



香港中文大學

The Chinese University of Hong Kong

Large Language Model in EDA

Bei Yu

Department of Computer Science & Engineering

Chinese University of Hong Kong

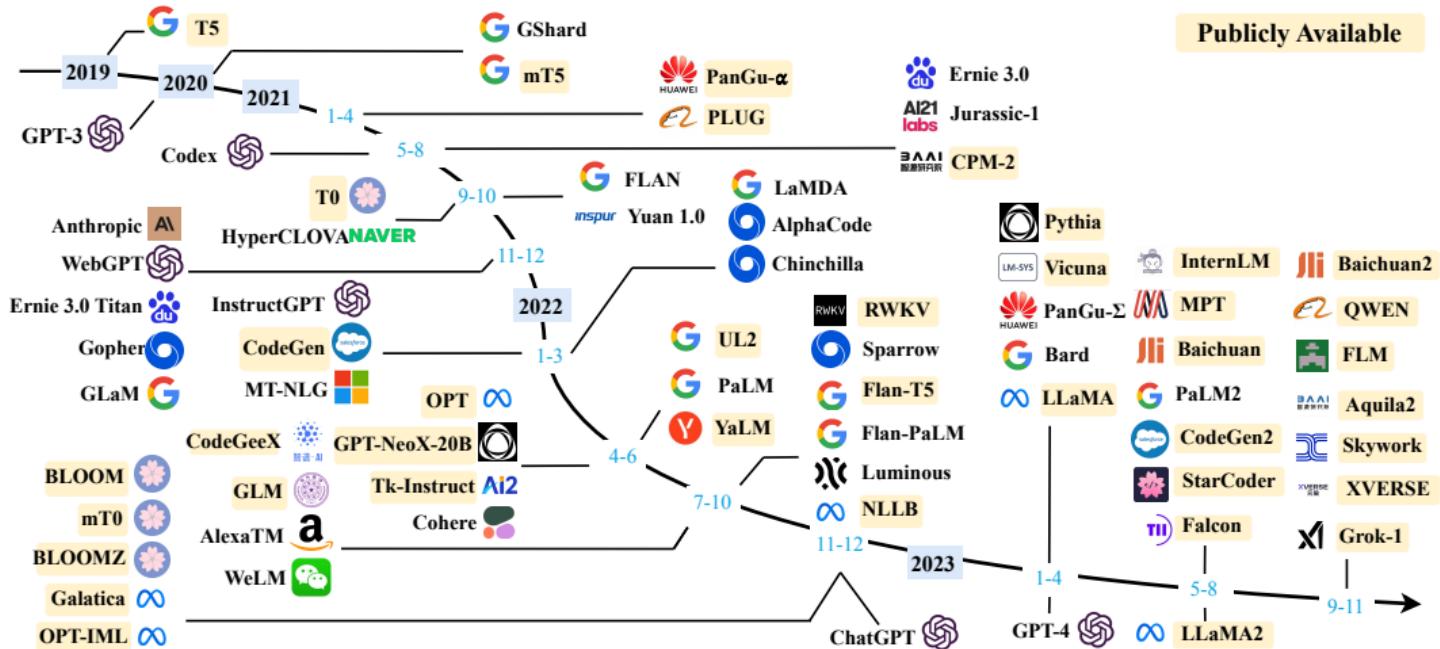
byu@cse.cuhk.edu.hk





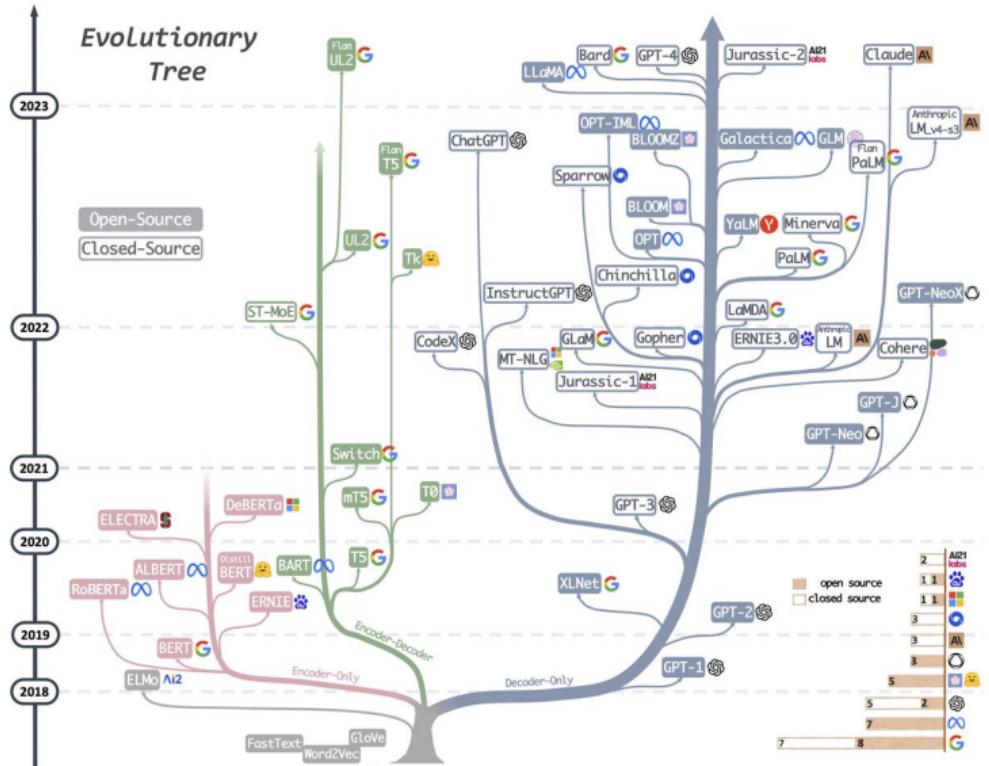
CLI Command Line Interface
GUI Graphical User Interface
CUI Conversational User Interface

LLM Development Timeline¹



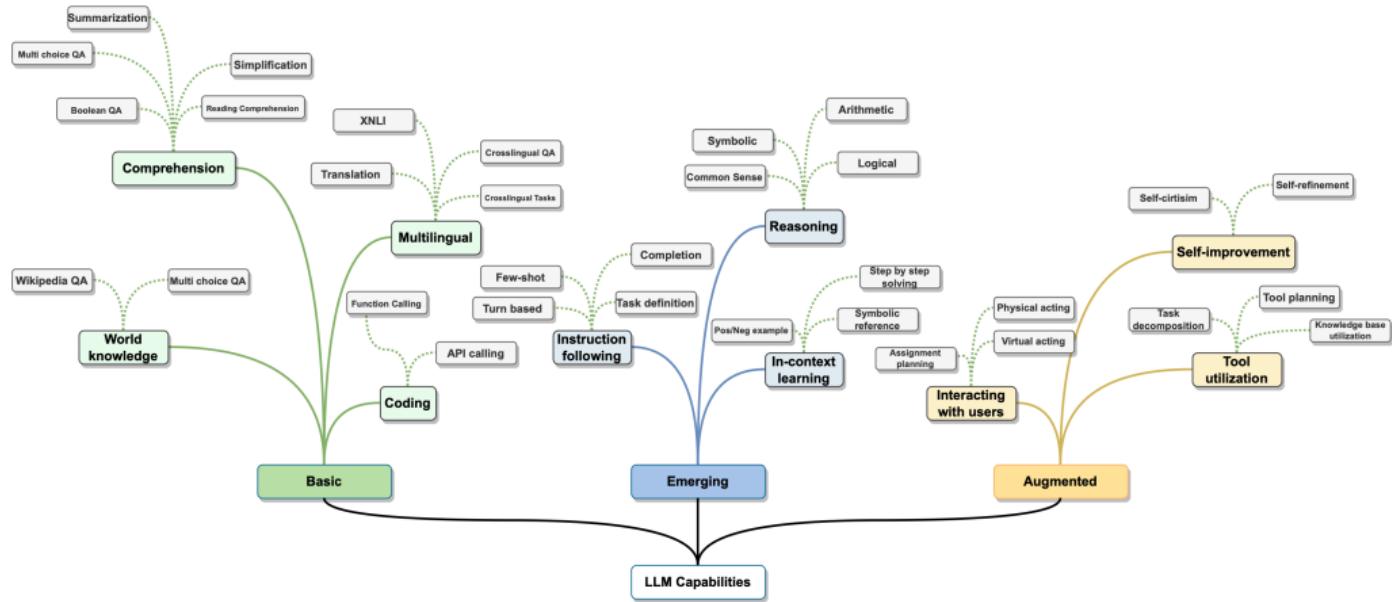
Wayne Xin Zhao et al. (2023). "A survey of large language models". In: *arXiv preprint arXiv:2303.18223*.

LLM Structure²



Jingfeng Yang et al. (2024). "Harnessing the power of LLM in practice: A survey on ChatGPT and beyond". In: ACM TKDD 18.6, pp. 1-32.

LLM Capabilities³





Challenges and Improvements

```
矩阵 a= tensor([[ 0.3348, -0.1575, -0.0552, 0.4606, -0.4005],  
[ 1.2785, -0.0594, -2.4832, -0.1052, -1.5892],  
[-0.3722, -0.1244, -0.6635, 0.6163, -0.8171],  
[-1.4294, -0.6973, -0.0049, -0.8030, -1.2963]]) 矩阵 tensor([[-1.6391, -1.8575],  
[ 1.4069, 0.8627],  
[ 0.4147, -0.0830],  
[ 0.7041, 0.1294],  
[-1.6289, 0.2686]])请计算a和b的矩阵乘法
```



矩阵 A 是一个 4x5 矩阵，矩阵 B 是一个 5x2 矩阵。由于矩阵 A 的列数（5）与矩阵 B 的行数（5）相等，所以这两个矩阵可以相乘。乘积矩阵的大小为 4x2。

矩阵相乘的方法是将第一个矩阵的每一行与第二个矩阵的每一列相乘，然后求和。以下是矩阵相乘的结果：

```
css Copy code  
[ [-0.1878, -0.6610],  
[ 1.5415, -3.5917],  
[ 0.3561, 0.1316],  
[-0.9432, 2.0052]  
]
```

Correct Ans:

$$\begin{bmatrix} 1.2766, & 0.0329 \\ 6.1799, & -2.6547 \\ -0.8886, & 0.1450 \\ 0.8300, & -0.1394 \end{bmatrix}$$



Challenges and Improvements

S

矩阵 a= tensor([[0.3348, -0.1575, -0.0552, 0.4606, -0.4005],
[1.2785, -0.0594, -2.4832, -0.1052, -1.5892],
[-0.3722, -0.1244, -0.6635, 0.6163, -0.8171],
[-1.4294, -0.6973, -0.0049, -0.8030, -1.2963]]) 矩阵 tensor([[-1.6391, -1.8575],
[1.4069, 0.8627],
[0.4147, -0.0830],
[0.7041, 0.1294],
[-1.6289, 0.2686]])请计算a和b的矩阵乘法

根据您提供的矩阵，计算矩阵 a 乘以矩阵 b 的矩阵乘法结果：

```
css
import torch

a = torch.tensor([[0.3348, -0.1575, -0.0552, 0.4606, -0.4005],
                 [1.2785, -0.0594, -2.4832, -0.1052, -1.5892],
                 [-0.3722, -0.1244, -0.6635, 0.6163, -0.8171],
                 [-1.4294, -0.6973, -0.0049, -0.8030, -1.2963]])

b = torch.tensor([[ -1.6391, -1.8575],
                 [ 1.4069, 0.8627],
                 [ 0.4147, -0.0830],
                 [ 0.7041, 0.1294],
                 [-1.6289, 0.2686]])

result = torch.matmul(a, b)
print(result)
```

Regenerate response

运行代码后，会得到矩阵乘法的结果。

Send a message.

Correct Ans:

$$\begin{bmatrix} 1.2766, & 0.0329 \\ 6.1799, & -2.6547 \\ -0.8886, & 0.1450 \\ 0.8300, & -0.1394 \end{bmatrix}$$

How Machine Can Understand Picture?



R-CNN: Region-Based CNN



Girshick et al, "Rich feature hierarchies for accurate object detection and semantic segmentation", CVPR 2014.
Figure copyright Ross Girshick, 2015; [source](#). Reproduced with permission.

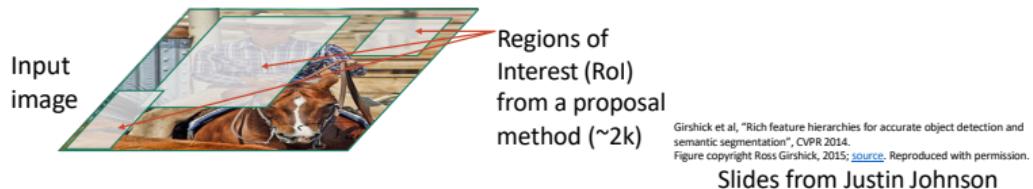
Slides from Justin Johnson

Machine Learning is to Fit A Function $f(x)$

How Machine Can Understand Picture?



R-CNN: Region-Based CNN

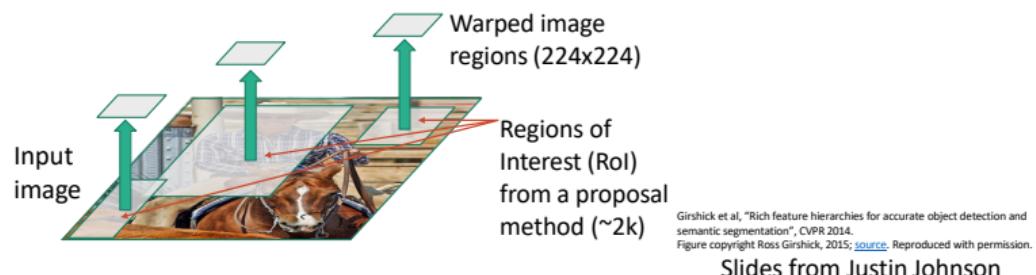


Machine Learning is to Fit A Function $f(x)$

How Machine Can Understand Picture?



R-CNN: Region-Based CNN



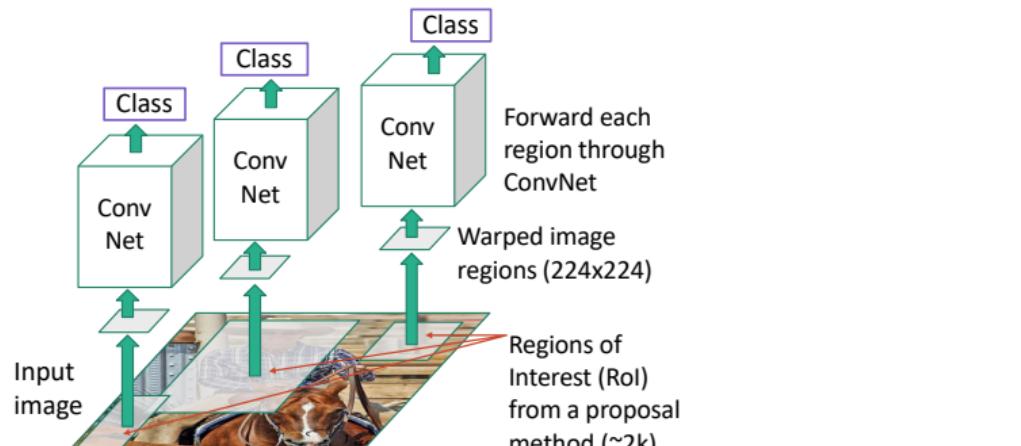
Machine Learning is to **Fit** A Function $f(x)$

How Machine Can Understand Picture?



R-CNN: Region-Based CNN

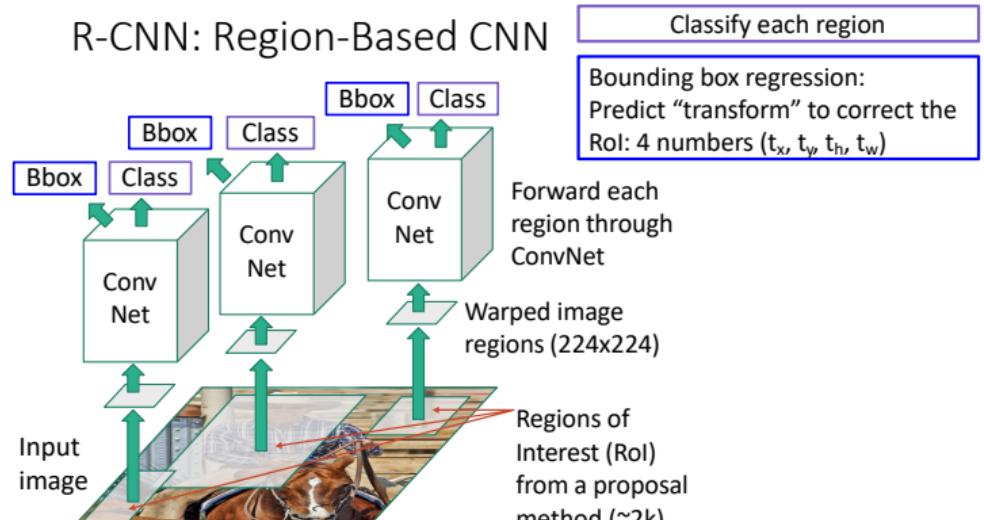
Classify each region



Slides from Justin Johnson

Machine Learning is to **Fit** A Function $f(x)$

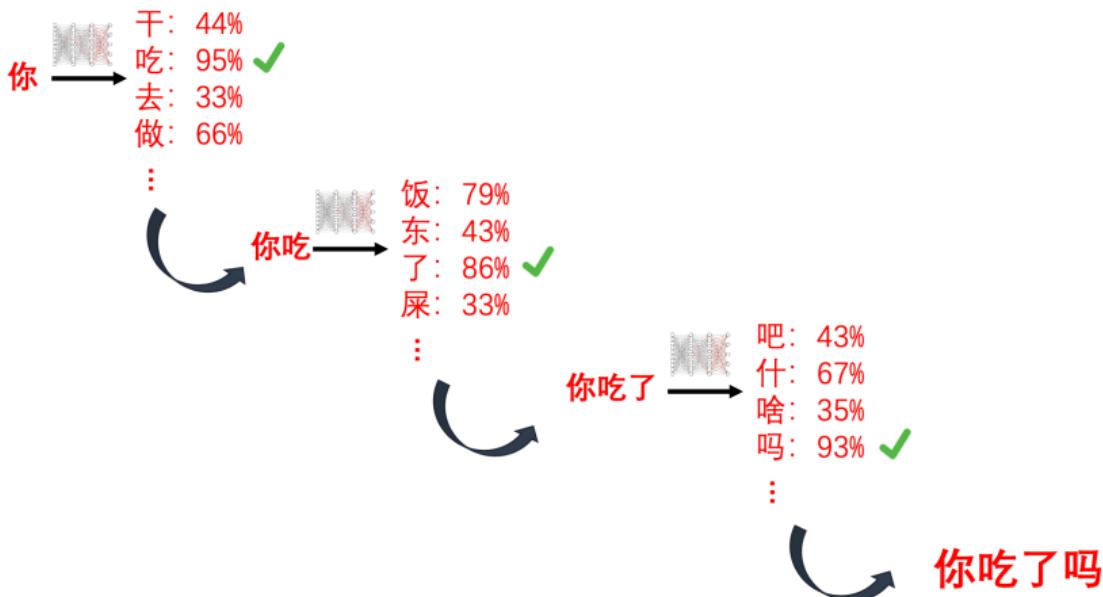
How Machine Can Understand Picture?



Slides from Justin Johnson

Machine Learning is to **Fit** A Function $f(x)$

How Machine Can Understand Text?

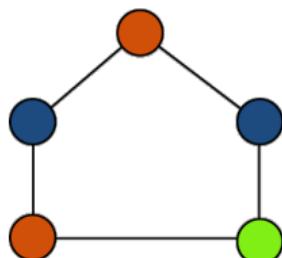


Machine Learning is to Fit A Function $f(x)$

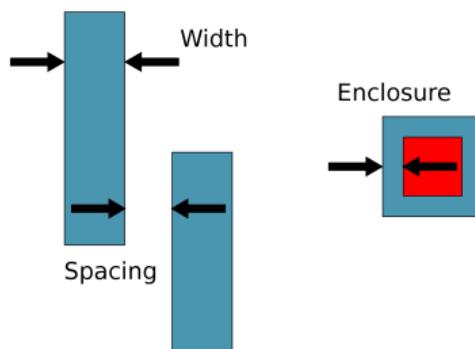


- Combinatorial Problem (e.g. Coloring)
- Handling Complicated Rules

The three basic DRC checks



(a) Graph Coloring

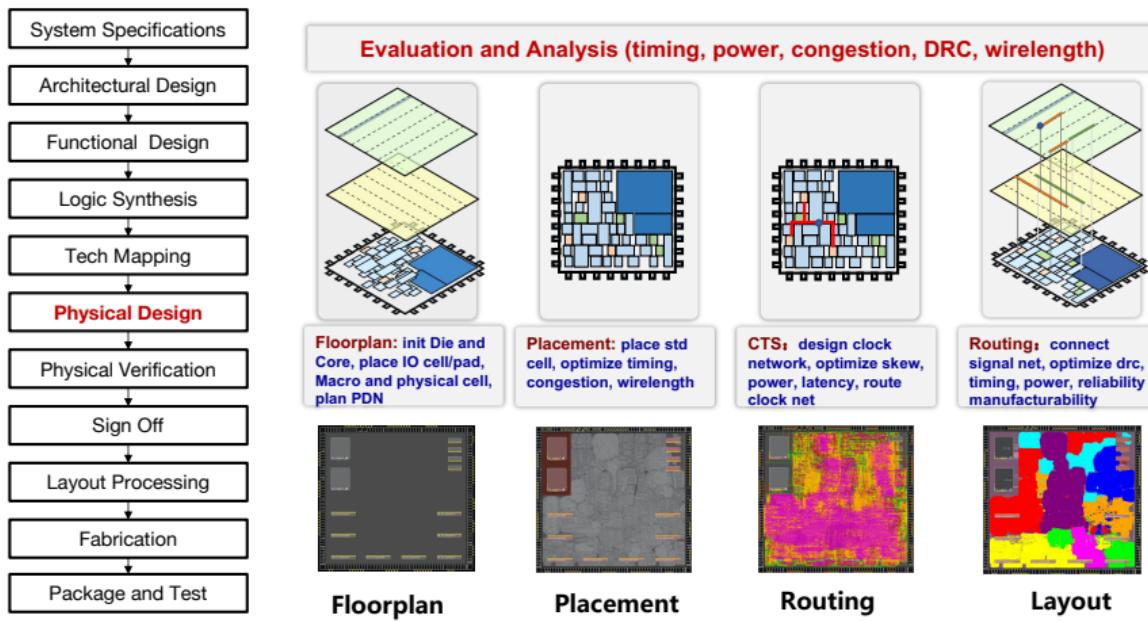


(b) Design Rule Checking

Complicated Circuit Design and EDA Flow



A typical EDA flow...⁴

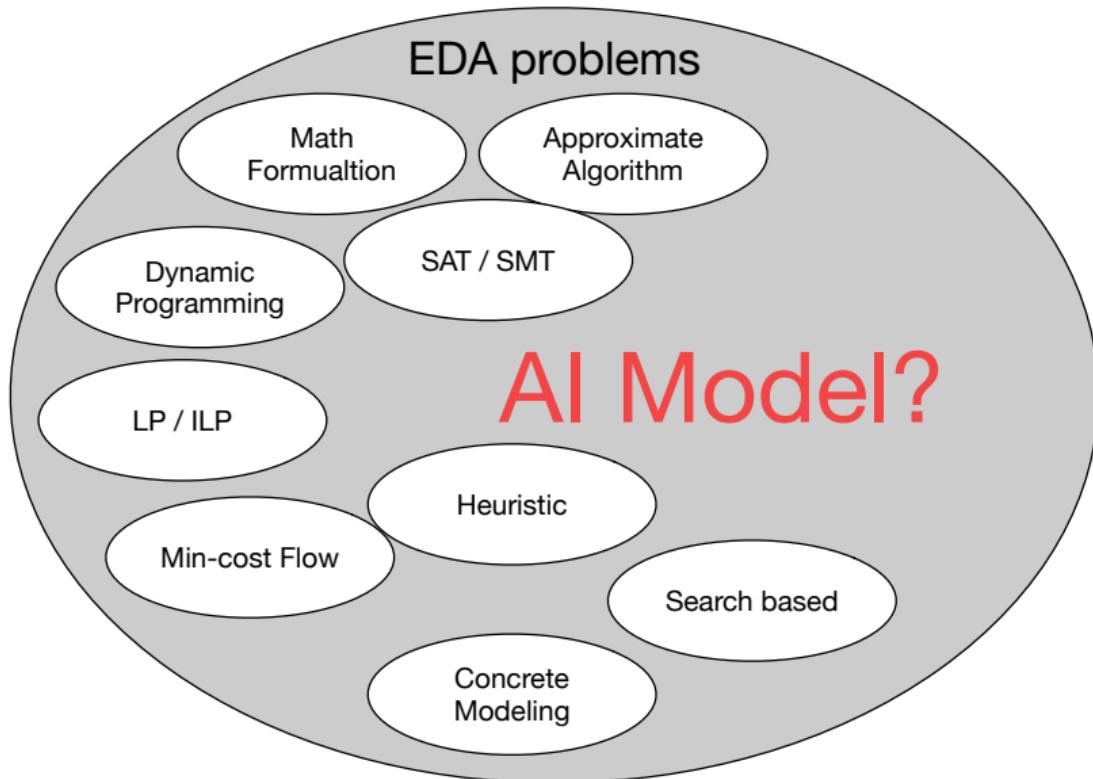


⁴Xingquan Li et al (2024). “iEDA: An Open-Source Intelligent Physical Implementation Toolkit and Library”. In: *Proc. ASPDAC*.



EDA problems

AI Model?



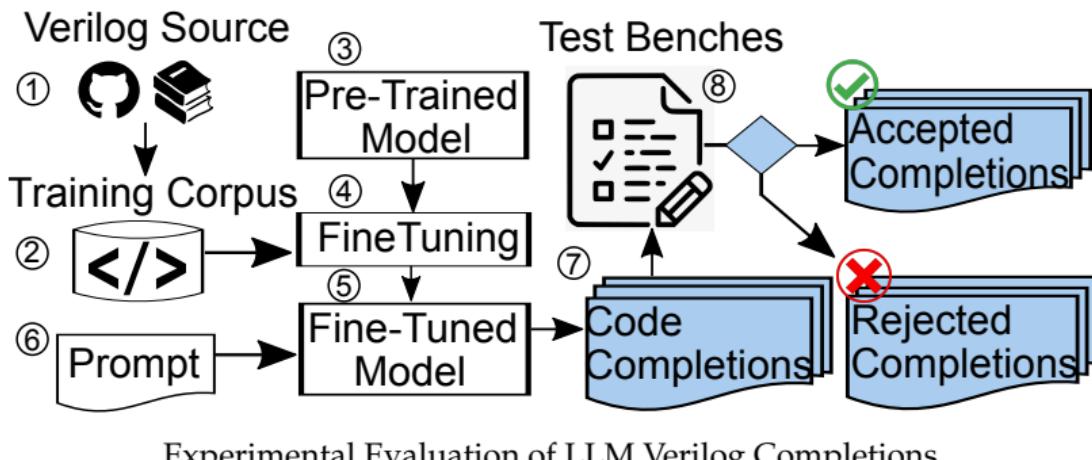


① RTL Generation and Debug

② Flow Script Generation

③ Documentation QA

RTL Generation and Debug





```
module example(
    input [31:0] x,
    output [31:0] y,
    output [31:0] z,
    output [31:0] w
);

assign y = 9 * x ;
assign z = 23 * x ;
assign w = 81 * x ;

endmodule
```

```
module example(
    input [31:0] x,
    output [31:0] y,
    output [31:0] z,
    output [31:0] w
);

assign y = 8 * x + x;
assign z = 32 * x - 9 * x;
assign w = 8 * (9 * x) + 9 * x;

endmodule
```



Before rewriting

```
module example(
    input wire s,
    input wire [31:0] A, B, C, D,
    output reg [32:0] Z
);

always @ (s, A, B, C, D) begin
    if (s)
        Z <= A + B;
    else
        Z <= C + D;
end

endmodule
```



Before rewriting

```
module example(
    input wire s,
    input wire [31:0] A, B, C, D,
    output reg [32:0] Z
);

reg [31:0] Z1, Z2; // Intermediate values

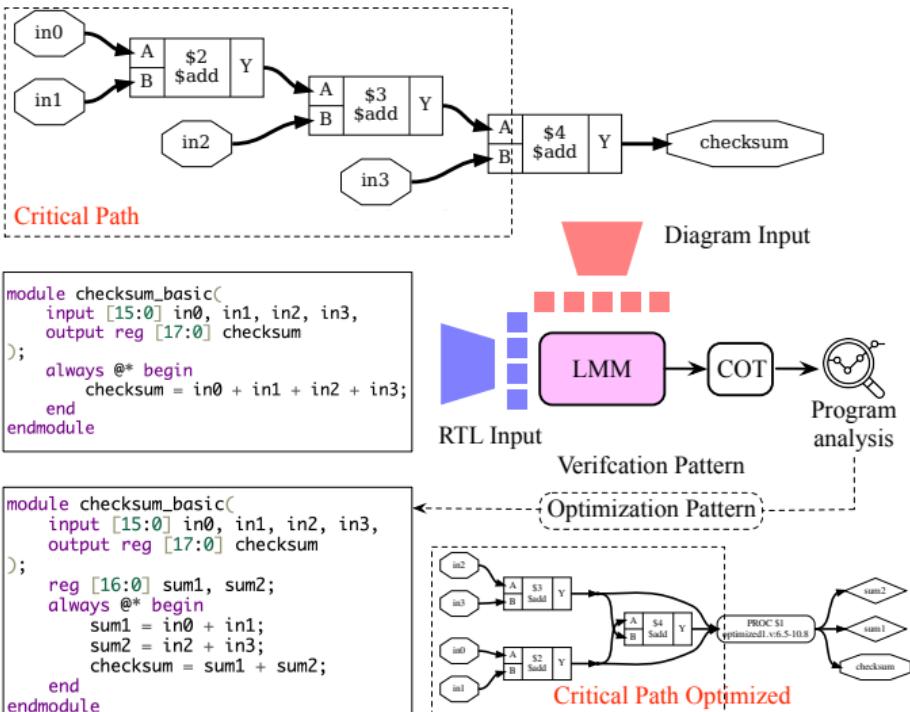
always @ (s, A, B, C, D) begin
    if (s)
        Z1 <= A;
    Z2 <= B;
end else begin
    Z1 <= C;
    Z2 <= D;
end
end

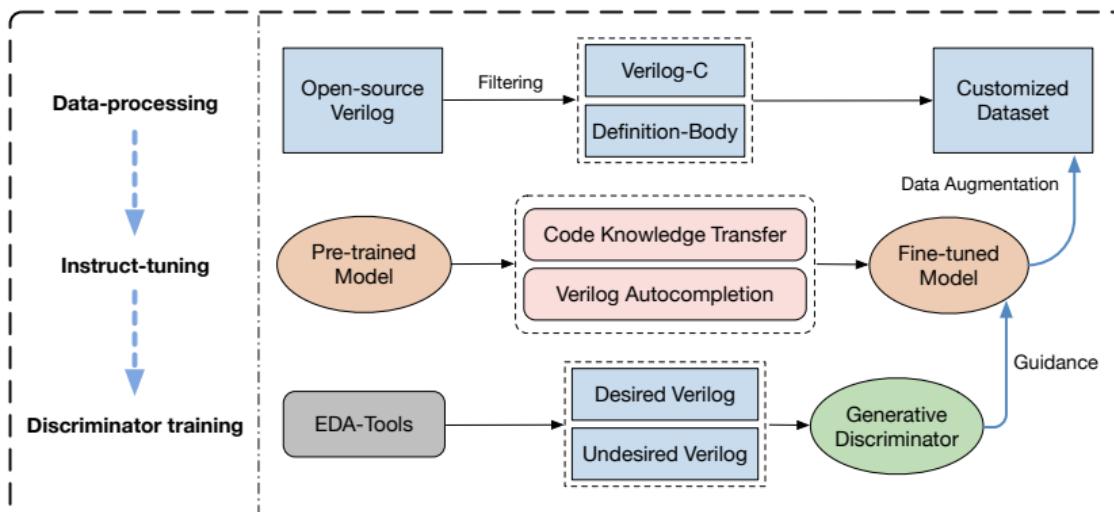
always @(Z1, Z2) begin
    Z <= Z1 + Z2;
end

endmodule
```

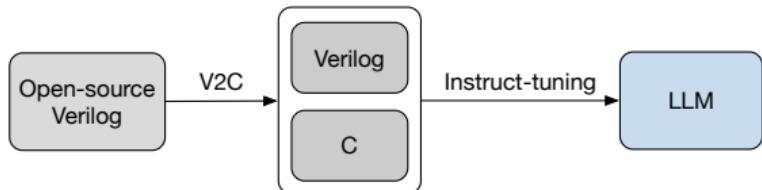
After rewriting

RTLRewrite: Multi-Modality





Align Verilog with C to generate self-supervised dataset.



- Tool **V2C** to convert the Verilog into C
- Verilog-C pairs are used for LLMs **instruct-tuning**
- Alignment from C to Verilog.

System Prompt:

You are an experienced Verilog engineer.

Instruction:

You are supposed to translate the following Verilog into C program.

```
module d_latch(
    input d,
    input en,
    input rstn,
    output reg q
);

    always @ (en or rstn or d)
        if (!rstn)
            q <= 0;
        else
            if (en)
                q <= d;
endmodule
```

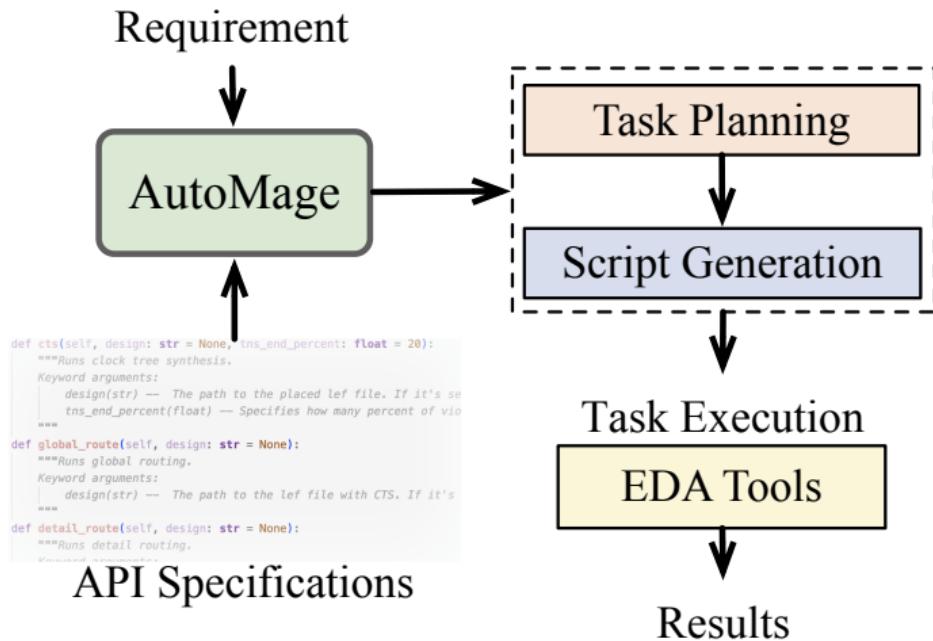
Answer:

```
#include <stdio.h>
#include <stdbool.h>

void d_latch(
    int d,
    bool en,
    bool rstn,
    int *q) {

    if (!rstn) {
        *q = 0;
    } else if (en) {
        *q = d;
    }
}
```

Flow Script Generation





Task Planning and Script Generation

#1: User Requirement

For the design named "aes" on the platform "asap7", please perform synthesis with a clock period of 5, followed by floorplan with a core utilization of 70%. Then, execute placement with a density of 0.8. Next, proceed with CTS to fix 40% of violating paths. Finally, evaluate the performance after routing using "power" metric.

#2: Task Decomposition

```
task1: set up the EDA tool
func: set_up()
args:
    design_name: "aes"
    platform: "asap7"
```

```
task2: perform synthesis
func: run_synthesis()
args:
    clock_period: 5
```

```
task3: execute floorplan
func: floorplan()
args:
    core_utilization: 70
```

```
task4: perform placement
func: placement()
args:
    density: 0.8
```

```
task5: perform CTS
func: cts()
args:
    tns_end_percent: 40
```

```
task6: perform routing
func: global_route()
detail_route()
```

```
task7: evaluation
func: get_metric()
args:
    stage: "route"
    metrics: ["power"]
```

#3: Script Generation

```
# Initialize
eda = chateda()
```

```
# Set up the EDA tool
eda.setup(design_name="aes", platform="asap7")
```

```
# Perform synthesis
eda.run_synthesis(clock_period=5)
```

```
# Execute floorplan
eda.floorplan(core_utilization=70)
```

```
# Perform placement
eda.placement(density=0.8)
```

```
# Perform CTS
eda.cts(tns_end_percent=40)
```

```
# Perform routing
eda.global_route()
eda.detail_route()
```

```
# Evaluate the performance after routing
Performance = eda.get_metric("route", ["power"])
```



Our defined API specifications

```

class chateda:
    def __init__(self) -> None:
        pass

    # Setup
    def setup(
        self,
        str: str,
        design_name: str,
        platform: str,
        flow_home: str = ".",
        verilog: str = None,
        sdc: str = None,
    ):
        """Setup EDA tool.
        Keyword arguments:
        design_name -- The name of the top-level module of the design.
        platform(str) -- Specifies process design kit or technology node to be used. Supported options are "asap7", "nanangate45", "sky130", and "gtlf".
        flow_home(str) -- The path to the flow home directory.
        verilog(str) -- The path to the design Verilog files.
        sdc(str) -- The path to design constraint (SDC) file.
        pass
    """
    pass

    # Synthesis
    def run_synthesis(self, clock_period: int = None, abc_area: bool = False):
        """Runs synthesis.
        Logic synthesis can't be executed without setting up.
        Keyword arguments:
        clock_period -- Clock period to be used by STA during synthesis. Default value read from constraint.sdc
        abc_area(bool) -- Strategies for Yosys ABC synthesis: Area/Speed. Default ABC_SPEED.
        pass
    """
    pass

    # Generate the gate netlist file in the default path.
    pass

    # Floorplan
    def floorplant(
        self,
        str: None,
        netlist: str = None,
        core_UTILIZATIONS: float = None,
        core_MARGIN: float = None,
        core_MARGININT: int = None,
        macro_PLACE_HALO: int = None,
        macro_PLACE_CHANNEL: int = None,
    ):
        """Runs floorplan.
        Floorplan can't be executed without logic synthesis.
        Keyword arguments:
        netlist(str) -- Path to gate-level netlist. If it's set to None, the gate-level netlist file will be read in the default path.
        core_UTILIZATIONS -- Core utilization percentage (0-100).
        core_ASPECT_RATIO(float) -- The core aspect ratio (height / width).
        core_MARGIN(int) -- The margin between the core area and die area, in multiples of SITE heights. The margin is applied to each side.
        macro_PLACE_HALO(int) -- horizontal/vertical halo around macros (microns). Used by automatic macro placement.
        macro_PLACE_CHANNEL(int) -- horizontal/vertical channel width between macros (microns). Used by automatic macro placement when RTLMP_FLOW is disabled. Imagine channel=18 and halos=6. Then macros must be 38 apart but standard cells must be 5 away from a macro.
        pass
    """
    pass

    # Place
    def placementself, design: str = None, density: float = 0.7):
        """Runs placement.
        Placement can't be executed without floorplanning.
        Keyword arguments:
        design(str) -- Path the floorplanned lef file. If it's set to None, the floorplanned lef file will be read in the default path.
        density -- The desired placement density of cells. It reflects how spread the cells would be on the core area. 1.0 = closely dense, 0.0 = widely spread.
        pass
    """
    pass

    # Generate the placed lef file in the default path.
    pass

```

```

    # CTS
    def cts(self, design: str = None, tns_end_percent: float = 20):
        """Runs clock tree synthesis.
        CTS can't be executed without placement.
        Keyword arguments:
        design(str) -- The path to the placed lef file. If it's set to None, the placed lef file will be read in the default path.
        tns_end_percent(float) -- Specifies how many percent of violating paths to fix [0-100]. Worst path will always be fixed
        pass
    """
    pass

    # Route
    def global_route(self, design: str = None):
        """Runs global routing.
        Global routing can't be executed without CTS.
        Keyword arguments:
        design(str) -- The path to the lef file with CTS. If it's set to None, the lef file with CTS will be read in the default path.
        pass
    """
    pass

    # Generate the global routed lef file in the default path.
    def detail_route(self, design: str = None):
        """Runs detail routing.
        Detail routing can't be executed without global routing.
        Keyword arguments:
        design(str) -- The path to the global routed lef file. If it's set to None, the global routed lef file will be read in the default path.
        pass
    """
    pass

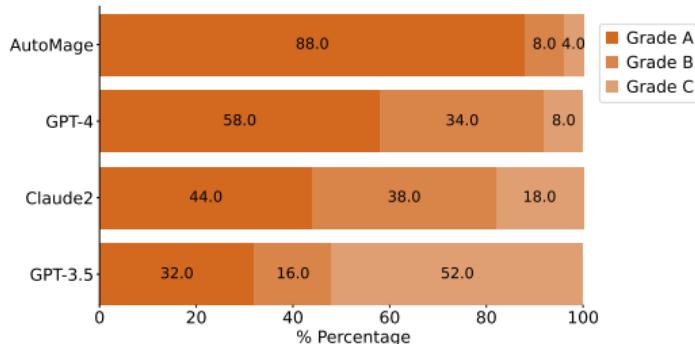
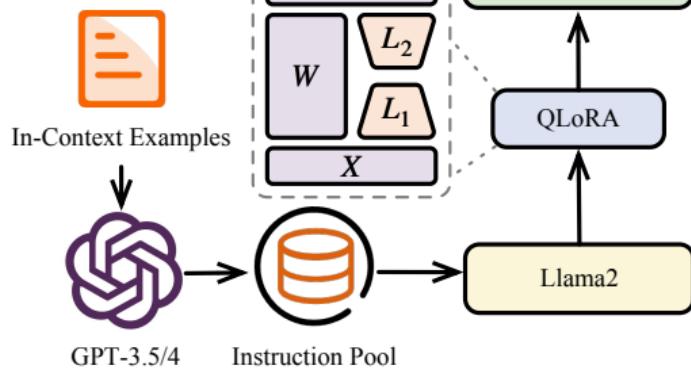
    # Finishing
    def report(self):
        """Runs final report.
        Final report can't be executed without density fill.
        pass
    """
    pass

    # Get metric
    def get_metric(self, stage: str, metrics: list):
        """Get metric of a stage.
        Get metric can't be executed before the provide stage has been executed.
        Keyword arguments:
        stage(str) -- The stage to get the metric from.
        Available values are: "floorplant", "place", "cts", "route", "final".
        metrics(list) -- The metrics to be measured. They should be provided in a list.
        Available values are: "tns", "wns", "area", "power".
        Returns:
        performance(float) -- The performance value of metrics. The smaller the better.
        pass
    """
    pass

    # Tuning
    def tunefunc, param:
        """Runs parameter tuning for chateda.
        Keyword arguments:
        param -- A function that runs the target flow and return a metric for parameter tuning.
        The function should return the concerned results as its function parameters.
        The function should return the concerned results as its return value.
        param -- The flow parameters to be tuned.
        param should be a dictionary with the following format:
        {param_name: {"min": min, "max": max, "step": step}}
        pass
    """
    pass

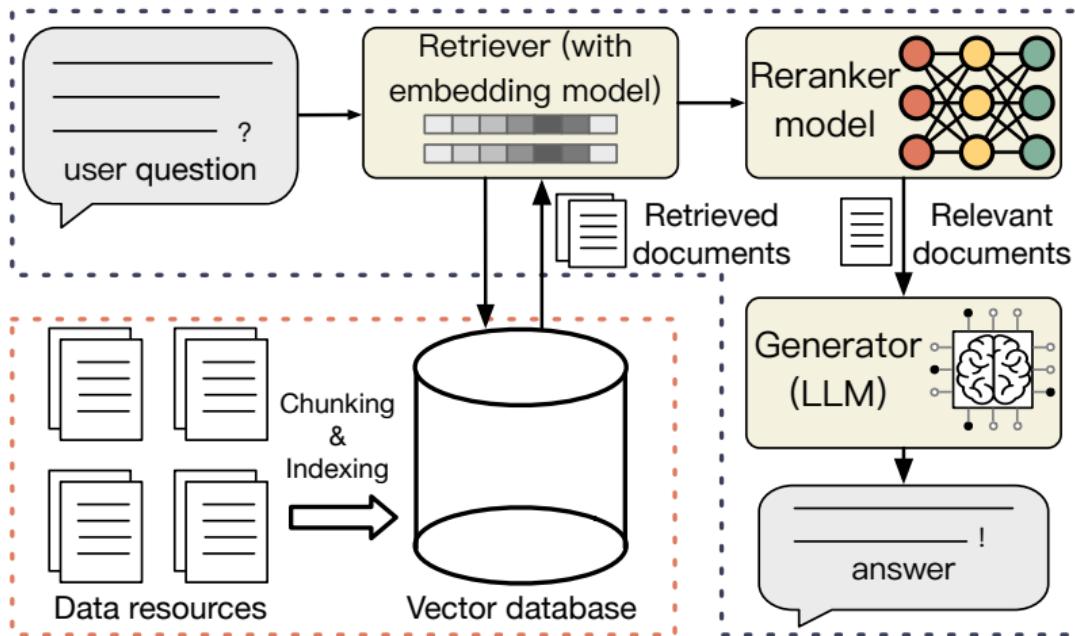
```

Instruction Tuning & Evaluation

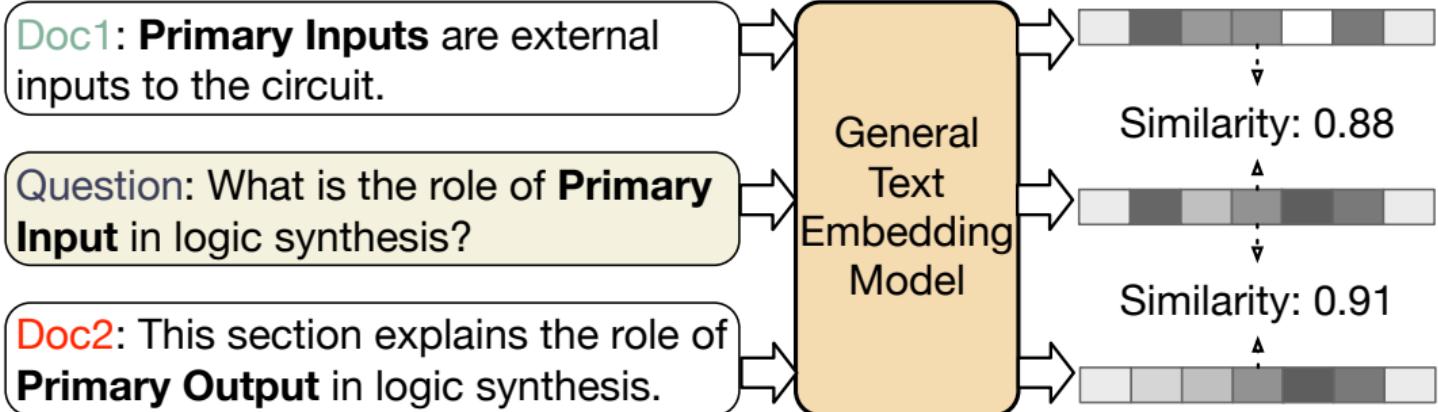


- **Grade A:** correct in planning and script
- **Grade B:** correct **only** in planning
- **Grade C:** fault in both

Documentation QA



Text Embedding Model: Failure Case



Text Embedding Model: Contrastive Fine-tuning



Rationale: improve semantic understanding for domain terminologies and concepts

- x_i : EDA-related query
- x_i^+ (pos sample): answer to x_i
- x_i^- (neg sample): Query with another EDA term

- Loss function (in-batch negative+hard negative):

$$-\log \frac{e^{sim(x_i, x_i^+)/\tau}}{\sum_{j=1}^M (e^{sim(x_i, x_j^+)/\tau} + e^{sim(x_i, x_j^-)/\tau})}$$

query (x_i): What is the significance of the **clock period** in EDA, and how does it impact STA (static timing analysis)?

positive sample

x_i^+ : **Clock period** is the maximum time for signals to propagate through the circuit during each clock cycle.

hard negative sample

x_i^- : What is the significance of the **clock skew** in EDA, and how does it impact STA (static timing analysis)?

Conclusion

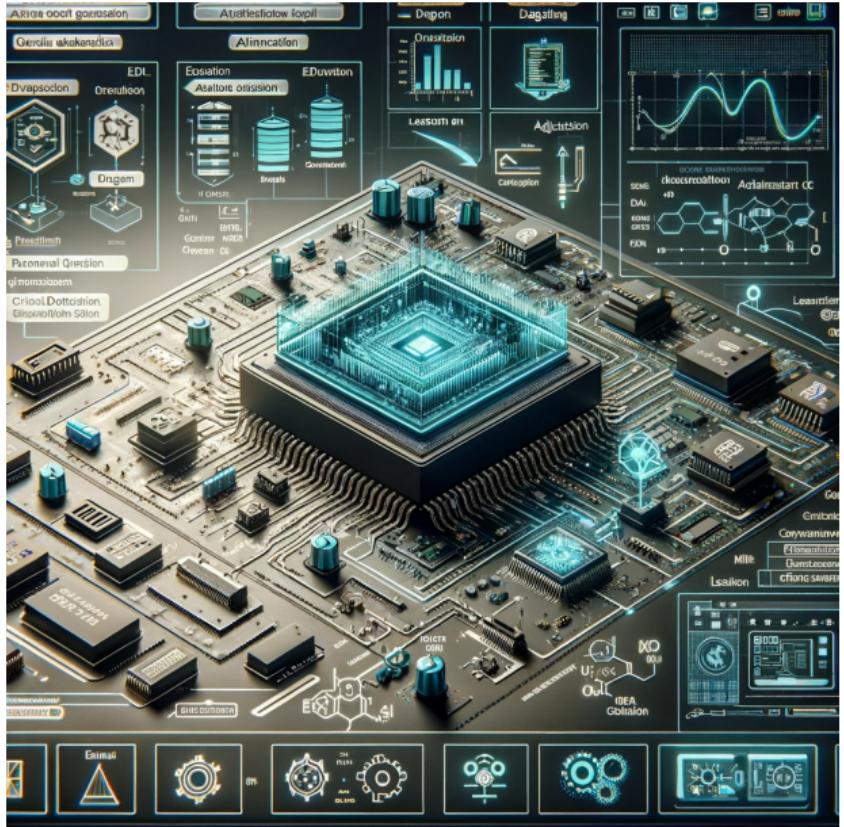


Summary: Applications of LLM

- RTL Automatic Generation
- EDA Tool Script Generation
- RTL&EDA Flow Debug

Other applications:

- Design and Tool Parameter Optimization
- Layout-level Optimization
- **Education**: toward Customers or Universities



Challenges: LLM + EDA



- Understand and customize RTL
- Understand and customize complicated EDA flow
- **Security**: How not to **hallucinate**
- **Privacy**: How not to leak IP
- Ethical use of AI

