

# Laporan Automata

---

**Nama: Asep Haryana Saputra**

**Kelas: TINFC-2023-04**

**NIM: 20230810043**

## I. Pendahuluan

---

Proyek ini adalah tugas mata kuliah Teori Automata yang bertujuan untuk mengimplementasikan konsep-konsep automata dan teori compiler dalam sebuah Domain Specific Language (DSL) untuk konfigurasi Machine Learning. Proyek ini menggabungkan pemahaman mendalam tentang finite automata, context-free grammar, dan implementasi lexer-parser untuk membuat alat yang dapat mengubah sintaks sederhana menjadi kode Python yang production-ready.

Dengan menggunakan SimpleNeural-DSL, pengguna dapat mendefinisikan arsitektur neural network dengan cara yang deklaratif dan mudah dipahami, tanpa perlu menguasai kompleksitas TensorFlow secara mendalam. Proyek ini mendemonstrasikan penerapan langsung teori automata dalam aplikasi praktis.

## II. Deskripsi Proyek

---

SimpleNeural-DSL adalah Domain Specific Language yang dirancang khusus untuk konfigurasi dan pembuatan model Machine Learning. Sistem ini terdiri dari beberapa komponen utama yang saling terintegrasi:

1. Lexer (Tokenizer) - Menganalisis kode input dan mengidentifikasi token-token berdasarkan regular expressions dan finite automata.
2. Parser - Menganalisis urutan token dan membangun Abstract Syntax Tree (AST) menggunakan teknik recursive descent parsing.

3. Semantic Analyzer - Melakukan validasi makna, type checking, dan symbol table management.

4. Code Generator - Menghasilkan kode Python yang menggunakan TensorFlow/Keras dari representasi AST.

### **III. Arsitektur Sistem**

---

Sistem SimpleNeural-DSL dibangun dengan arsitektur yang modular dan mengikuti pola pipeline compiler tradisional:

INPUT (File .sndsl)

↓

LEXER - Tokenization menggunakan DFA

↓

PARSER - Syntax Analysis menggunakan Context-Free Grammar

↓

SEMANTIC ANALYZER - Type Checking dan Validation

↓

CODE GENERATOR - Python Code Generation

↓

OUTPUT (File .py dengan TensorFlow/Keras)

## **IV. Implementasi Automata**

---

### **A. Lexer (DFA Implementation)**

Lexer mengimplementasikan Deterministic Finite Automata (DFA) untuk mengenali token. Sistem ini dapat mengenali lebih dari 30 jenis token termasuk: keywords (model, layer, optimizer, train), identifiers, numbers, strings, dan operators.

State transitions dalam DFA meliputi:

- START state: state awal
- IDENTIFIER state: untuk mengenali nama variabel dan keyword
- NUMBER state: untuk mengenali literal angka
- STRING state: untuk mengenali string literals
- ACCEPT state: state penerimaan
- ERROR state: untuk input yang tidak valid

### **B. Parser (CFG Implementation)**

Parser mengimplementasikan grammar Context-Free dengan production rules yang mendefinisikan struktur program DSL. Menggunakan recursive descent

parsing untuk menganalisis urutan token dan membangun AST yang merepresentasikan struktur program.

### C. Semantic Analysis

Semantic analyzer memeriksa: type compatibility, range validation, symbol table consistency, dan layer configuration rules. Setiap parameter divalidasi sesuai dengan constraint yang telah didefinisikan.

## V. Pengujian dan Hasil

---

Pengujian dilakukan pada beberapa level:

1. Unit Testing - Menguji setiap komponen (lexer, parser, semantic analyzer) secara terpisah
2. Integration Testing - Menguji alur lengkap dari input DSL hingga code generation
3. End-to-End Testing - Mengeksekusi generated code dan memverifikasi bahwa model dapat di-train dengan baik

Hasil pengujian menunjukkan bahwa sistem berhasil:

- ✓ Melakukan tokenization dengan akurat
- ✓ Mem-parse berbagai variasi sintaks DSL
- ✓ Mendeteksi dan melaporkan error dengan pesan yang informatif
- ✓ Menghasilkan kode Python yang dapat dieksekusi
- ✓ Menghasilkan model yang dapat melakukan training dengan TensorFlow/Keras

## **VI. Kesimpulan**

---

Proyek SimpleNeural-DSL berhasil mengimplementasikan konsep-konsep fundamental dari teori automata dan compiler design dalam aplikasi praktis. Sistem ini mendemonstrasikan penggunaan Finite Automata untuk lexical analysis, Context-Free Grammar untuk syntax analysis, dan teknik-teknik compiler modern untuk semantic analysis dan code generation.

Hasil implementasi menunjukkan bahwa kombinasi antara teori automata yang kuat dan engineering practices yang baik dapat menghasilkan tools yang powerful dan user-friendly. SimpleNeural-DSL membuktikan bahwa dengan abstraksi yang tepat, pengguna domain dapat menulis model ML yang kompleks dengan sintaks yang sederhana dan intuitif.