Whitepaper

Building Intelligent Digital Twin Models

using

Neural Networks

2025

Table of Contents

1. Introduction

In Simio, complex decision logic can be simplified by using neural network regression to infer information when the relationship between inputs and outputs is complicated, such as estimating order lead times. Simulation runs can not only use neural network models for estimation and inference but also automatically generate synthetic training data to monitor their predictive performance and retrain them. By using neural networks to simplify decision logic within a simulation model, the focus shifts to modeling system components and their interactions. This makes the digital twin simulation model easier to build, understand, debug, and maintain.

Neural networks are a subset of machine learning algorithms, inspired by the human brain. A neural network model is comprised of node layers: an input layer, one or more hidden layers, and an output layer. A network with two or more hidden layers is referred to as a deep learning network.

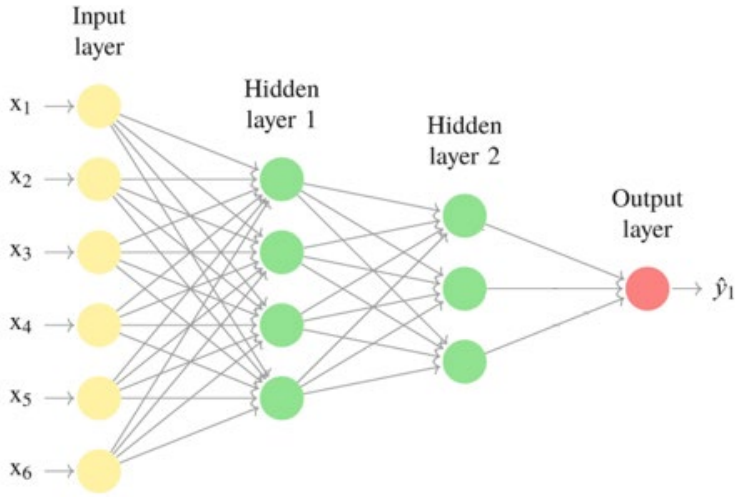

Figure 1: Typical Feed Forward Neural Network Illustration

A feedforward neural network is the first and simplest form of a neural network, in which data moves in only one direction from the input layer to the output layer (without loopbacks). The network has parameters called weights and biases, which are established through a process called training. These weights and biases are then used to transform the inputs at each node into an output that is sent to the next layer of nodes.

A regression neural network predicts one or more numerical outputs given a set of numerical inputs. Although neural networks are widely known for modeling complex problems such as image recognition and generative AI applications like ChatGPT, they are also well-suited for regression applications, such as predicting a KPI in a system given the current system state. This makes them an ideal framework for embedding AI in digital twin simulation models.

2. Simio Digital Twins & Neural Networks

Simio is the first and only discrete-event-based digital twin simulation software to offer comprehensive, embedded AI features, fully supporting the creation and automatic training of regression neural networks within a model—without requiring Python or Java programming or integration with external third-party AI tools. Simio enables the definition of one or more neural network models, which can then be referenced for inference within a Simio model through a neural network element.

2.1 Synthetic Training Data for Neural Networks

One of the major challenges of AI is obtaining the required labeled training data. Many AI applications fail due to the unavailability of training data. Labeled training data is never available when evaluating a new system, and in cases involving existing facilities, any recorded data becomes invalid once a new part is introduced or a change in flow occurs in the system. That's why Simio's built-in auto-labeler for creating synthetic training data is critical to the success of an intelligent modeling application.

In many cases, machine learning algorithms are used to select among available choices. In simulation models, users face complex decisions, such as which entity to process next and which production line should be used to process a new order. The problem with using a neural network for selection in a simulation model is that there is no built-in mechanism to auto-label the correct choice without running separate simulations for each option and comparing the results. This approach is difficult and impractical in most applications.

A much better approach is to convert the selection problem into a regression problem, where the goal is to predict a KPI of interest and then choose the smallest or largest KPI value. For example, when selecting between alternate production lines, the line with the smallest predicted

makespan can be chosen. Within the Simio model, the actual makespan for the selected line can be recorded, and the error between the predicted and actual makespan can be used to train the neural network to improve its future predictions. Hence, the model-based makespan for the selected line is recorded to label the inputs for use as training data.

Recording labeled training data for the neural network is typically a two-step process: first, the inputs for predicting the KPI are recorded, and later, the actual KPI value as determined by the model is established. For example, when predicting the makespan for a candidate production line, the inputs may include the number of jobs of each type currently waiting at each station along the candidate line. These inputs are then used by the neural network to predict the makespan for that job on that line.

Once the line with the smallest predicted makespan is selected, the job moves down the line, and upon completion, the actual makespan is recorded and matched with the original inputs to the neural network. This matching between inputs and outputs is done using a training key (e.g., the job ID). The labeled training record is then added to the training repository. The entire process of recording inputs, matching them with outputs, and adding training records to the repository is automated within Simio.

2.2 Embedding Third Party Machine Learning Algorithms

Simio's built-in AI features support the definition, training, and use of the classic feedforward regression neural network. However, as illustrated in Figure 2, users are not limited to this machine learning algorithm. Any machine learning regression model from over 50 third-party providers, including Google and Microsoft, that supports the ONNX model exchange format can be imported and used within Simio. Users can build and train models in third-party tools, then import them into Simio for complex decision-making within a model. In addition to importing models, Simio also supports exporting them to the ONNX format. Additionally, Simio models can generate synthetic labeled training data for export, allowing seamless integration with third-party tools.

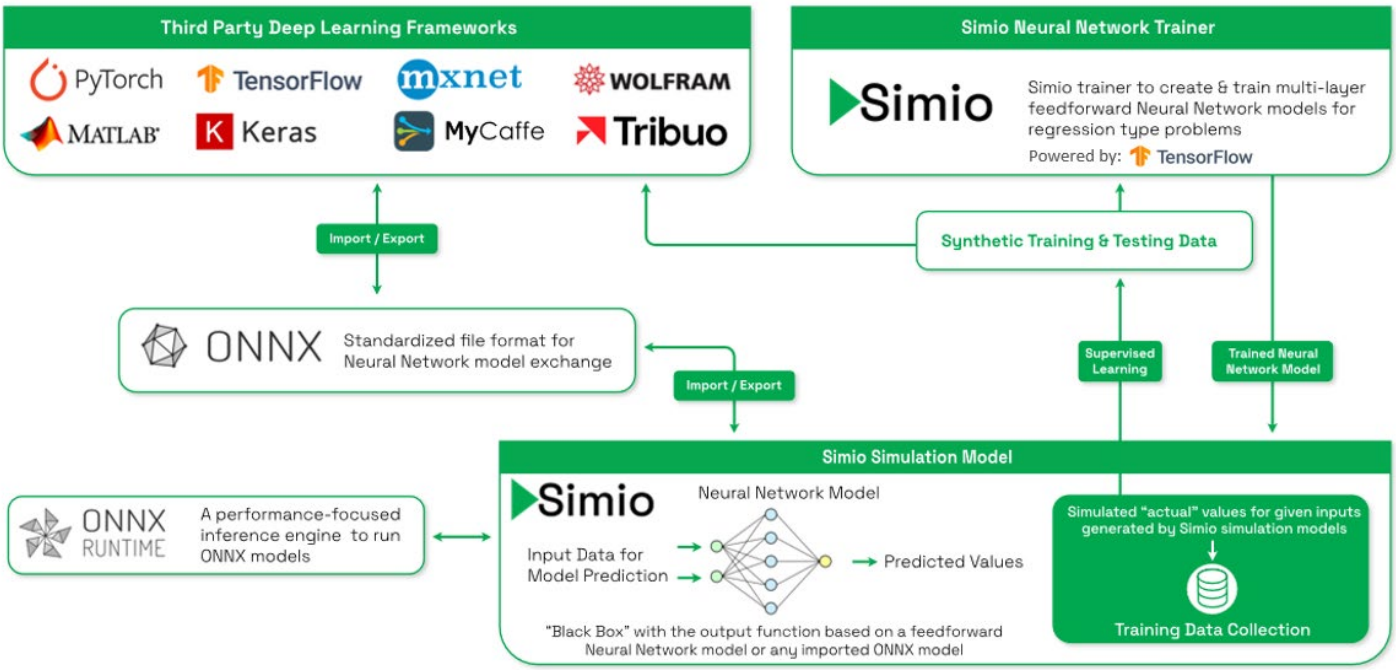

Figure 2: Using Simio Neural Network Trainer as well as Third Party ONNX Models

3. Defining, Training, and Using Neural Networks in Simio

This Simio neural network framework utilizes a Neural Network Model and its associated Neural Network Element. The Neural Network Model represents a trained neural network regression model for inference. It can either be a feedforward model, created and trained directly in Simio (no code or third-party framework required), or an imported ONNX model. The Neural Network Model definition is integrated into the simulation logic via a Neural Network Element. This element defines the model's inputs and "actual" output, along with event-based simulation triggers for recording input and output values. Additionally, the element provides a Predicted Value function that can be used in any simulation expression for inference purposes.

The relationship between a Neural Network Model and a Neural Network Element is one-to-many, allowing a single Neural Network Model to apply to multiple use cases in the simulation. This relationship between the Neural Network Model and its associated Neural Network Elements is illustrated in Figure 3.

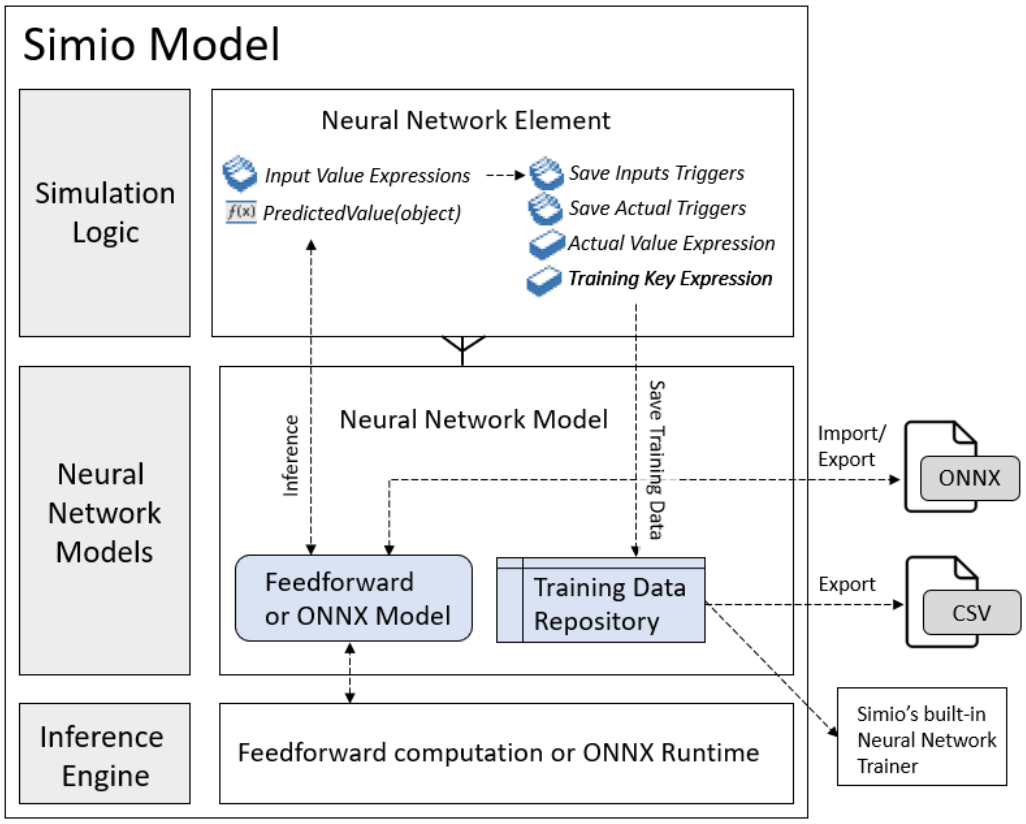

Figure 3: Neural Network Model and Elements

3.1 Training and Testing Neural Networks

Note that the Neural Network Element plays two important roles in a Simio model. First, it is used to predict KPIs, such as makespan, to optimize the decision logic within the model. Second, it automatically records synthetic training data, which can then be used to retrain and improve the predictive power of the neural network model. This process is illustrated in Figure 4.

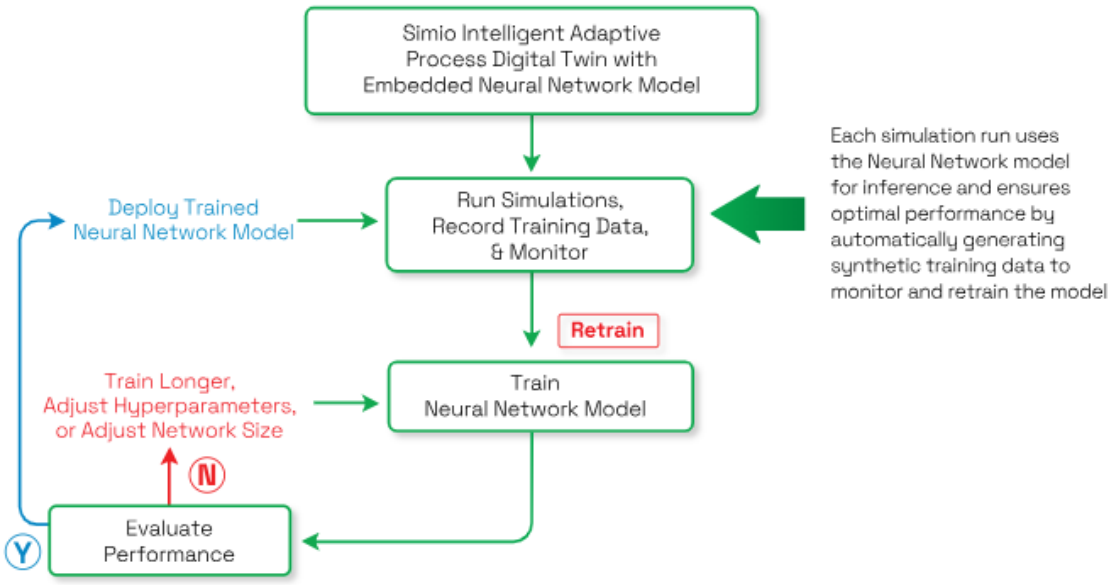

Figure 4: Neural Network Training Process

The synthetic training data recorded and saved in the training repository can be used internally to train Simio's feedforward neural networks or exported for use by third-party tools. Training Simio's feedforward networks is done using Simio's built-in training algorithm, powered by TensorFlow, an open-source ML and AI software library from Google. TensorFlow, recognized as one of the most popular deep learning frameworks, enhances Simio's neural network training with state-of-the-art performance. This training algorithm iteratively works through all records in the training dataset, one record at a time and in random order, making incremental adjustments to the neural network's weights and bias parameters to reduce prediction error.

When training, it is common to separate the training records into three distinct datasets: Training, Validation, and Test. The Training dataset is used to fit the model by adjusting parameters to minimize prediction error. The Validation dataset is used to provide an unbiased estimate of the model's performance and stops training when adaptability to new data no longer improves. The Test dataset provides a final unbiased estimate of the model's performance. This Test dataset is important because the process of model "tuning" to optimize performance on the validation data set can result in a form of overfitting.

A challenge in training a neural network model is determining how long to train. Too many epochs can lead to overfitting, while too few may result in an underfit model. A compromise is to train on the training dataset and stop when performance on the validation dataset begins to degrade (i.e., when validation error starts increasing). This simple, effective, and widely used method to train a neural network is called “early stopping”.

In the simplest case, training stops as soon as the mean squared error on the validation dataset is greater than the error at the end of the previous epoch. However, because neural network training is stochastic and validation error may fluctuate, introducing a delay or patience period is generally beneficial. Simio’s Neural Network Trainer includes a Validation Patience setting, which specifies the number of consecutive epochs the mean squared error on the validation dataset can be greater than or equal to the error from the previous epoch before training is automatically stopped.

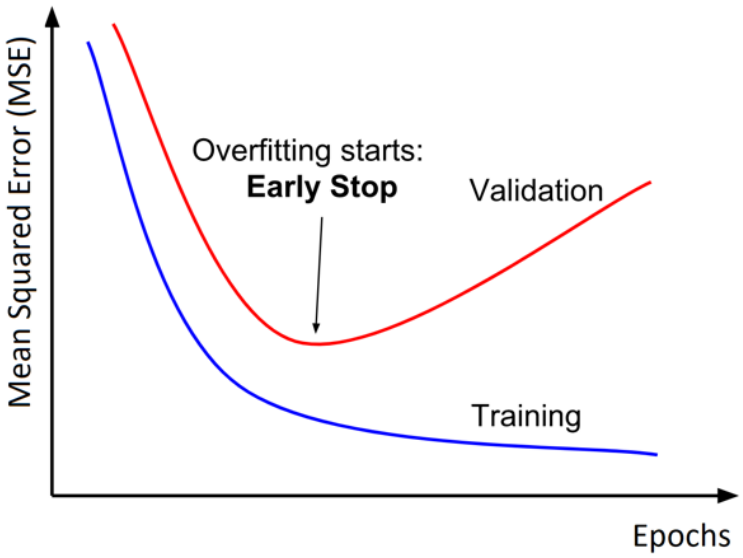

Figure 5: Neural Network Early Stopping Logic

Simio’s Neural Network Trainer is used to specify training hyperparameters and initiate training. Key hyperparameters include the allocation of training records to the Validation and Test datasets, the maximum number of random training epochs to run, and the number of nodes in each hidden layer of the neural network. The trainer displays training progress through a graph that depicts the mean squared error between the predicted and actual values for each of the three datasets.

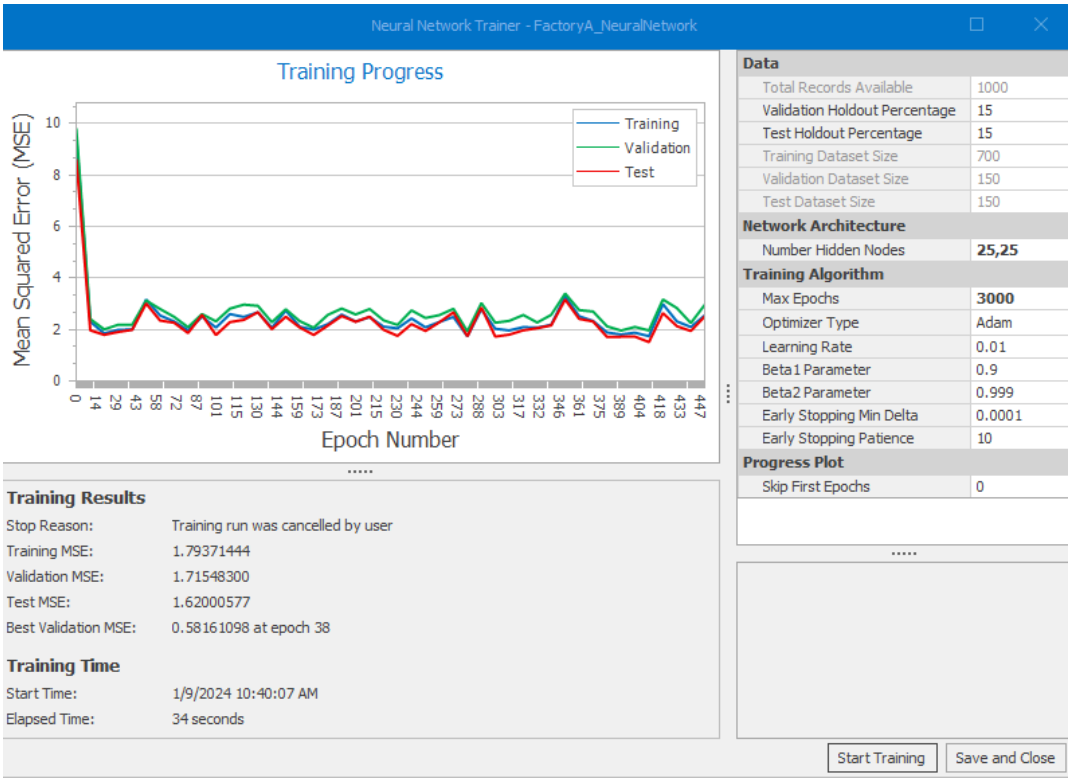

Figure 6: Simio's Neural Network Trainer Powered by TensorFlow

3.2 Neural Networks in Production Planning and Scheduling

Simio's AI features are particularly useful in production planning Digital Twin applications, where the neural network can be trained to predict critical KPIs, such as the dynamically changing production lead time for a factory or a production line within a factory. The neural network learns the impact of changeovers, secondary resources, business rules, and other production complexities that affect KPI predictions. The intelligent Digital Twin captures complex relationships that would otherwise be impossible to include in a model.

Neural network KPI predictions can then be used to optimize decisions both within the factory and across the supply chain. Within the supply chain, the neural network supports critical supplier sourcing decisions by predicting the production lead time for each candidate supplier and selecting the lowest-cost producer that can complete the order on time. AI-based factory sourcing within the supply chain Digital Twin eliminates the need for Master Production Scheduling software, which employs a rough-cut capacity model that ignores production

constraints such as changeovers, assumes fixed lead times regardless of factory loading, and schedules into artificial time buckets using a heuristic algorithm. This approach results in rough-cut, non-optimal schedules that require hours of computing time to produce and fail to align with detailed factory schedules.

4. Summary

In summary, Simio harnesses the power of AI to enhance simulation, enabling the creation of Intelligent Digital Twins that optimize decision-making within a model. Additionally, neural networks replace complex decision logic in a model by learning the impact of interactions between system components, allowing neural networks to accurately predict critical KPIs, such as actual production lead time based on current workload and product mix.

Simio's feedforward neural networks as well as any ONNX-compatible machine learning model, can be directly embedded in a Simio model. These networks can be trained using data automatically recorded by the model. Models incorporating AI and Simio's neural network constructs are not only easier to build but also more powerful in optimizing decision-making.