Theory of Computation Project Details

Abhinav Nakarmi and Kuber Shahi

Project Topic: *Neural Network for Synthesizing Deterministic Finite Automata* (link).

Project Description: In theory of computation, deterministic finite automata (DFA) is an abstract mathematical model that is useful in solving various problems such as pattern matching and lexical analysis. Its application is also widely found in control systems such as in elevators, traffic light among other things. Researchers have been trying to find efficient methods for automating the generation of DFAs. Some available methods include SAT-solvers, optimization methods and meta-heuristics [1,2,3].

Interestingly, there have also been some research done on synthesizing deterministic finite automata through Machine Learning. Subsequently, for our project, we have chosen a research paper that aims to construct a deterministic finite automata according to a set of words belonging to some formal language using a recurrent neural network (RNN).

Therefore, the problem statement of the project is the following:

- **Input**: a Regular Language L to generate the list of tuples (w, ans) for training where ans = 1 if $w \in L$, and ans = 0 if $w \notin L$.

  For example: for language L = { $w \in \sum^* | \sum =$ { a, b} and w has no same neighboring characters}, then list = [(abab, 1), (abaa, 0),...]

- **Output**: DFA M such that L(M) = L

Project Goals: The primary goal of our project is to understand, explain and implement a working model of a neural network that can synthesize deterministic finite automata for a given formal language. In detail, our goals are:

- To understand how neural networks (specifically RNNs) are used in building DFAs.
- To understand the novel architecture based on RNN proposed by the paper in synthesizing DFAs correctly and quickly.
- To implement the proposed RNN model and achieve the results obtained by the paper
- To get an idea about different RNN models used in synthesizing DFAs mentioned in the paper.

Plans towards goals: We aim to achieve the aforementioned goals by

- Getting necessary knowledge about Recurrent Neural Networks, first-order network and second-order network.

- Breaking down the architecture of the proposed neural network and understand each section thoroughly. The sections include input layer, transition layer, normalization, output layer, adder and output neuron.
- Implementing basic Recurrent Neural Network and subsequently implement proposed neural network for synthesizing DFA.

Technical Details: We will implement the RNN model shown in the figure below (Fig 1) in Jupyter Notebook (Python). Since the model is a new one based on RNN, we won't be able to use any of the machine learning packages and have to code the proposed model from scratch. After building the network, we will also build the training data as defined by the paper and train our model on it.

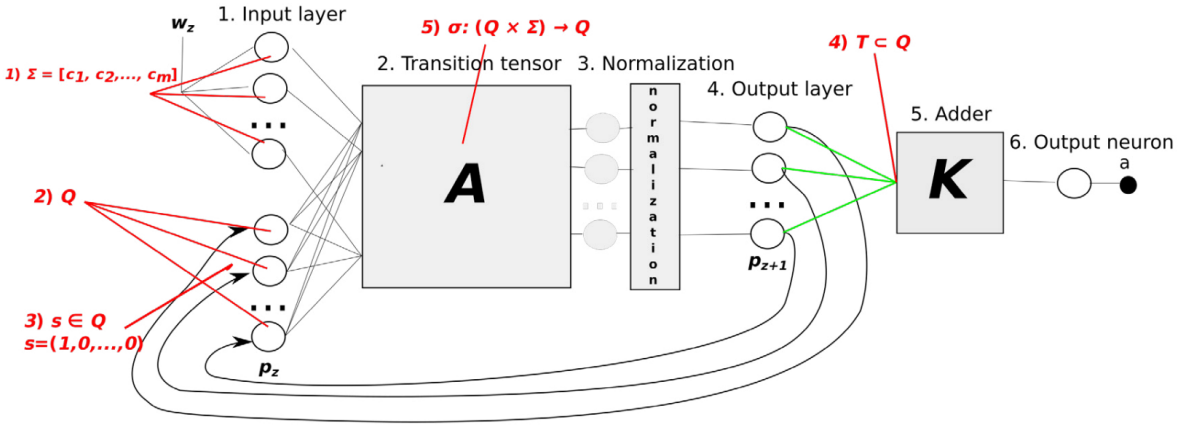

Fig. 1: Summary scheme of NN with marked parameters of DFA

To test the model, we will test the model on words of some alphabets as done by the paper, and compare the obtained results to validate our model.

References:

[1] Ulyantsev and F. Tsarev, "Extended finite-state machine induction using sat-solver," in Machine Learning and Applications and Workshops (ICMLA), 2011 10th International Conference on, vol. 2. IEEE, 2011, pp. 346–349.

[2] Chivilikhin, A. Shalyto, and V. Vyatkin, "Inferring automata logic from manual control scenarios: Implementation in function blocks," in Trustcom/BigDataSE/ISPA, 2015 IEEE, vol. 3. IEEE, 2015, pp. 307–312.

[3] K. Meinke, "Cge: A sequential learning algorithm for mealy automata." in ICGI, vol. 10. Springer, 2010, pp. 148–162.