Abstract

Parsing with finite automata networks implies, in one way, the conversion of a regular expression into a minimal deterministic finite automaton, while parsing with neural networks involves parsing of a natural language sentence. In ‘Parsing with finite automata networks’ finite automata are frequently combined using a set of rules for various operations like union, concatenation, and kleene closure; while in ‘Parsing with neural networks’ an incremental tree is
usually obtained, by using a set of rules for connecting a possible parse tree to the previously obtained incremental tree. Apparently, all the above rules that are being applied in parsing whether with finite automata networks or with neural networks belong to some graph transformation rules. These rules depict a new concerned area of grammars known as graph grammar, that is, a grammar that operates on graphs. This research paper presents a twofold investigation on the use of graph grammar as it explores an attempt to use both aspects of graph grammars (to generate a valid language and to parse a language for its validity) for parsing with (i) neural networks and (ii) finite automata networks.

Reference

- Driska, M., M. Makula, and u. Beňušková [2007]. “Organization of the state space of a simple recurrent network before and after training on recursive linguistic structures”. Neural Networks. vol. 20, no. 2, pp. 236-244.
- Thompson, K. [1968]. “Regular expression search algorithms”. Communications of the ACM. vol. 11, no. 6, pp. 419-422.

**Index Terms**

Computer Science  
Computational Linguistics

**Key words**

Parsing  
Graph grammar  
Regular expression  
Natural language

processing