Teaching a Neural Network to Fly Autopilot

Otito Mbelu (G00397738)

Abstract—The goal of this project is to train a Automatic Neural Network to navigate through an endless horizontally scrolling tunnel without clipping any of the edges. A programmed agent capable of consistently navigating the tunnel accurately was designed and implemented, and was further used as a trainer to generate training data for the Neural Network. Model selection was done using a brute force approach that calls for a grid search using varying ranges of hyper parameters to select an optimum model. Metrics for model evaluation was based on F1-Accuracy score and Mean Cross-Entropy Loss through cross-validation with training data. A network configuration of 21-20-3 with Rectified Linear Unit Activation Function (RELU) with a test accuracy score of 100% and MCCE loss of 0.1839 was selected as the best model. Furthermore, an autopilot agent utilizing Fuzzy Logic was implemented to solve the problem.

1 INTRODUCTION

The game consists of a 30×20 grid where the ship is at a fixed column and can navigate vertically i.e. up and down. A programmed agent capable of flawlessly navigating the tunnel in a consistent manner was designed to be the basis of training data generation. The agent's navigation algorithm was designed in the basis of navigating the ship through the middle of the weighted opening in the next n columns with respect to the ship current location. This implies that the next movement of the ship will attempt to get closer to the weighted center of the next n columns. A look forward horizon of 3 was chosen as the value for n. After trials with the ranges $2 \rightarrow 5, 3$ was selected because it performs the smoothest navigation through the tunnel amongst the values trialed. This resulted in a grid of $n \times (2n + 1)$ i.e. $3 \times 7 = 21$ input features. The following sections describes the design and model selection strategy used to implement the various agents.

2 AGENT DESIGN

All the AutoPilot agents implements the base abstract class Agent, which provides all the functionality for sampling and preprocessing data.

2.1 Programmed Agent

The behavior of this agent was implemented on the $ProgrammedAgent$ class. To predict which step to take, the weighted center of the opening of the cave is evaluated for a given horizon n using the softmax function of the proximity as weights. $W = \sigma(N)$, where σ is the softmax function and W is weight, $W = \{w_1, w_2, ..., w_n\}$ and $N = \{1, 2, ..., n\}$. The weighted center is evaluated by taking the mean of the dot product of the weight and the center of each column (See Fig. 1). $y = \frac{1}{n}\sum_{i=1}^{n} W_i \times \frac{h_i}{2} = \frac{1}{2n}(W \cdot H)$ where h_i is the height of the cave's opening for column i and H is the set of heights. The final prediction is determined by the value of y with respect to the ships current row. The ship will navigate UP if $y > player\_row$ and DOWN if $y < player\_row$ and will STAY put if $y = player\_row$

3 SAMPLING

Buffering

Data sampling strategy involves storing the entire grid beyond the ship's location (column) in a buffer and continuously consume the buffer until it runs out. The buffer will be refilled as soon as the buffer is exhausted. A pointer which points to the current position in the buffer was used to track the position of the buffer. The size of the buffer is given by $b_w \times b_h$, where b_h is the height of the stage and b_w starts from column next to the ship to the very end of the stage on the horizontal axis. This is demonstrated in Fig. 2. This implies that new data is sampled every $b_w - n$ frames, where b_w is the width of the buffer and n is the horizon.

Feature Extraction

At any given frame the agent will have to extract the relevant features from the buffer which is the next n columns beyond the ship (see Fig.2) with a height of $2n + 1$. The relevant features is given as $F = \bigcup_{i=1}^{n} F_i$, where F_i is the ith column of the relevant features and the union operation represents arraycopy. $F_i = \bigcup_{j=p(b_h)+r-n}^{2n+1} B_j$, where p = pointer, r is player row, b_h is the height of the buffer and B_j is the jth column of the buffer B. Data extraction runs at a big O of $O(|F|)$ where $|F| = n \times (2n + 1)$, assuming that arraycopy is done at $O(1)$.

4 FEATURE DESIGN

The value 3 was chosen for n i.e. the horizon, which resulted in 21 input features for the Neural Network. A classification model was chosen as three different output is expected, thus the output of the Neural Network consists of 3 nodes.

The selected input features for the model does not include the position of the ship. This decision was made after analyzing the training data which showed that the one-hot encoded values for the position only varies in 2 out of 630 training dataset. And thus was considered redundant.

5 Model Selection

A grid search algorithm taking different values for α, β, epochs, activation functions, loss functions, training speed, number of nodes per layer, number of hidden layers was implemented and used to search for the best model. The following range of values were used for the grid search, $\alpha \rightarrow (0.01, 0.001, 0.0001)$, $\beta \rightarrow (0.5, 0.75, 0.95)$, $min\_error \rightarrow (0.0001, 0.00001)$, $epochs \rightarrow (300, 500)$, $loss\_functions \rightarrow (CEE, MSE, SSE)$, $act\_functions \rightarrow (TANH, RELU, ISRLU)$, $layers \rightarrow (1, 2)$ Table 1 shows a summary of different models selected and their respective performances.

Metrics

Cross validation of 5 fold was used for model evaluation, and final performance recorded with unseen test data. F1-Accuracy was used as the first benchmark for model selection while the cross entropy loss was used as a tie breaker for models with similar F1-Accuracy scores.

Best Model

The best model was selected after evaluating all the models using unseen test data and the overall best performant model recorded a test accuracy of 100% and Mean Categorical Cross Entropy Error of 0.1839. The hyper-parameters and configuration for the best model are listed as follows: $\alpha = 0.01, \beta = 0.75, loss = CEE, max\_epochs = 300$, hidden layers = 1, topology = 21-20-3.

6 Extra - FuzzyAgent

This agent was implemented using a Fuzzy Inference System (FIS).

Fuzzification

The input variable to FIS is given by ϕ and ψ. The universe of discourse is given by the Φ and Ψ which both have ranges of $-90^\circ \rightarrow 90^\circ$. $\phi \in \Phi$, represents the angle of inclination of the ship with respect to the top clip of the cave opening, while $\psi \in \Psi$, represents the angle of declination of the ship with respect to the bottom clip of the cave opening. ϕ and ψ were evaluated as the weighted average of the angles between the ship and the cave's top and bottom edges of the opening along its horizon respectively. $\phi = W \cdot \vec{\phi}, \psi = W \cdot \vec{\psi}$, where $\phi = \{\phi_1, \phi_2, ..., \phi_n\}, \psi = \{\psi_1, \psi_2, ..., \psi_n\}$ and $\phi_i = tan^{-1}\left(\frac{y_{t_i}}{x_{t_i}}\right)$ and $\psi_i = tan^{-1}\left(\frac{y_{b_i}}{x_{b_i}}\right)$. The output of the FIS $f(\phi, \psi)$ were evaluated using the Center Of Gravity (COG) defuzzification technique.

Membership Function

The Membership functions for the input variables were defined by the Linguistic variables $high$ and low, and maps ϕ and ψ to μ_Φ and μ_Ψ respectively. The output $f(\phi, \psi)$ was defined using the linguistic variable up, $down$, and $still$ (See Fig.5)

Rules

The rules for FIS were defined as follows:

1 : IF ϕ IS low AND ψ IS low THEN output IS down;
2 : IF ψ IS high AND ϕ IS high THEN output IS up;
3 : IF ψ IS low AND ϕ IS high THEN output IS still;
4 : IF ϕ IS low AND ψ IS high THEN output IS still;

7 Conclusion

A programmed agent was designed as a training agent and was used to generate training data. A total of 1671 Neural Network models using 673 training data was built. 25% of the training data was reserved for testing and 75% was used for training and validation. Tabulated summaries of the evaluation results are presented in the Appendix. Both the NeuralNetwork Agent and FuzzyAgent were successfully implemented to navigate through the cave successfully. Buffering was implemented, which significantly reduced the frequency of sampling data. Data processing and evaluation libraries were developed to properly evaluate the models. However, there have been recorded rare instances of the FuzzyAgent clipping the edges of the cave resulting in a crash.

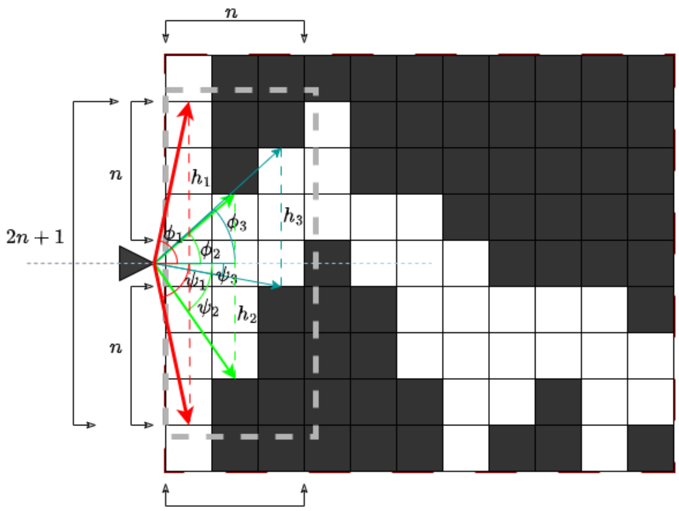
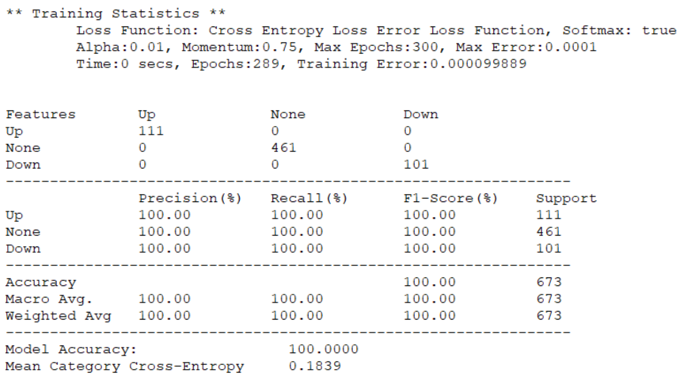


Fig. 1. Stage



```
** Training Statistics **
        Loss Function: Cross Entropy Loss Error Loss Function, Softmax: true
        Alpha:0.01, Momentum:0.75, Max Epochs:300, Max Error:0.0001
        Time:0 secs, Epochs:289, Training Error:0.000099889

Features        Up              None            Down
Up              111             0               0
None            0               461             0
Down            0               0               101
------------------------------------------------------------------
                Precision(%)    Recall(%)       F1-Score(%)     Support
Up              100.00          100.00          100.00          111
None            100.00          100.00          100.00          461
Down            100.00          100.00          100.00          101
------------------------------------------------------------------
Accuracy                                        100.00          673
Macro Avg.      100.00          100.00          100.00          673
Weighted Avg    100.00          100.00          100.00          673
------------------------------------------------------------------
Model Accuracy:                 100.0000
Mean Category Cross-Entropy     0.1839
```

Fig. 3. Report

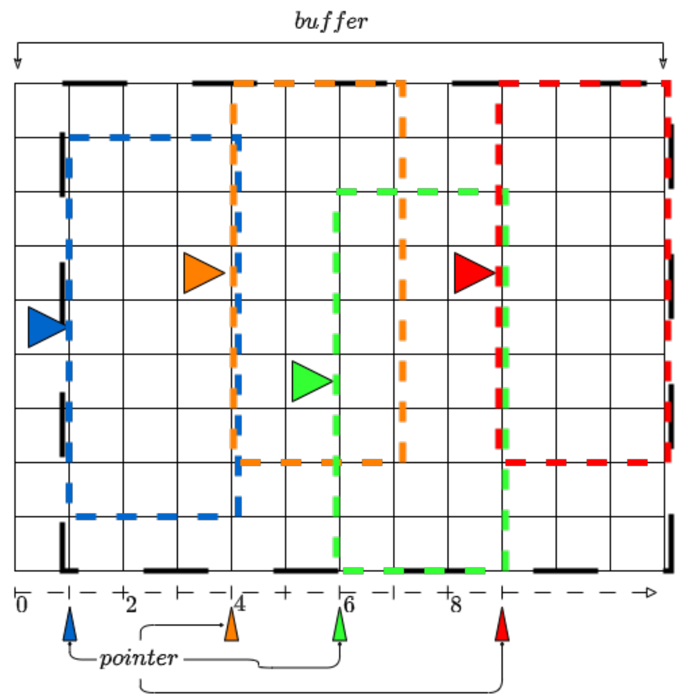


Fig. 2. Sampling

Index	Layers	Act.	Loss	Epochs	Alpha	Beta	Acc.%	MCE
1132	21-15-20-3	ISRLU	SSE	500	0.01	0.95	100.00	0.181
1120	21-15-20-3	ISRLU	SSE	500	0.01	0.5	100.00	0.182
1048	21-15-20-3	ISRLU	MSE	500	0.01	0.5	100.00	0.183
1067	21-20-45-3	ISRLU	MSE	300	0.01	0.95	100.00	0.183
1049	21-20-45-3	ISRLU	MSE	500	0.01	0.5	100.00	0.183
30	21-12-3	ISRLU	CEE	500	0.01	0.95	100.00	0.183
31	21-15-3	ISRLU	CEE	500	0.01	0.95	100.00	0.183
33	21-12-3	ISRLU	CEE	300	0.01	0.95	100.00	0.183
240	21-12-3	ISRLU	SSE	500	0.01	0.95	100.00	0.183

TABLE 1
Top Models

APPENDIX A
FIGURES & TABLES

List of figures and tables

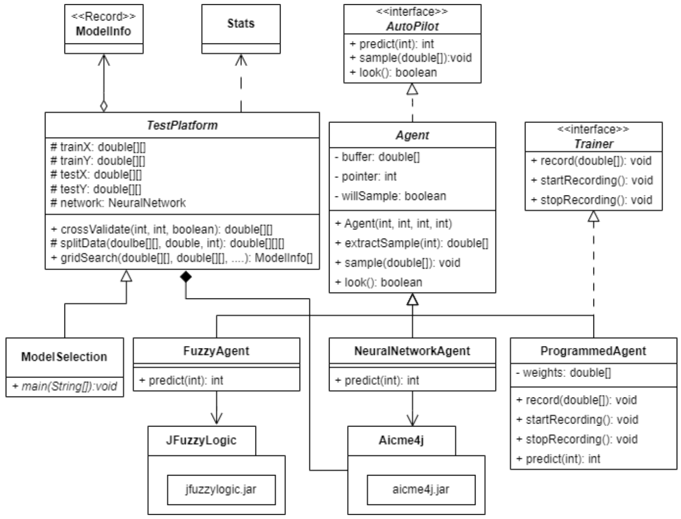


Fig. 4. UML Diagram

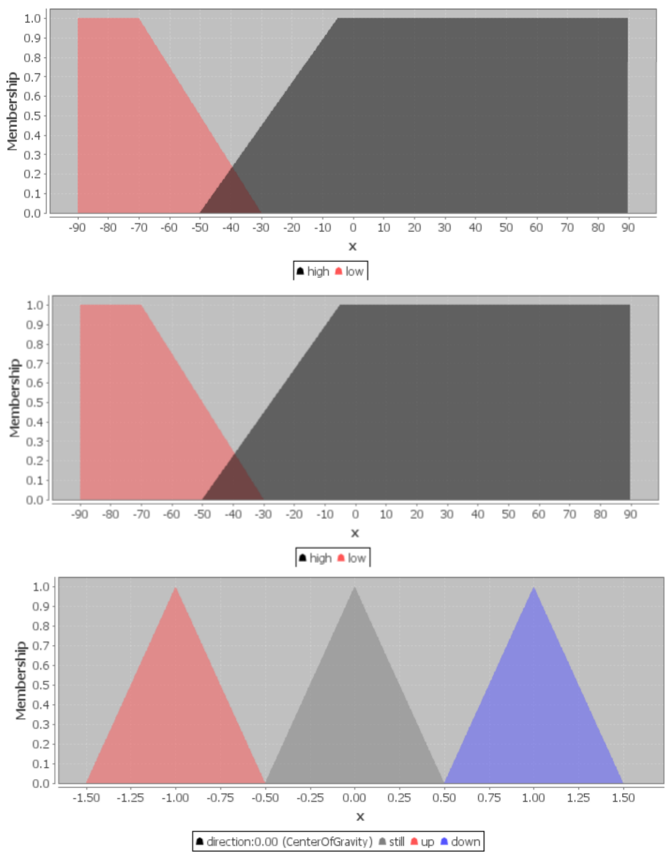


Fig. 5. Membership Functions