leq \gamma_2,\ t \in [T_{in}, T] \end{cases} \quad (5)$$

This is again a multi-objective optimization problem. We can reduce it to a standard problem defining the new objective function as a linear combination of $f_{j_1}, \ldots, f_{j_8}$, i.e.

$$f(n, \tau_{j_1}, \tau_{j_2}, \ldots, \tau_{j_8}) = \sum_{k=1}^{8} \alpha_k\ f_{j_k}(n, \tau_{j_k}) \quad (6)$$

Where we set all $\alpha_k = 1$ as the mice are equiweighted.

Finally, we should also take into account wet-lab dependent issue. This means that we cannot administrate the vaccine formulation at arbitrary time, but twice a week, at most (i.e., Monday and Thursday). The positive side is that in this way we can drastically reduce the search space to approximately 115 days in which the vaccine can be administrated. This problem has been tackled with a genetic algorithm that uses tournament selection, uniform crossover, and classical mutation operators on chromosomes composed by binary strings of 115 bits. The GA fitness function (6) is evaluated using the SimTriplex model.

E. Neural Networks

Artificial Neural Networks or, more simply, Neural Networks (NNs) are biologically-ispired computing paradigms based on network structures that take inspiration from the brain. Neural networks are designed to progressively learn by examples, usually without any "a priori" task-specific knowledge, determining and improving their own set of relevant characteristics from the training material.

A NN is based on a set of nodes (artificial neurons) connected by a series of links (analogous to synapses) used by neurons to transmit a signal from one to another. The artificial neuron that receives the signal can process it and then signal artificial neurons connected to it.

Commonly, the signal transmitted between two neurons is a real number, as the output of each neuron is calculated by a non-linear function of its inputs. Neurons and connections typically have a weight that is adjusted (increased or decreased) during the learning process. Neurons may have thresholds that impose that the output signal is sent only if the aggregate input signal crosses the threshold. Usually, neurons are organized in layers, and signals travel from the first (input), to the last (output) layer, after passing through the various layers of the network.

The initial idea of the NN approach was to deal with problems in a similar way as the human brain does. NNs are nowadays applied in many scientific fields, from computer vision to speech recognition, from video games to medical diagnosis and prediction. Interested readers can find further informations about NN design in the book by Hagan et al [12].

III. Integrating GA with neural networks

GA can be easily used on a broad range of problems to optimize a given fitness-function towards a proposed goal. A possible issue that may arise when using GA is connected to the intrinsic nature of the fitness function itself. As GA require many evaluations of the fitness function, its running cost may represent a limiting factor. To this end, it is important to find a way to reduce the total computational effort or, at least, to drive the research towards better candidate solutions with a similar effort, by finding ways to approximate the fitness calculus. Fitness approximation using NN is a very hot topic [1]. For example, NN have been applied combined to GA to improve the solution of the Ms Pacman game [15], or as surrogate models to optimize radiotherapy treatment [9].

In this scenario, the use of use neural-network (NN) based predictors to quickly estimate the fitness of individuals for choosing the best one has been taken under examination. In this preliminary analysis, we introduced the use of NN inside the mutation and crossover operators in order to empower the exploring of the search space and to drive the research towards better solutions. The preliminary approach we present here is very simple. When we apply the mutation operator, we produce two random mutations of the same chromosome instead of one, and we then use the NN to predict the one with the best fitness. This chromosome will be chosen. A similar approach is using during the application of the crossover operator. When two chromosomes are chosen for crossover, the operator is applied two times to obtain two possible random offsprings couples, and the couple that contains the offspring with the best predicted NN fitness value is chosen.

To implement the NN we used the Matlab routines with a training set of 3200 elements. In order to translate any candidate protocol (composed by 115 bits) in a suitable input for the Matlab NN tool, we divided any input element (i.e., the binary string representing a vaccination protocol) in a set of six integer inputs, each one taking into account 19 bits. In this way, we decomposed the research space into a set of six periods, each one representing a time-window of approximately two months. It must be noted that in this way we considered 114 bits, neglecting the last one. However, this does not represent a problem since, from a biological point of view, an injection made in the last day of an experiment is of little use.

The output of each protocol is obviously represented by its fitness value, computed using the SimTriplex model on a set of randomly chosen virtual mice. The training set has been generated from data coming from the first generations of the GA itself, and this process can be easily automated and made just in time as the execution of the GA starts. From preliminary experiments, we found that a higher number of training elements (i.e. > 3200) that came from the GA execution did not improve the training procedure. This is probably due to the fact that the GA starts to converge towards an optimal solution and then most chromosomes start to become very similar, producing redundant results that do not improve the training of the net. The training-validation-testing sub-groups have been randomly obtained from the total training data using 70% - 15% - 15% ratios.

The number of neurons has been obtained using the following rule of thumb:

$$N_h = \frac{N_s}{(\alpha \cdot (N_i + N_o))}$$

where $N_i = 6$ is the number of input neurons, $N_o = 1$ is number of output neurons, $N_s = 3200$ number of samples in training data set, and $\alpha = 6$ is an arbitrary scaling factor to be chosen usually in the range 2-10. After many experiments we obtained from the standard NN Matlab procedure a two-layer feed-forward network with sigmoid hidden neurons and linear output neurons, trained with Levenberg-Marquardt back propagation algorithm with a total $R = 0.54211$ (figure 1). We know that such a value, measuring the correlation between outputs and target, is not very high. As a consequence of that, we decided to use in this first implementation the net as described i.e., only to compare two possible scenarios in order to choose the best one. It is worth nothing here that, whilst the rising of α and thus, the lowering of the number of neurons, led to unreliable results, its lowering did not allow to increase the correlation between the outputs and target.

As already stated, the GA must evaluate the efficacy of a protocol over a set of virtual individuals. We used the same

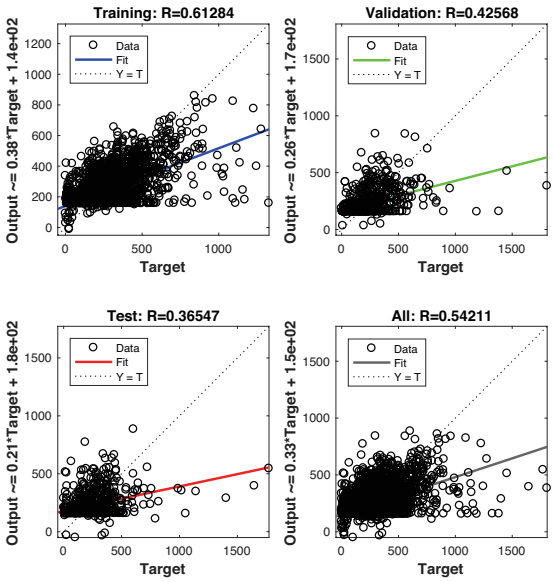

Fig. 1. Matlab Neural Network regression plots on the training, validation, test and total data.

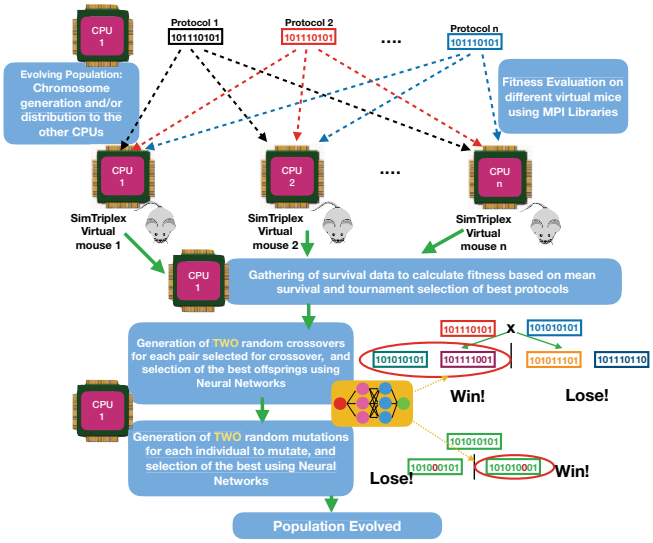

Fig. 2. Graphical representation of the procedure followed to evolve the population. CPU 1 creates the initial population or distributes the existing population to the other CPUs. Then, all protocols are evaluated in parallel using MPI on n=8 different virtual mice using different instances of SimTriplex model. Each virtual mouse is identified by a different random seed. After evaluation, CPU 1 collects survival information from the other CPUs and proceeds to calculate the fitness values using mean survival, then executes tournament selection. Chromosomes selected for both crossover and mutation will undergo towards the respective procedures. During crossover, two random crossovers are generated and the couple of offsprings that contains the best estimated protocol, according to the NN, will be part of the new population. Similarly, two random mutations for each chromosome selected for mutation are produced and the best mutated chromosome, according to the NN prediction, is selected.

number of virtual mice ($m = 8$) that we used in previous experiments, as also previously described in section II-D. In our past experience we found that $m = 8$ was a good trade-off between the need to keep the research computationally feasible, and the necessity to have a minimal representative sample of the virtual mice population. Increasing m will probably improve the research but will require much more time.

It should be noted that the use of a broader population for the training set would probably reduce the variability over the fitness function and would globally improve the NN results, as the effects of possible virtual "outliers" that contribute to increase the mean error would be dampened. As a consequence of that, further experiments that involve the use of different tools, and strategies have already been started and will be presented in due course.

The parallel GA has been implemented using ANSI C and MPI libraries in order to evaluate the efficacy of candidate protocols on different individuals at the same time. Each protocol (represented by a chromosome) is evaluated on $m = 8$ randomly chosen virtual mice with the SimTriplex ABM using different MPI threads. The simulation results are thus collected from the master thread that calculates the fitness values using this data, evolves the population by applying tournament selection, crossover and mutation operators, and then redistributes the newly evolved chromosomes population to the other threads for further evaluation. The Matlab routines that manage the trained NN have then integrated inside the GA algorithm using the Matlab mex engine and compiled with the mex compiler to obtain an integrated C-MPI-Matlab shell executable. In figure 2 we show the steps followed by the GA procedure for evolving the population at each generation.

IV. Results & Discussion

To compare the GA results with and without the presence of the NN optimization, we ran the algorithm 4 times for each scenario using 4 different random seeds, for a total of 8 simulations. Each simulation took into account a population of 40 chromosomes, and each chromosome was evaluated on a set of 8 virtual mice with the SimTriplex model for every generation. The number of generations has been set to 100. This means that a complete instance of the GA procedure requires 32000 calls to the SimTriplex model. Each execution of the GA algorithm required on average a total time that varies from 15 to 25 hours on a 4-core (8 threads) i7 CPU using 8 MPI threads. We found that the evaluation of the NN had a minor impact on the total execution time, and temporal variations among different simulations (both in presence or in absence of the optimization) were more likely due to the time required to execute the SimTriplex simulator. This time is highly variable and depends on the random virtual mouse and on the protocol to be tested. Usually, good chromosomes that lead to total survival may require more time to be evaluated in respect to worse chromosomes that lead to premature death of the virtual mouse, but this is not always true. Indeed, the mean execution time for the GA

procedure without the NN optimization was (mean $\pm$ s.d.) $\approx 25,526 \pm 8,54$ hours, whereas the use of the NN optimization lead to $\approx 21,647 \pm 3,67$ hours.

As the time required to calculate the fitness function may vary consistently, we analyzed the temporal evolution of two scenarios by considering the number of generations rather than the real execution time, as this first value exactly describes the number of calls to the SimTriplex model. The temporal evolution of the fitness function (mean and standard deviation) for the two scenarios is presented in figure 3.

From the figure one can envisage that just after generation 30 the two mean behaviors start to follow different paths, with the NN enabled GA leading to lower (better) fitness values. This difference becomes more evident after generation 50, where the two curves present a small, but significative difference. Statistical difference between the two curves has been established by a two-tailed Wilcoxon procedure ($p = 0.0020$), confirming as the application of the NN predictor leads to a different mean behavior.

As the correlation between the NN outputs and the target values obtained during the training phase was not very high, we would have expected that the effects of the NN were more pronounced at the beginning of the GA execution, since at that time the candidate solutions are more heterogeneous, and thus it would have been easier to discern among the coarse differences among good and bad vaccination schedules. Indeed, the contribution of the NN is starting stronger from generation 30, when the chromosomes population is already converging towards the near optimal solution. This suggests that the NN is able to discern even fine rather than coarse differences in protocols' efficacy.

The mean fitness values obtained at the end of the GA procedure for the two samples are (mean $\pm$ SD) $9,303 \pm 1,77825$ and $6,64955 \pm 0,84928$ without and with the NN optimization, respectively. Even with a so low number of experiments, such final values seem to be statistically different ($p = 0,0359$) according to a two-tailed t test.

V. Conclusion

The use of computational models empowered the possibility to increase our knowledge of the real world. Models can be used to test and validate hypotheses, and to suggest in much less time and with a reduced cost better strategies and procedures. In our past experience, with the use of the SimTriplex agent based model and genetic algorithms we were able to suggest an optimal vaccination schedule for a vaccine against breast cancer. However, such a research was computationally expensive due to the time required by the SimTriplex model. To this end we started to investigate novel strategies and algorithms to improve our approach. Presented results confirm that the NN is potentially able to improve the GA results with a similar computational effort. However, some points should be better investigated. For example, strategies for achieving a better training of the net should be examined and are now under investigation. These strategies could include a higher number of virtual mice to damper the mean error as well as a better design of the training set with data not only coming from the GA algorithm itself. Furthermore, the possibility to "statistically substitute" the SimTriplex model with NN, using statistical techniques like bayesian modeling, is under evaluation and results will be presented in due course.

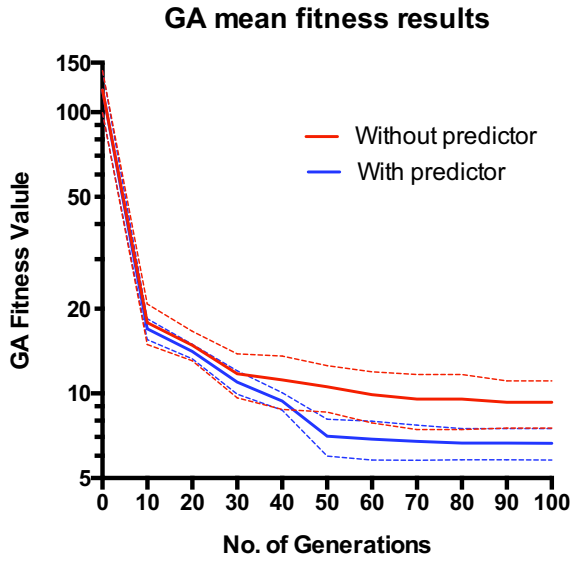

Fig. 3. GA Mean (solid lines) and SD (dashed lines) fitness time evolution with and without the use of the NN predictor (blue and red lines, respectively). Fitness values (y axis) are represented on a $\log_{10}$ scale to improve readability.

References

[1] M. Bhaacharya, Evolutionary Approaches to Expensive Optimisation. *Arxiv - Computers & Society*, **2(3)** (2013), 53–59. DOI:h p://dx.doi.org/10. 14569/IJARAI.2013.020308.

[2] W.J. Bosl, Systems biology by the rules: hybrid intelligent systems for pathway modeling and discovery, *BMC System Biology*, **1(13)** (2007).

[3] F. Castiglione, F. Pappalardo, C. Bianca, G. Russo, S. Motta, Modeling biology spanning different scales: An open challenge, *BioMed Research International*, **2014(2014)**, (2014), Article number 902545.

[4] I. Choi, A.W. Chung, T.J. Suscovich, S. Rerks-Ngarm, P. Pitisuttithum, S. Nitayaphan, J. Kaewkungwal, R.J. O'Connell, D. Francis, M.L. Robb, N.L. Michael, J.H. Kim, G. Alter, M.E. Ackerman, C. Bailey-Kellogg, Machine learning methods enable predictive modeling of antibody feature:function relationships in RV144 vaccines, *PLoS Computational Biology*, **11(4)**, (2015), e1004185.

[5] D. Corne, M. Dorigo, F. Glover, New Ideas in Optimization, Advanced Topics in Computer Science, *McGraw-Hill*, (1999).

[6] S. Croci, G. Nicoletti, L. Landuzzi, C. De Giovanni, A. Astolfi, C. Marini, E. Di Carlo, P. Musiani, G. Forni, P. Nanni, P.-L. Lollini, Immunological Prevention of a Multigene Cancer Syndrome, *Cancer Research*, **64**, (2004) 8428–8434.

[7] D.D. Chaplin. Overview of the Immune Response, *The Journal of allergy and clinical immunology*, **125(2)**, (2010).

[8] C. De Giovanni, G. Nicoletti, L. Landuzzi, A. Astolfi, S. Croci, A. Comes, S. Ferrini, R. Meazza, M. Iezzi, E. Di Carlo, P. Musiani, F. Cavallo, P. Nanni, P.-L. Lollini, Immunoprevention of HER-2/neu transgenic mammary carcinoma through an interleukin 12-engineered allogeneic cell vaccine, *Cancer Res.*, **64(11)**, (2004) 4001–4009.

[9] J. Dias, H. Rocha, B. Ferreira, M. do Carmo Lopes, A genetic algorithm with neural network fitness function evaluation for IMRT beam angle optimization. *Central European Journal of Operations Research* **22, 3(9)** (2014), 431–455. DOI:h p://dx.doi.org/10.1007/s10100-013-0289-4.

[10] Z. Ezziane, Applications of artificial intelligence in bioinformatics: A review, *Expert Systems With Applications*, **30:1**, (2006) 2–10.

[11] A. Gordon. Overview of vaccines and vaccination, *Mol. Biotechnol.*, **29(3)** (2005).
[12] M.T. Hagan, H.B. Demuth, M.H. Beale, O. De Jesús, Neural network design, II ed. 2014, Martin Hagan Ed. 800pp.
[13] P.-L. Lollini, S. Motta, F. Pappalardo, Discovery of cancer vaccination protocols with a genetic algorithm driving an agent based simulator, *BMC Bioinformatics*, **7:352** (2006) doi:10.1186/1471-2105-7-352.
[14] T. Manning, R.D. Sleator, P. Walsh: Naturally selecting solutions, *Bioengineered*, **4:5**, (2013) 266–278, DOI: 10.4161/bioe.23041
[15] M. Morosan, R. Poli, Speeding up Genetic Algorithm-based Game Balancing using Fitness Predictors, *GECCO 2017, the Genetic and Evolutionary Computation Conference Companion, July 15th-19th* (2017), DOI: 10.1145/3067695.3076011.
[16] P. Nanni, G. Nicoletti, C. De Giovanni, L. Landuzzi, E. Di Carlo, F. Cavallo, S.M. Pupa, I. Rossi, M.P. Colombo, C. Ricci, A. Astolfi, P. Musiani, G. Forni, P.-L. Lollini, Combined allogeneic tumor cell vaccination and systemic interleukin 12 prevents mammary carcinogenesis in HER-2/neu transgenic mice, *J. Exp. Med.*, **194(9)**, (2001) 1195–1205.
[17] C.H. Ooi, P. Tan, Genetic algorithms applied to multi-class prediction for the analysis of gene expression data, *Bioinformatics*, **19:1**, (2003), 37–44.
[18] A. Palladini, G. Nicoletti, F. Pappalardo, A. Murgo, V. Grosso, V. Stivani, M.L. Ianzano, A. Antognoli, S. Croci, L. Landuzzi, C. De Giovanni, P. Nanni, S. Motta, and P.-L. Lollini. In silico modeling and in vivo efficacy of cancer-preventive vaccinations. *Cancer Research*, **70(20)**, 7755–7763, 2010.
[19] F. Pappalardo, P.-L. Lollini, F. Castiglione, S. Motta, Modelling and Simulation of Cancer Immunoprevention vaccine, *Bioinformatics*, **21(12)**, (2005) 2891–2897.
[20] F. Pappalardo, E. Mastriani, P.-L. Lollini, S. Motta, Genetic Algorithm against Cancer, *Lecture Notes in Computer Science*, **3849**, (2006) 223–228.
[21] F. Pappalardo, S. Motta, P.L. Lollini, E. Mastriani, Analysis of vaccine's schedules using models, *Cellular Immunology*, **244(2)**, (2006) 137–140.
[22] R. Patel, I.M. Longini, and M.E. Halloran. Finding optimal vaccination strategies for pandemic influenza using genetic algorithms.*Journal of Theoretical Biology*, **234(2)**, (2005) 201–212.
[23] M. Pennisi, R. Catanuto, F. Pappalardo, S. Motta, Optimal vaccination schedules using simulated annealing, *Bioinformatics* , **24(15)**, (2008), 1740–1742, 10.1093/bioinformatics/btn260.
[24] M. Pennisi, R. Catanuto, F. Pappalardo, S. Motta, E. Mastriani, A. Cincotti, Simulated Annealing and Optimal Protocols, *Journal of Circuits, Systems and Computers*, **18(08)**, (2009), 1565–1579, 10.1142/S0218126609005770.
[25] M. Pennisi, F. Pappalardo, P. Zhang, S. Motta, Searching of optimal vaccination schedules, *IEEE Engineering in Medicine and Biology Magazine*, **28 (4)** (2009) 67–72, 10.1109/MEMB.2009.932919.
[26] L. Sanders, Optimal Vaccination of a General Population Network via Genetic Algorithms, BioRxiv preprint. https://doi.org/10.1101/227116.
[27] S. Shah, A. Kusiak, Cancer gene search with data-mining and genetic algorithms, *Computers in Biology and Medicine*, **37:2**, (2007) 251–261.