Startup-O - Sproutfish ML Approach Paper

Karn Dalmia

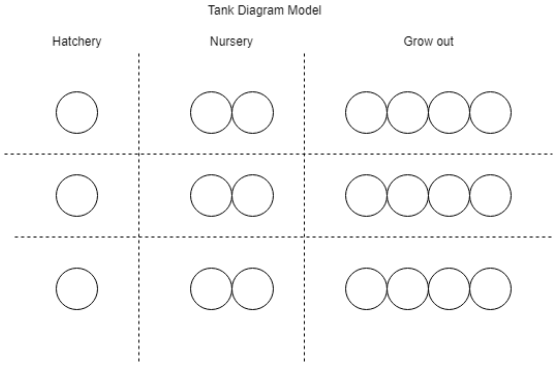
July 2019

1 Introduction

For this project, our goal is to optimize aquaculture growth practices by optimizing a combination of several "optimizing" variables whilst toggling several tune-able parameters.

1.1 Development Stages

In the tank model,we will have three stages of development: Hatchery, Nursery, and Growout. Each stage of development will have different input and output parameter combinations, and will effectively be evaluated using three different (independent) groups of neural network architectures.



2 Existing Approaches

With the existing method, all the data is collected by hand, categorized by type, and uploaded to a CSV or Excel file for further processing. In the data pipeline, there is a predefined set of input parameters, be it water temperature, pH level, average weight, biomass, that's normalized and fed to a relevant machine learning model (ANN, SVM, etc) with a time-series component. Once done, it tries to predict a series of output parameters of high importance to an end user.

Our approach will be somewhat similar to that of paper with two notable differences.

First, we will strive to upload relevant sensor + manual data to a cloud location at a prescribed frequency. Once done, we can host a machine learning pipeline to output the optimizing parameters we are trying to optimize for.

Second, in order to evaluate which set of tun-able parameters best optimize the optimizing parameters, we hope to write an add on optimizing function that scores the numeric output parameters. Once done, we can take the tunable parameter configuration with the highest numerical scores of the optimizing, and output that to the farmer.

3 Targeted Productivity Outcomes

The approach is unique in that it targets optimisation of existing farm capacities on across farm types and species - particularly relevant to existing aquaculture farms and practices. The goal is to adapt farming practices "on-the-fly" across environmental, species specific and water quality configurations that have a tendency to change at short notice. This is in contrast to highly capital intensive, advanced farms set up from scratch in the manner of industrial plants.

The learning algorithms consume data from an existing farm process, optimise the variables across growth stages, and inform the farm management via easy to use apps that allow for parameter management that optimises business outcomes. These can vary from mortality rate reduction at the hatchery level, to stocking density increases and feed cost reductions at the grow-out stage.

Methodologies that minimise upfront expenses and hence pay for themselves due to enhanced output have been favoured to increase the relevance to smaller farmers across the region. The cumulative effects of the learning when applied across the growth stages are expected to compound the benefit effects - resulting in an higher effective capacity utilisation and overall risk reduction in crop production - independent of specific species of fish.

4 Optimizing and Tunable Parameters

4.1 What to optimize for?

Our neural network model effectively inputs several parameters for a window of time (usually seven days), (one for each stage), and outputs predictions of those same parameters, some of which contribute to our overall optimization function. Below is a list of our current optimization parameters and formula by growth stages, subject to change:

4.1.1 Hatchery

- Mortality (minimize): $\frac{fish\ that\ died}{total\ number\ of\ fish}$

4.1.2 Nursery

- Growth Rate: if our growth rate model is $L_\infty(1 - e^{-K\times(t-t_0)})$, then we want to maximize K, which is 1-$e^{-k_2\times K}$, where 'k_2' is an adjustable constant w.r.t growth rate for a 7-day window.

4.1.3 Growout

- Stocking Density (maximize): $\underline{1 - e^{-k_1\times x}}$, where k_1 is a constant w.r.t. stocking density.
- Feed-Conversion Ratio (maximize): $\underline{1 - e^{-k_2\times x}}$, where k_2 is a constant w.r.t feed-conversion ratio.

Optimization Score = $\sum_i w_i \times param\_score_i$, where $\sum_i w_i == 1$; for simplicity, we will assume equal weights for multi-optimizing outputs.

4.2 What are our tune-able parameters?

We will work with several tune-able parameters, listed below:

4.2.1 Water Parameters (all stages)

- **Temperature**:
  - Range: $0°C - 45°C$;
  - Step Size: $0.5°C$
- **Dissolved Oxygen Level** :
  - Range: [6-9]($\frac{mg}{L}$);
  - Step Size: 0.1($\frac{mg}{L}$)
- **pH level**:
  - Range: 7.5-8.3;
  - Step Size: 0.05
- **Salinity**:
  - Range: [30-35]ppm;
  - Step Size: 0.1ppm
- **Amonia Level**:
  - Range: [0-0.10]($\frac{mg}{L}$);
  - Step Size: 0.01($\frac{mg}{L}$)
- **Nitrate Level**:
  - Range: [0-0.15]($\frac{mg}{L}$)
  - Step Size: 0.01 ($\frac{mg}{L}$)

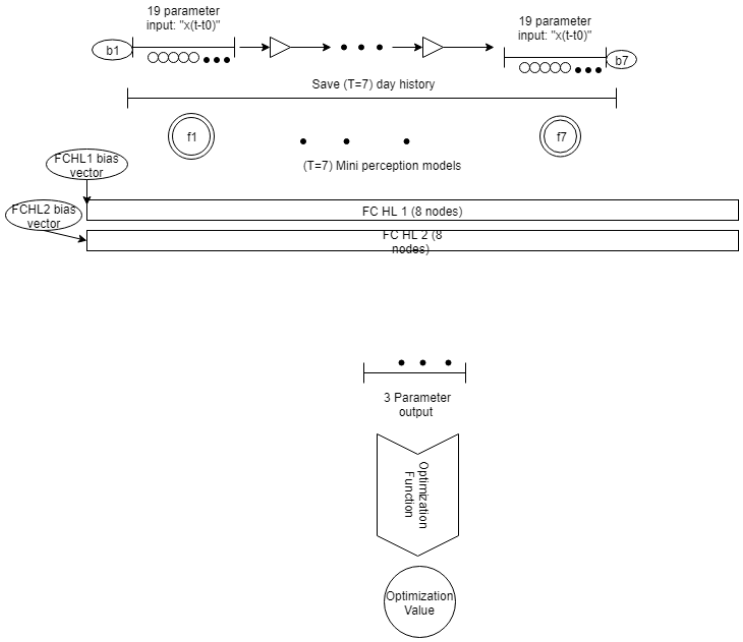
4.2.2 Manual Inputs (Nursery + Growout)

- Feed frequency: $1 - e^{-k \times x}$, where k is a constant, and x is the frequency at which food is fed.
- Power consumption: $\frac{Average\ power\ consumption\ T{=}7\ day\ history}{Max\ Power\ Consumption}$

5 ML Architecture

5.1 High level overview

In our trainable machine learning model, we want to train the parameter weights for our inputs and corresponding outputs. In this architecture, we will have $n_i \times T$ inputs (number of input parameters we capture for a T day history), two hidden layers, each with some intermediate number of layers, and n_i output layers, which will be the variables we are trying to optimize for.
For a particular configuration, we will cycle for D-days (usually 7), yielding a total of D-T+1 data-points for any particular window and cycle time. Below, we present a brief diagram of our neural network architecture.



5.2 Multiple ML Models

There are generally three stages of life. Hatchery, nursery, and growout. We can effectively create three separate neural network architectures to evaluate each one independently.

6 How to solve our problem with ML?

6.1 How to tune?

Given we have 4 tun-able parameters, and given there is a range of values for each one, we basically need 81 configurations (either parameter can go up or down or not change by a certain delta), to yield all possible changes. It is important to note that in the initial stages, with an untrained neural network, we won't be able to predict which toggle configuration is best. In this case, we basically need to input different parameters based on prior inference.

6.2 What to save?

Since our neural network model, in it of itself, doesn't account for the optimization function (i.e that's its own add-on to the piece), we need to basically save the toggle changes with the highest optimization value calculated from the formula above after training (described below). Once done, we can repeat the process for next cycle for the new configuration.
Since we will have 3 sets of tanks, each cycle will effectively be $\underline{81 \times \frac{D-T+1}{3}}$, where 81 is the number of toggle configurations, 3 is the distributed work over all the rows of tanks, and the formula in the numerator is what was described above.

6.3 Training period

We will initially train our neural network model for a 20-cycle period (described above), after which the optimal tun-able parameter combinations will be estimated using the ranges, step sizes, and optimization function specified above. Thereafter, predictions can be outputted.

6.4 Frequency of Data and Relevant Calculations

We will receive the input parameters, tun able as well as other changing non tun-able parameters, on a daily basis. One of the optimization values, feed-conversion ratio, will arrive weekly. For simplicity, we will use the same most recent feed-conversion ratio for any 7 day period. Growth rate will be inferred from a 7-day history of data points from the prescribed formula above; the relevant data will arrive daily. Power consumption data will arrive daily, but will be a 7-day historical average in the optimization calculation.

6.5 How to predict?

Once we have sufficient training data for our neural network, optimizing for the best possible set of configurations, we can predict which toggle configuration will yield the highest optimization. It is important to note that once we have the neural network in place,we can use the new configurations as training data as well, once we see the actual output.

7 Cloud Communication

7.1 Sending/Receiving the data

All relevant sensor data needs to be sent to a cloud location (either Google Cloud or AWS) at the frequencies described above. Once done, the relevant predictions (toggle configurations) will be outputted to an output server.

7.1.1 Google Cloud Dataflow/Apache Beam SDK

The key goal with Google Cloud Dataflow is to stream sensor data from the fish tank sites to a cloud location. For each of the 6 tun-able parameters (temperature, Dissolved Oxygen Level, pH level, Salinity, Amonia Level, and (Nitrate Level)). Relevant electronic devices with Wi-Fi connectivity should send that data to the Cloud Location at the frequencies discussed above. The Apache Beam SDK will subsequently be used to interface with the said cloud location.

7.1.2 Cloud Machine Learning Pipeline with Tensorflow

Once data has been uploaded to the relevant cloud location, we need to effectively extract, pre-process, train, and predict the most optimal parameter configurations using the variables and optimizing function described above, and write back the optimal configurations back to Google Dataflow.

7.1.3 Google Dataflow to Firebase

Finally, once the optimal configurations of the tun-able parameters have been written back to Google Dataflow, we need to basically send the data of the top most, say 10 optimal configurations, to the iOS application for the farmer.

7.1.4 Cloud Diagram