Original Paper Pipeline[Early RTL Delay Prediction Using Neural Networks]

**1. RTL Design Generation (MetaRTL HW Generator)**

* Random RTL circuits are generated using **MetaRTL** based on:
  + Primitive component types (logic, mux, demux, branch, arithmetic)
  + Configurable bit widths, number of inputs, and connections

**2. RTL → Gate-Level Netlist (Yosys)**

* The Verilog RTL designs are synthesized using **Yosys** (open-source logic synthesis tool).
* Output: **Gate-level netlist** mapped to 40nm CMOS cells.

**3. Static Timing Analysis (OpenSTA)**

* The synthesized netlist is analyzed with **OpenSTA** to extract:
  + Pin-to-pin **slew (rise/fall)**
  + Pin-to-pin **delay (rise/fall)**
* Timing arcs include:
  + Register-to-Register
  + Input-to-Output
  + Register-to-Output
  + Input-to-Register

**4. Feature Extraction & Dataset Construction**

* For each **pin-to-pin connection**:
  + Extract **features** like:
    - Component type (one-hot or embedded)
    - Bit widths (input/output)
    - Number of inputs/outputs
    - Fanout (input/output connections)
    - Slew-in
    - Pin index
  + Extract **labels** (targets):
    - Slew out (rise/fall)
    - Delay (rise/fall)
* Create two datasets:
  + **DS-I**: Logic & MUX only
  + **DS-II**: All 37 primitives (logic + mux + arithmetic + branch)

**5. Model Training (MLP / TabMLP)**

* Two architectures:
  + **MLP**: Uses one-hot encoded component type
  + **TabMLP**: Uses embedded component type + numeric features
* Objective: **Multi-output regression** for:
  + Slew rise
  + Slew fall
  + Delay rise
  + Delay fall
* Use **TPE** for hyperparameter tuning

**6. Slew Propagation Algorithm**

* For components not connected to primary inputs:
  + Use **predicted output slew** of previous gate as **input slew**
* Predict **sequentially** from input gates to deeper logic
* Slew propagation affects delay prediction accuracy

**7. Model Evaluation**

* Evaluate on 4-bit adders (BKA, KSA, SKA)
* Metrics used:
  + **R² score**
  + **MAE**, **MSE**
* Compare predictions with OpenSTA results
* Test generalization to unknown input slew values

**8. Runtime Benchmark**

* Compare runtime of NN models vs. full Yosys + OpenSTA flow
* ML models are **~8.4× faster** than synthesis + STA tools

**9. Baseline Comparison**

* Compare against:
  + Random Forest
  + Decision Tree
* MLP and TabMLP outperform shallow models