REPORT

Title: - "Machine Learning Based digital logic synthesis".

Mentor's Name: -- Dr. Binod Kumar

Authors: -- Abhishek Jilowa (B20CS001), Aman Mithoriya (B20CS004) and Harsh Sharma (B20CS017).

Abstract: -

The purpose of logic synthesis is to find structural representations of Boolean functions with the least amount of latency and area possible. The implementation accurately replicates the function if the function is completely described. If the function isn't fully stated, the implementation must only be true on the care set. While most logic synthesis algorithms use SAT and Boolean approaches to precisely implement the care set, we look into learning in logic synthesis, seeking to exchange exactness for generality. This research is directly related to machine learning, in which the training set is the care set and the implementation is anticipated to generalise to a validation set. The purpose is to train 100 functions using a set of care minterms and then test the implementation with a set of validation minterms taken from the same function. We make this benchmark suite available and provide a full comparison of the various learning methodologies.

Introduction: -

The purpose of logic regression is to find a Boolean model with binary covariates that accurately predicts the response of an unknown system. It can be found in a variety of applications, including data analysis, reverse engineering, model verification, and testing. Different variants of the problem can be formulated depending on how the unknown system behaves and how the Boolean model is represented. The unknown system is formulated as a completely described Boolean function f: B |I| B |O|, In the Logic Regression on High Dimensional Boolean Space, as a black-box input-output (IO) relation generator with inputs I and outputs O. The Boolean space addressed has a high dimensionality, with |I| and |O| in the hundreds or thousands. The goal is to create a small circuit that accurately depicts the IO generator. Its importance in non-equivalence diagnosis, engineering change order, semantic diagnosis, data path recognition, and other verification tasks motivates the challenge.

Design Problem Formulation: -

We must create a simple Boolean logic circuit that matches the input-output relations of the given generator with greatest accuracy, given a black boxed input-output relation generator and its knowledge of input/output variables.

Possible Methods to solve the problem: -

1) A multi-layer perceptron (MLP), a binary decision tree (BDT) with functional decomposition, and an RF are all included.

2) During synthesis, a solution based on multi-level ensemble feature selection, recommendationnetwork-based model training, subspace-expansion-based prediction, and accuracy-node joint exploration.

3) Methods based on decision trees and neural networks (NN) are another option. Multiple models are trained for each benchmark, with three being chosen for the ensemble.

4) Another option is to use tree-based machine learning models to convert tree nodes to SOP (sum of products) words. The model is either a decision tree with infinite depth or a 125-tree extreme gradient boosting (Boost) model with a maximum depth of five trees.

Methodology adopted for the project: -

We have decided to adopt the following methodologies to solve the problem: -

1) Machine Learning Model Ensemble.

2) Decision Tree for Logic Learning

3) Random Forest

Justification of the design choices: -

1)Since problem is based on classification (For example if there are 10 inputs and 2 outputs then we need to classify the 2 output for different set of input as 1 or 0 based on the training we do on 10 inputs). Random Forest is very useful technique to classify the objects, so it could be used here.

2)Since the scale of the input dataset is very large, we cannot create one or two trees but need to create hundreds of trees. This concept is known as random forest. A random forest is made up of numerous different decision trees, which also can be a useful tool to approach this problem.

3) *Decision Tree/Random Forest* are easy to convert into SOP expressions.

Result: -

S. No.	No. of Inputs	No. of Outputs	Data Size	Accuracy(%)
1.	26	4	300	97
2.	37	2	500	92
3.	44	5	500	91
4.	32	1	6400	82.6

Conclusion: -

- Boolean functions can be learned by Random Forest based methods.
- We could obtain SOP efficiently from Random Forest.
- We were able to achieve accuracy up to 92 percent for medium sized datasets and for large datasets we were able to achieve accuracy up to 85 percent (Accuracy may vary with parameters).
- We explored the connection between logic synthesis of incompletely specified functions and supervised learning.
- We were able to achieve the desired logic circuit.

References: -

https://arxiv.org/pdf/2012.02530.pdf https://scikit-learn.org/stable/modules/tree.html

https://link.springer.com/content/pdf/10.1007/s10115-021-01565-5.pdf https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html

https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html

https://scikit-learn.org/stable/auto_examples/tree/plot_unveil_tree_structure.html