Nested Active Learning for Efficient Model Contextualization and Parameterization

Chase Cockrell, Jonathan Ozik, Nick Collier, and Gary An

Abstract: The description of the environment in which a biomedical simulation operates (model context) and parameterization of internal model rules (model content) requires the optimization of a large number of free-parameters; given the wide range of variable combinations, along with the intractability of ***ab initio*** modeling techniques which could be used to constrain these combinations, an astronomical number of simulations would be required to achieve this goal. In this work, we utilize a nested active-learning workflow to efficiently parameterize and contextualize an agent-based model of sepsis. **Methods:** Billions of microbial sepsis patients were simulated using a previously validated agent-based model (ABM) of sepsis, the Innate Immune Response Agent-Based Model (IIRABM). Contextual parameter space was examined using the following parameters: cardio-respiratory-metabolic resilience; two properties of microbial virulence, invasiveness and toxigenesis; and degree of contamination from the environment. The model's internal parameterization, which represents gene expression and associated cellular behaviors, was explored through the augmentation or inhibition of signaling pathways for 12 signaling mediators associated with inflammation and wound healing. We have implemented a nested active learning approach in which the clinically relevant model environment space for a given internal model parameterization is mapped using a small Artificial Neural Network (ANN). The outer AL level workflow is a larger ANN which uses a novel approach to active learning, Double Monte-Carlo Dropout Uncertainty (DMCDU), to efficiently regress the volume and centroid location of the CR space given by a single internal parameterization. **Results:** A brute-force exploration of the IIRABM's content and context would require approximately $3*10^{12}$ simulations, and the result would be a coarse representation of a continuous space. We have reduced the number of simulations required to efficiently map the clinically relevant parameter space of this model by approximately 99%. Additionally, we have shown that more complex models with a larger number of variables may expect further improvements in efficiency.

Introduction

Sepsis in an inflammatory condition with a mortality rate of between 28%-50%(1). Numerous mechanistic computational simulations of acute inflammation and sepsis have been utilized over the past two decades(2-9). These models have demonstrated that the sepsis population is much more heterogeneous than previously thought and this can be reflected by utilizing a range of multidimensional parameters that correlate to biologically plausible behaviors and phenotypes. Despite insights generated form these methods, there remain considerable challenges in the calibration and parameterization of the models. The description (contextualization) of the environment in which a biomedical simulation operates and parameterization of internal model rules (model content) requires the optimization of a large number of free-parameters; given the wide range of variable combinations, along with the intractability of ***ab initio*** modeling techniques which could be used to constrain these combinations, an astronomical number of simulations would be required to achieve this goal.

The problem of combinatorial complexity in the selection of model parameters is well-established in the computational/biological modeling communities (10-14). In previous work (2), we utilized high-performance computing to demonstrate the need for comprehensive "data coverage" among possible model states as well as the importance of internal parameter variation (as compared to model structure) to capture the full range of biological heterogeneity seen clinically. In order to render this task computationally tractable, we have employed a nested active learning approach in order to efficiently and comprehensively explore model parameter space.

IIRABM: The primary model analyzed in this work is the Innate Immune Response Agent Based Model (IIRABM) (3, 15). The IIRABM is an abstract representation/simulation of the human inflammatory signaling network response to injury; the model has been calibrated such that it reproduces the general clinical trajectories seen in sepsis. The IIRABM operates by simulating multiple cell types and their interactions, including endothelial cells, macrophages, neutrophils, TH0, TH1, and TH2 cells as well as their associated precursor cells. The simulated system dies when total damage (defined as aggregate endothelial cell damage) exceeds 80%; this threshold represents the ability of current medical technologies to keep patients alive (i.e., through organ support machines) in conditions that previously would have been lethal. The IIRABM is initiated using 5 external variables – initial injury size, microbial invasiveness, microbial toxigenesis, environmental toxicity, and host resilience

Active Learning: Active learning (AL) is a sub-field of machine learning (ML) which focusses on finding the optimal selection of training data to be used to train a ML or statistical model (16). AL can be used for classification (17, 18) or regression (19, 20). AL is considered to be an ideal technique for modeling problems in which there is a large amount of unlabeled data and manually labelling that data is expensive. In these circumstances (specifically the costly data labelling) AL provides to most generalizable and accurate model for the cheapest cost, which for the purposes of this work, is computation time.

EMEWS: Our ML models are trained and integrated using the Extreme-scale Model Exploration With Swift (EMEWS) framework (21-23). EMEWS enables the creation of high-performance computing (HPC) workflows for implementing large-scale model exploration studies. Built on the general-purpose parallel scripting language Swift/T (24), multi-language tasks can be combined and run on the largest open science HPC resources (25) via both data-flow semantics and stateful resident tasks. The ability that EMEWS provides for incorporating model exploration algorithms such as AL, implemented in R or Python, allows for the direct use of the many libraries relevant to ML that are being actively developed and implemented as free and open source software.

Methods

The lower-level AL procedure seeks to find the boundary of the parameter space deemed "clinically relevant" (2) as a function of four parameters which describe the context in which the IIRABM operates: two measures of microbial virulence (invasiveness and toxigenesis), host resilience, and environmental toxicity. In this scheme, there are two classes: clinically relevant and not clinically relevant. We assume that there exists some function,

$$y = f(\vec{x}), x \in \chi \subset \mathbb{R}^n, y \in \mathbb{R}$$

which accurately classifies model context parameters can be approximated given input data from the training set:

$$D_{train} = \{x_j^t, f(x_j^t)\}$$

for $j = 1, \ldots, n$. The NN model uses a binary cross-entropy (26) loss function, in which the loss is given by:

$$L = -\sum_{i=1}^{2} y_i \log(\hat{y}_i)$$

Where y_i is the ground truth value and $\hat{y}_i$ is the NN-approximated score. The AL algorithm begins with a randomly selected set of 20 points. The IIRABM simulation then runs a fixed number of stochastic replicates of the input points to determine class membership. This information is then used to train the ML model. The algorithm then ranks the remaining unlabeled parameterizations by class-membership uncertainty (see Eq. 1).

$$\{x_{i+1}\} = \min_x \left(0.5 - P_i(y|x)\right)$$

Those parameterizations whose class is most uncertain in the current ML model are then selected for labeling and the process repeats until a stopping criterion is reached; for the purposes of this work, once the cross-validation accuracy crossed 0.95, the algorithm was stopped.

The upper-level AL workflow uses a modified version of Dropout-based AL for regression presented in (20), hence referred to as Double Monte-Carlo Dropout Uncertainty Estimation (DMCDUE). The goal of this AL-workflow is twofold: to predict the volume of CR space and to predict the centroid location of CR-space, given a model internal parameterization. For each regression task, we assume that there exists a function,

$$y = f(\vec{x}), x \in \chi \subset \mathbb{R}^n, y \in \mathbb{R}$$

which approximates a map of CR space as a function of internal model parameterization which comprises the training set:

$$D_{train} = \{x_j^t, f(x_j^t)\}$$

for $j = 1, \dots, n$. The NN model uses a mean-squared error (MSE) loss function, given by:

$$L = \sum_{i=1}^{n} \left(f(x_i^t) - \hat{f}(x_i^t)\right)^2$$

Where f_i is the ground truth value for either the CR volume or centroid and $\hat{f}_i$ is value regressed by the NN. In this scheme, we utilize a four-layer fully-connected neural network with a 256-Dropout-128-Dropout architecture. The dropout layer (27) serves to provide a stochastic variability to the output of the NN.

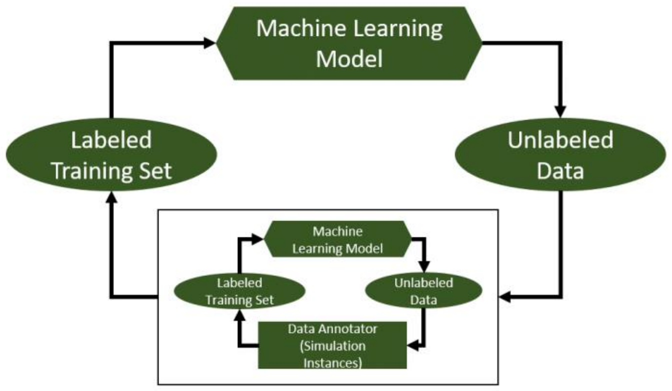


Figure 1: A diagram illustrating the nested active learning workflow

We begin by pre-selecting 10,000 (out of 40,353,607) internal parameterizations randomly; this random selection then makes up the available pool, $\mathcal{P}$,of unlabeled data. From this pool, we begin the AL procedure by selecting 100 internal parameterizations randomly from $\mathcal{P}$. These internal parameterizations are then fed into the lower-level AL workflow, which is used to map the CR space and return an approximate volume and center-point. This data is then used to train the upper-level neural net (see Fig. 1). The trained NN is then used to predict the volume or centroid location for the remaining unlabeled data for 10 stochastic replicates (the dropout layer provides stochasticity). The parameterizations from $\mathcal{P}$ which have the highest variance are selected for labeling, and this process repeats. Pseudocode for this procedure is given below:

```
1. Initialize training pool $\mathcal{P}_U$; upper-level dataset $D_{IP}$; $z_u$, the maximum size of the dataset; and $m_u$ samples to be added on each iteration,
2. While $|D_{IP}| < z_u$:
    a. For each element i in $D_{IP}$:
        i. Initialize training pool $\mathcal{P}_L$; lower-level dataset $D_{EP}$; $z_l$, the maximum size of the dataset; and $m_l$ samples to be added on each iteration,
        ii. While $|D_{EP}| < z_l$:
            1. Train network on $D_{EP}$
            2. Obtain rank $r_j$ for each $x_j$ in $\mathcal{P}_L$ according to maximum class-uncertainty
            3. Label the set of $m_l$ parameterizations from $\mathcal{P}_L$
            4. Add the annotated data to $D_{EP}$
            5. Calculate stopping metrics, stop if appropriate
    b. Train network on $D_{IP}$
    c. Obtain rank $r_i$ for each $x_i$ in $\mathcal{P}_U$ according to maximum regression variance
    d. Label the set of $m_u$ parameterizations from $\mathcal{P}_u$
    e. Add the annotated data to $D_{IP}$
    f. Calculate stopping metrics, stop if appropriate
```

Source code and input data can be found at: https://bitbucket.org/cockrell/iirabm_al/

Results

In the lower-level AL workflow, we map CR space as a function of four parameters, external to the IIRABM's internal rule set. An example of this space can be seen in Fig. 2. In this figure, outcome spaces for patients with low environmental toxicity (toxicity=1) to high environmental toxicity (toxicity=10) are shown from left to right. Each point represents 4000 *in silico* patients (40 injury sizes, 100 stochastic replicates). Points are color-coded based on the outcomes generated. The CR space is shown in green.

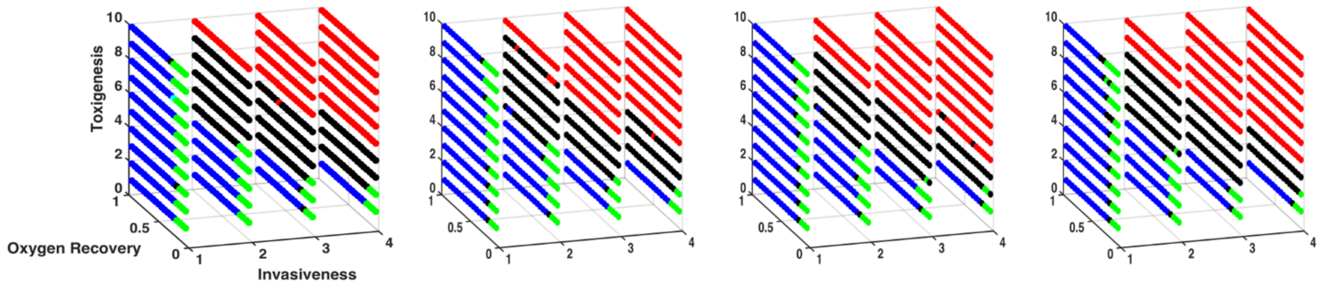


Figure 2: Outcome spaces for patients with low environmental toxicity (toxicity=1) to high environmental toxicity (toxicity=10) are shown from left to right.

We utilized seven different ML models to map the CR space: Artifical Neural Net, Adaptive Boosting, Naïve Bayesian, Random Forest, TreeBag, AdaBoost M1, Bag – Flexible Discriminant Analysis with Generalized Cross Validation. Results from this are shown in Fig. 3, which displays the F-score as a function of AL iteration number (and by proxy, dataset size).

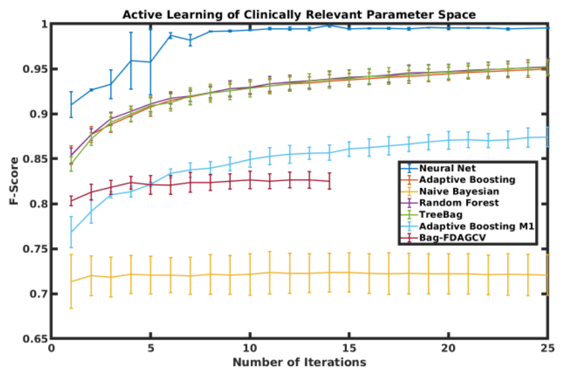
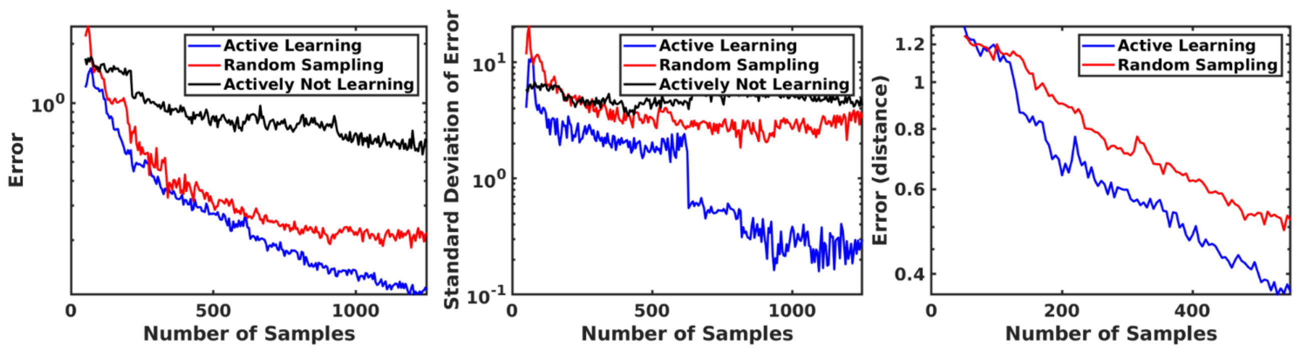


Figure 3: F-score as a function of Active-Learning iteration for a suite of ML techniques

It is readily apparent that a NN is the best type of ML model for mapping this space. By iteration 10, which uses 1000 parameterizations (out of 8800 possible), we can achieve an average class-prediction accuracy of >98%. The resulting ML model is then utilized to efficiently calculate the location and centroid of the CR space and train the upper-level neural net. Results from the upper-level AL procedure are shown in Figure 4. In panel A, we display the percent volume error as a function of the number of training samples for AL, Random Sampling (RS), and "Actively Non Learning" (ANL). ANL refers to utilizing the opposite of the AL sampling criterion. In this case, for AL we chose samples that maximized prediction variance; for ANL, we chose samples that minimized prediction variance. As expected, AL outperforms RS and requires fewer samples to converge to the error minimum. Additionally, both methods significantly outperform ANL, as expected. In Panel B, we show the standard deviation of the error for the previous three methods. Here, AL significantly outperforms RS in that the intelligent sampling criterion leads to a suite of models with a larger degree of precision in the volume prediction, whereas the changes in standard deviation of the error are minimal for ANL and minimal for RS after the first few samples. Panel C displays the error (as a Euclidean distance) as a function of the number of samples. Once again, AL outperforms RS, though by a relatively modest amount.



Discussion

We have described a nested active learning workflow which efficiently and accurately can characterize a high-dimensional Random Dynamical System. We remove inefficiencies due to oversampling small regions of parameter space using the Double Monte-Carlo Dropout Uncertainty Estimation (DMCDUE) approach. We note that AL outperforms RS in both the volume and centroid location predictions, but the greatest strength comes from the significant increase in precision generated by a suite of AL-trained models.

This work demonstrates that comprehensive (and accurate) exploration of computational models with many parameters is both possible and computationally tractable, given current techniques in machine learning and high-performance computing.

Acknowledgements

This research used high performance computing resources of the National Energy Research Scientific Computing Center, a DOE Office of Science User Facility supported by the Office of Science of the U.S. Department of Energy under Contract No. DE-AC02-05CH11231, as well as resources provided by the Vermont Advanced Computing Core (VACC). Additionally, this research was supported in part by NIH through resources provided by the University of Chicago Computation Institute (Beagle2) and the Biological Sciences Division of the University of Chicago and Argonne National Laboratory, under grant 1S10OD018495-01. This work was also supported by funds from Lawrence Livermore National Laboratory under Award #B616283.

Figure Captions

Figure 1: A diagram illustrating the nested active learning workflow

Figure 2: Outcome spaces for patients with low environmental toxicity (toxicity=1) to high environmental toxicity (toxicity=10) are shown from left to right. Each point represents 4000 *in silico* patients (40 injury sizes, each with 100 stochastic replicates). Points are color-coded based on the outcomes generated. Blue points represent simulations that healed under all circumstances. These points are distributed in regions of space where host resilience is high and the bacterial virulence is low (lower invasiveness and lower toxigenesis). Red points represent simulations that always died from overwhelming infection; these points are distributed in regions of high bacterial virulence (higher values for invasiveness and toxigenesis). Black points represent simulations that either died from overwhelming infection or healed completely and mark the boundary between simulations that always heal and simulations always die from infection. Pink points represent simulations which either died from overwhelming infection or hyperinflammatory system failure; these points are found primarily in the simulations that were treated with antibiotics and had low values for environmental toxicity and host resilience. Green points represent the Clinically Relevant simulations as these parameter sets lead to all possible outcomes; these points are distributed in regions of low to middle values of the

host resilience parameter and moderately virulent infections. For all classes of simulation, the final outcomes are primarily dependent on the host resilience and microbial virulence

Figure 3: Results from Lower-Level AL – Clinically Relevant space (see Fig. 2) is mapped as a function of four parameters, external to the IIRABM's internal rule set. We utilized seven different ML models to map the CR space: Artifical Neural Net, Adaptive Boosting, Naïve Bayesian, Random Forest, TreeBag, AdaBoost M1, Bag – Flexible Discriminant Analysis with Generalized Cross Validation. The F-score is shown on the y-axis as a function of the number of AL iterations performed.

Figure 4: Results from Upper-Level AL –In Panel A, we show the percent volume error as a function of the number of input training samples using Active Learning (AL), Random Sampling (RS), and Actively Not Learning (ANL), in which the learning criterion is the opposite of the AL criterion. We see that AL arrives at a more accurate prediction with fewer samples than RS or ANL. In Panel B, we show the standard deviation of the error of the volume prediction for the three above methods and note that AL not only generates a suite of more accurate models, but also has a much higher degree of precision. In Panel C, we show the error (in this case a Euclidean distance) in the centroid location prediction. AL once again outperforms RL.

1. Wood KA, Angus DC. Pharmacoeconomic implications of new therapies in sepsis. Pharmacoeconomics. 2004;22(14):895-906.
2. Cockrell C, An G. Sepsis reconsidered: Identifying novel metrics for behavioral landscape characterization with a high-performance computing implementation of an agent-based model. J Theor Biol. 2017;430:157-68.
3. Cockrell RC, An G. Examining the controllability of sepsis using genetic algorithms on an agent-based model of systemic inflammation. PLoS computational biology. 2018;14(2):e1005876.
4. An G, Nieman G, Vodovotz Y. Computational and systems biology in trauma and sepsis: current state and future perspectives. International journal of burns and trauma. 2012;2(1):1.
5. Goldman D, Bateman RM, Ellis CG. Effect of sepsis on skeletal muscle oxygen consumption and tissue oxygenation: interpreting capillary oxygen transport data using a mathematical model. American Journal of Physiology-Heart and Circulatory Physiology. 2004.
6. Vodovotz Y, Billiar TR. In Silico Modeling: Methods and Applications toTrauma and Sepsis. Critical care medicine. 2013;41(8):2008.
7. Vodovotz Y. Computational modelling of the inflammatory response in trauma, sepsis and wound healing: implications for modelling resilience. Interface focus. 2014;4(5):20140004.
8. Siqueira-Batista R, Gomes A, Possi M, Oliveira A, Sousa F, Silva C, et al., editors. Computational modeling of sepsis: perspectives for in silico investigation of antimicrobial therapy. II International Conference on Antimicrobial Research-ICAR2012 Lisbon (Portugal); 2012.
9. An G, Nieman G, Vodovotz Y. Toward computational identification of multiscale "tipping points" in acute inflammation and multiple organ failure. Annals of biomedical engineering. 2012;40(11):2414-24.
10. Karp RM. On the computational complexity of combinatorial problems. Networks. 1975;5(1):45-68.
11. Sneddon MW, Faeder JR, Emonet T. Efficient modeling, simulation and coarse-graining of biological complexity with NFsim. Nature methods. 2011;8(2):177.
12. Hopfield JJ, Tank DW. "Neural" computation of decisions in optimization problems. Biological cybernetics. 1985;52(3):141-52.
13. Edwards R, Glass L. Combinatorial explosion in model gene networks. Chaos: An Interdisciplinary Journal of Nonlinear Science. 2000;10(3):691-704.
14. Neumann F, Witt C. Combinatorial optimization and computational complexity. Bioinspired Computation in Combinatorial Optimization: Springer; 2010. p. 9-19.

15. Cockrell C, Christley S, An G. Investigation of Inflammation and Tissue Patterning in the Gut Using a Spatially Explicit General-Purpose Model of Enteric Tissue (SEGMEnT). PLoS computational biology. 2014;10(3):e1003507.
16. Cohn DA, Ghahramani Z, Jordan MI. Active learning with statistical models. Journal of artificial intelligence research. 1996;4:129-45.
17. Brinker K. On active learning in multi-label classification. From Data and Information Analysis to Knowledge Engineering: Springer; 2006. p. 206-13.
18. Huang S-J, Jin R, Zhou Z-H, editors. Active learning by querying informative and representative examples. Advances in neural information processing systems; 2010.
19. Schein AI, Ungar LH. Active learning for logistic regression: an evaluation. Machine Learning. 2007;68(3):235-65.
20. Tsymbalov E, Panov M, Shapeev A, editors. Dropout-Based Active Learning for Regression. International Conference on Analysis of Images, Social Networks and Texts; 2018: Springer.
21. Ozik J, Collier NT, Wozniak JM, Macal CM, An G. Extreme-Scale Dynamic Exploration of a Distributed Agent-Based Model With the EMEWS Framework. IEEE Transactions on Computational Social Systems. 2018(99):1-12.
22. Ozik J, Collier N, Wozniak JM, Macal C, Cockrell C, Friedman SH, et al. High-throughput cancer hypothesis testing with an integrated PhysiCell-EMEWS workflow. BMC bioinformatics. 2018;19(18):483.
23. Ozik J, Collier NT, Wozniak JM, Spagnuolo C, editors. From desktop to large-scale model exploration with Swift/T. 2016 Winter Simulation Conference (WSC); 2016: IEEE.
24. Wozniak JM, Armstrong TG, Wilde M, Katz DS, Lusk E, Foster IT, editors. Swift/t: Large-scale application composition via distributed-memory dataflow processing. 2013 13th IEEE/ACM International Symposium on Cluster, Cloud, and Grid Computing; 2013: IEEE.
25. Wozniak JM, Jain R, Balaprakash P, Ozik J, Collier NT, Bauer J, et al. CANDLE/Supervisor: A workflow framework for machine learning applied to cancer research. BMC bioinformatics. 2018;19(18):491.
26. De Boer P-T, Kroese DP, Mannor S, Rubinstein RY. A tutorial on the cross-entropy method. Annals of operations research. 2005;134(1):19-67.
27. Srivastava N, Hinton G, Krizhevsky A, Sutskever I, Salakhutdinov R. Dropout: a simple way to prevent neural networks from overfitting. The Journal of Machine Learning Research. 2014;15(1):1929-58.