

2024（第六届）集成电路 EDA 设计精英挑战赛

赛题指南

一、 赛题名称

针对模拟电路的电路图到网表自动生成

二、 命题企业

国家集成电路设计自动化技术创新中心

三、 赛题 Chair

杜力

四、 赛题背景

（一） 概览

电路图与网表（Netlist）是模拟电路设计中的两个核心元素。电路图以图形的形式直观地呈现了电路的构成和互连关系，为设计师、工程师和维修人员提供了理解电路工作机制的直观工具。而网表则以文本格式详细列出了电路中各个元件的连接细节，是执行电路仿真的前提条件。尽管电路图向网表的转换在一定程度上已经实现了半自动化，但这一过程仍然存在效率和准确性的问题。此外，互联网上存在大量的电路图像资料，这些宝贵的数据资源却常常未被充分利用，没有转化为可供电路仿真使用的结构

化数据。这表明，电路图的数字化和数据化潜力远未被完全挖掘，而这一领域的进步将对电路设计和仿真的效率产生重大影响。因此，开发一种能够自动将电路图转换为网表的工具，对提高电路设计效率和准确性具有重要意义。本赛题旨在开发一种智能化工具，能够自动识别模拟电路图的各种元件连接关系以及整体电路功能，并准确无误地转换成对应的网表文件。

电路图转换到网表的流程可以分解为三个任务：1. 电路图器件识别，2. 各个器件之间的连接关系识别以及 3. 电路功能识别。

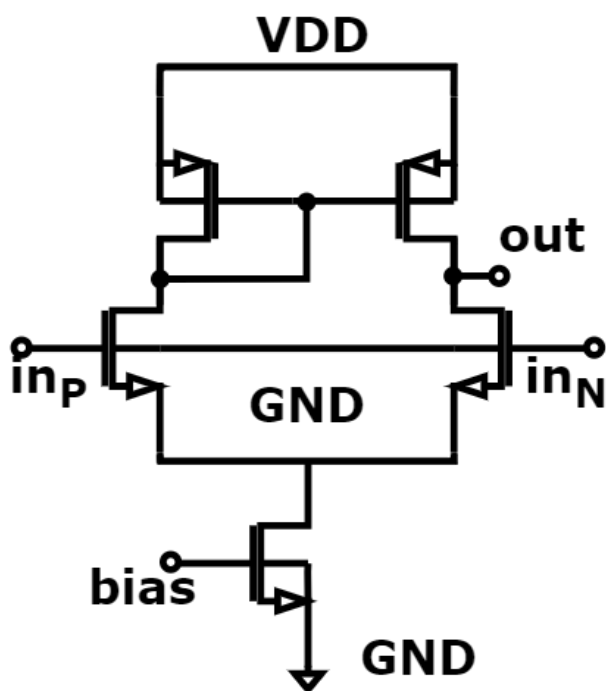
在任务 1 中，相较于传统的模式识别应用场景，电路图器件识别存在以下主要问题：1) **多样性与复杂性**：电路图中的元件种类繁多，从电阻、电容到更复杂的集成电路，每种元件都有不同的符号表示；2) **连接线的干扰**：电路图中的连接线可能会与元件符号交叉或遮挡，导致自动识别算法难以区分符号的边界和连接点；3) **注解和其他标记的混淆**：电路图中通常会包含额外的注解信息，如电压、电流标记，以及调试时手工添加的注记，这些都可能与元件符号混淆，干扰正确识别。

在任务 2 中，器件连接关系识别存在以下主要问题：1) **走线风格不同**：电路图会有带桥接的跨线画法或者用注明节点的电路画法，不便于统一寻路径算法对路径的识别；2) **线路弯曲与角度多变**：线路在电路图中可能以各种角度弯曲或转角，这些不规

则的形状可能增加算法识别的难度；3) **线宽不一致**：在不同的电路图中，线宽可能由于绘图风格不同而出现不一致，这可能会导致识别算法错误判断连接的完整性。

在任务 3 中，电路功能识别存在以下主要问题：1) **复杂性高**：电路图的复杂性可能会对自动识别系统构成挑战。多层次、高密度的电路设计可能难以通过自动化工具准确解析；2) **电路拓扑的变化**：即使是功能相同的电路，其拓扑结构（即组件的连接方式）也可能有很大差异。自动识别系统需要能够理解不同拓扑结构下的电路功能。

由于模拟电路极高的人工参与度以及以上提到的问题，业界始终没有成熟的全自动化电路图到网表转换工具。因此该赛题聚焦于对模拟电路的电路图到网表自动转换，参赛队伍需要设计并实现一个算法或软件，输入为标准格式的电路图，输出为带有器件功能批注且可以直接用于电路仿真软件的网表文件。不限实现方案。



原电路图

电路功能结果:

```
"circuit_type" : "amplifier"
```

网表识别结果:

```
[
  {
    "component_type" : "PMOS",
    "port_connection" : {
      "Drain" : "v1",
      "Gate" : "v1",
      "Source" : "Vdd",
      "Body" : "Vdd"
    }
  },
  {
    "component_type" : "PMOS",
    "port_connection" : {
      "Drain" : "out",
      "Gate" : "v1",
      "Source" : "Vdd",
      "Body" : "Vdd"
    }
  },
  {
    "component_type" : "NMOS",
    "port_connection" : {
      "Drain" : "v1",
      "Gate" : "InP",
      "Source" : "v2",
      "Body" : "Vss"
    }
  },
  {
    "component_type" : "NMOS",
    "port_connection" : {
      "Drain" : "out",
      "Gate" : "InN",
      "Source" : "v2",
      "Body" : "Vss"
    }
  },
  {
    "component_type" : "NMOS",
    "port_connection" : {
      "Drain" : "v2",
      "Gate" : "Bias",
      "Source" : "Vss",
      "Body" : "Vss"
    }
  }
],
]
```

图1. 电路图转网表与电路功能示意图

(二) 可行性分析

对于电路图转网表这一流程虽然不够成熟，但学术界也有相关的可参考工作如下：

[1]采用了 K-近邻算法 (KNN) 来对目标图像进行分类。该目标图像是基于 3500 张手写电路元件照片的像素分布特征进行分割的，并且达到了 93% 的器件识别正确率。但这个工作集中于识别手写电路，而且没有关注器件连接的识别，所以仍然有提升空间。

[2]使用经典 CV 模型 YOLOv5 对电路图像进行分析，训练模型以识别图像中的不同电路元件。该工作构建了 1,554 张电路图像的数据集,包含 13 种不同的电路元件,共计 11,500 个标注。并让 YOLOv5 基于这些包含多种元件标注的电路图像数据集，以实现高精度的元件分类。此外，在寻线上，该工作使用了在直线检测上表现良好的霍夫变换来实现对各个器件连接的检测。并达到最高 98%的器件识别率。此外，该工作还实现了对图片中器件参数的识别，进而直接生成带器件参数的网表文件。

[3] 使用经典 CV 模型 YOLOv8 对电路图像进行分析，训练模型以识别图像中的不同电路元件，并达到了 97.1%的正确率。并在固定的节点式画法下通过寻线算法以及对图片中的节点检测，实现了对一本教科书上所有电路图到网表的转换。但在该工作的开源数据集中，仍然存在很多明显的电路连接错误，代表着该工作的正确率仍有提升空间。

AnalogCoder [4] 采用了基于 GPT-4o 的一套模拟电路自动生成系统，并充分发挥了大模型的推理能力。实现了多种基础的模拟电路结构自动生成，并且表现出了对电路功能的理解。但在复杂模拟电路生成的程度上仍有提升空间。

[1]-[3]即使初步实现了对原理图转网表的自动流程，但在器件、连接关系识别准确度，画风识别通用性上仍有提升空间。[4]使用

了大模型来自动生成基础模拟电路模块，但在复杂模拟电路生成与理解上仍有提升空间。

(三) 参赛人员要求

需要对人工智能算法与模拟电路有所了解。面向专业为：计算机技术、人工智能、电子科学与工程、集成电路、数学等。

(四) 系统流程说明

将电路图转换为网表需要输入电路图片以在可行性分析中提及的方案为例，需要先将电路图上的器件进行精准识别并标注出每个器件的端口。其次，需要在已识别器件的电路图基础上对电路图中的网络走线进行识别，并进而将各个器件的连接关系记录下来。最后，将识别的器件与连接关系结合起来，生成电路网表。参赛者可与提供的 **golden data** 进行对比修正，优化模型。

在器件功能识别上，算法可结合输入的图片、前过程生成的网表来判断每个器件的功能。参赛者可与提供的 **golden data** 进行对比修正，优化算法。

赛题会提供新手代码样例作为参考。

1. 输入：模拟电路图片

➤ 需要识别的器件类型：见附录（一）

➤ 赛题提供的标注信息示例：见附录（二）

➤ 电路图片格式：png

2. 输出：1) 电路图对应的网表文件；2) 电路类型见附录（三），格式见附录（四）。

3. 方法：机器学习算法、卷积神经网络、图神经网络、多模态大模型等。

4. 注意事项：见附录（五）。

（五）提交说明

赛题最后需要提交两个部分：

1. 代码部分：提交可以执行的算法，支持的代码格式为：python 或加密的二进制文件，用于最后的评分；

2. 实验报告部分：以 word 格式提交队伍的作品简介，其中包括：技术方案、实验结果以及实验总结。

（六）评分标准

各参赛队的算法将在出题方提供的电路上进行测试。电路分为两部分：public data 与 hidden data。其中，参赛队可以看到 public data 中的电路图，以及对应的电路网表与功能用于优化迭代自己

的算法。Hidden data 为不可见的，参赛队伍仅可以在 hidden data 上进行测试，但只获得最后的准确度分析评分，用于队伍优化迭代自己的算法。最后仅通过在 hidden data 上的识别结果准确度进行评分。

对于每张电路图，将输出的电路网表与提供的正确网表与功能进行对比，针对电路网表在网表识别精准度、功能识别精准度与运行时间上进行评测。由国家集成电路设计自动化技术创新中心提供评测脚本。

其中，网表识别精准度关注器件种类识别与连接关系识别的正确率，性能指标关注识别电路时的运行时间。

整体网表识别精准度系数 K ，为每道题网表识别精准度系数 K_i 的平均值，其中 K_i 的定义为

$$K_i = GED(G_i, G_{i, golden})$$

其中，GED 为图编辑距离错误!未找到引用源。。 G_i 为第 i 题的网表预测结果所对应的图 (graph)， $G_{i, golden}$ 为第 i 题的正确网表对应的图，输出格式到网表的转换方式见附录 (六)。采用的算法为[6]，算法中所有的消耗 (cost) 都被设置为 1。

最终整体网表识别精准度系数 K 为每道题网表识别精准度系数求和后加 10 取对数倒数，其定义为：

$$K = \frac{1}{\lg(10 + \sum K_i)}$$

K 的范围在(0,1]区间内

下面是一个网表识别精准度系数计算实例：

<pre>[{ "component_type" : "PMOS", "port_connection" : { "Drain" : "v1", "Gate" : "v1", "Source" : "Vdd", "Body" : "Vdd" } }, { "component_type" : "PMOS", "port_connection" : { "Drain" : "out", "Gate" : "v1", "Source" : "Vdd", "Body" : "Vdd" } }, { "component_type" : "NMOS", "port_connection" : { "Drain" : "v1", "Gate" : "InP", "Source" : "v2", "Body" : "Vss" } }, { "component_type" : "NMOS", "port_connection" : { "Drain" : "out", "Gate" : "InN", "Source" : "v2", "Body" : "Vss" } }, { "component_type" : "NMOS", "port_connection" : { "Drain" : "v2", "Gate" : "Bias", "Source" : "Vss", "Body" : "Vss" } }],</pre>	<pre>[{ "component_type" : "PMOS", "port_connection" : { "Drain" : "v1", "Gate" : "v1", "Source" : "Vdd", "Body" : "Vdd" } }, { "component_type" : "PMOS", "port_connection" : { "Drain" : "out", "Gate" : "v1", "Source" : "Vdd", "Body" : "Vdd" } }, { "component_type" : "NMOS", "port_connection" : { "Drain" : "v1", "Gate" : "InP", "Source" : "v2", "Body" : "Vss" } }, { "component_type" : "NMOS", "port_connection" : { "Drain" : "out", "Gate" : "InN", "Source" : "v2", "Body" : "Vss" } }, { "component_type" : "PMOS", "port_connection" : { "Drain" : "v3", "Gate" : "Bias", "Source" : "Vss", "Body" : "Vss" } }],</pre>
正确电路网表	错误电路网表

图2. 网表识别精准度系数计算案例

在该例子中，错误电路网表的两处错误均在图中以红框标出。第一处错误为器件类型识别错误，误将 NMOS 识别成了 PMOS；第二处错误为端子连接错误，误将漏端连接到了 v3 网络，而非 v2 网络。本次比赛采用 GED（图编辑距离）定量分析正误网表间的差异。GED 的值为对一张图 G2 执行若干次节点或边的插入、删除、替换操作，使其转化为与另一张图 G1 同构的图所需的最小成本。若每种操作的成本均为 1，则 GED 即为该操作序列的长度。

在上述例子中，根据网表到图的转化规则（见附录（六）），为使错误网表的图 G2 转换为正确网表的图 G1，需进行 1 次边删除（PMOS 漏极到 v3 的边）、1 次节点替换（PMOS 换成 NMOS）、3 次边替换（PMOS 栅、源、衬底引出的边替换为 NMOS 相应端子连接的边）、1 次节点删除（删除 v3 节点）和 1 次边插入（NMOS 漏极到 v2 的边），共 7 次操作，故该题的 GED 值为 7。若网表识别无误，则 GED 值为 0。

此外，功能识别精准度 F 的计算方法为每道题功能识别精准度 F_i 的平均值，其中 F_i 的定义为：

$$F_i = \begin{cases} 1, & \text{correct Prediction} \\ 0, & \text{wrong Prediction} \end{cases}$$

当电路功能预测正确时，这道题的功能识别精准度为 1，否则为 0。

最终整体功能识别精准度 F 为每道题功能识别精准度 F_i 的平均，其定义为：

$$F = \frac{\sum F_i}{N}$$

此外，各组的代码运行时间 T 也将被记录并且作为最终的评判指标。

我们将会统计各组队伍的网表识别精准度系数 K 、功能识别精准度 F 以及运行时间 T 的最大值和最小值，并分别记为 K_{max} 和

K_{min} 、 F_{max} 和 F_{min} 、 T_{max} 和 T_{min} 。进而通过其他队伍的网表识别精准度系数、功能识别精准度与运行时间，分别记为 K_{mid} 、 F_{mid} 、 T_{mid} ，计算总得分 $Score$ 为：

$$Score = \frac{K_{mid} - K_{min}}{K_{max} - K_{min}} * 0.7 + \frac{F_{mid} - F_{min}}{F_{max} - F_{min}} * 0.2 + \left(1 - \frac{T_{mid} - T_{min}}{T_{max} - T_{min}}\right) * 0.1$$

$Score$ 越大排名越靠前。下面是一个计算得分的例子：

表 1.得分示例

队伍	网表识别精准度系数 K	功能识别精准度 F	运行时间 $T(s)$	$Score$
1	0.9	0.5	126	0.85
2	0.5	0.3	1247	0.35
3	0.1	0.4	22	0.13
4	0.2	1.0	90	0.38
最大值	0.9	1.0	1247	0.85
最小值	0.1	0.3	22	0.13

五、 参考资料

- [1] Huoming Z, Lixing S. Research on K nearest neighbor identification of hand-drawn circuit diagram[C]//Journal of Physics: Conference Series. IOP Publishing, 2019, 1325(1): 012233.

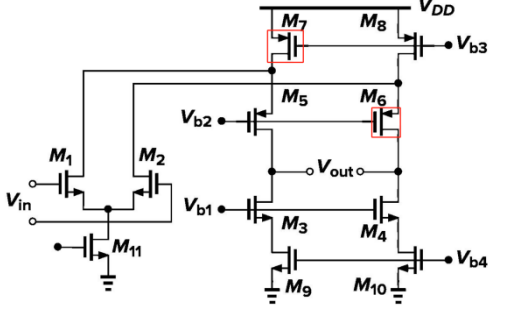
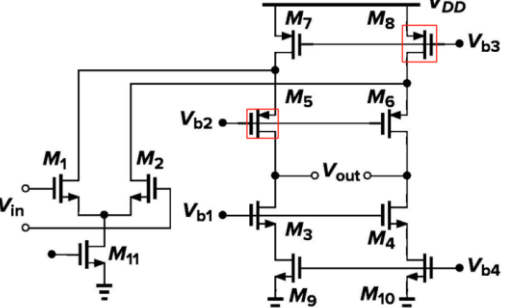
- [2] Sertdemir A E, Besenk M, Dalyan T, et al. From image to simulation: An ann-based automatic circuit netlist generator (img2sim)[C]//2022 18th International Conference on Synthesis, Modeling, Analysis and Simulation Methods and Applications to Circuit Design (SMACD). IEEE, 2022: 1-4.
- [3] Tao Z, Shi Y, Huo Y, et al. AMSNet: Netlist Dataset for AMS Circuits[J]. arXiv preprint arXiv:2405.09045, 2024.
- [4] Lai Y, Lee S, Chen G, et al. AnalogCoder: Analog Circuit Design via Training-Free Code Generation[J]. arXiv preprint arXiv:2405.14918, 2024.
- [5] Abu-Aisheh Z, Raveaux R, Ramel J Y, et al. An exact graph edit distance algorithm for solving pattern recognition problems[C]//4th International Conference on Pattern Recognition Applications and Methods 2015. 2015.
- [6] [GED 描述代码](#)

六、 附录

(一) 器件类型

下表为各位参赛队伍需要检测的器件类型，其中输出名称为最后输出结果中的需要统一的器件名称，标注名称为数据标注时的名称，端口为输出时该器件需要输出的端口名称。

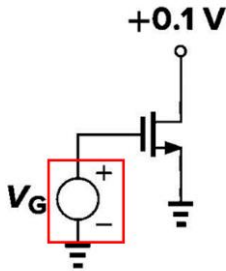
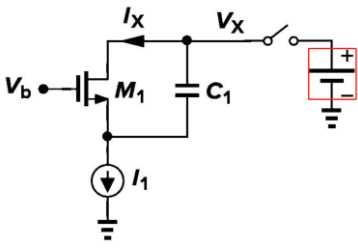
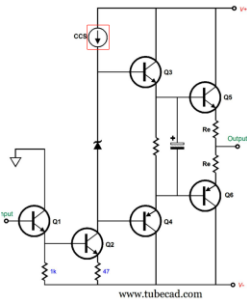
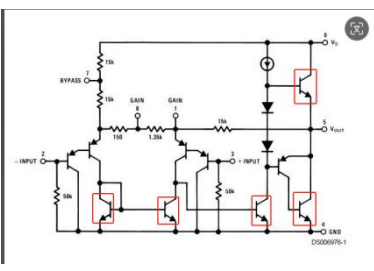
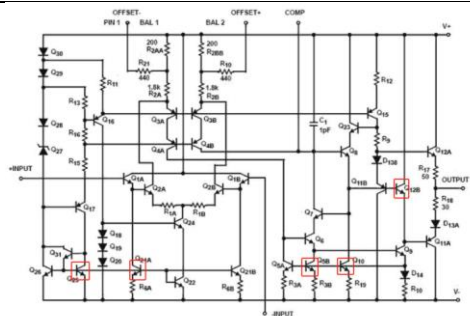
表 2.器件类型示例

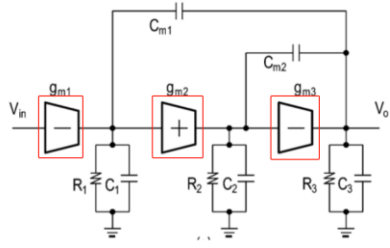
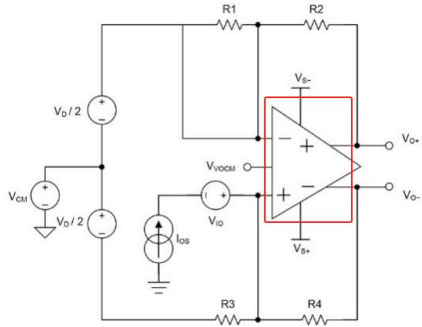
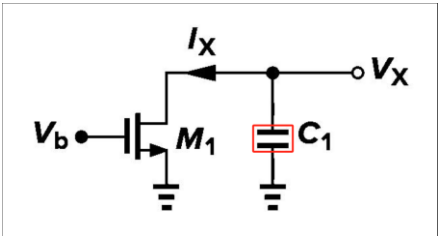
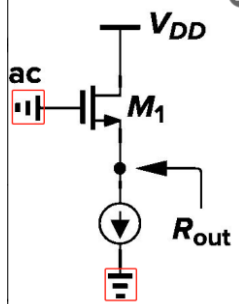
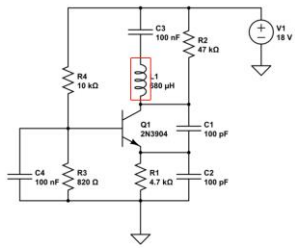
输出名称	标注名称	端口	图例
PMOS	PMOS_norma 1	Drain, Source, Gate	
	PMOS_cross	Drain, Source, Gate	

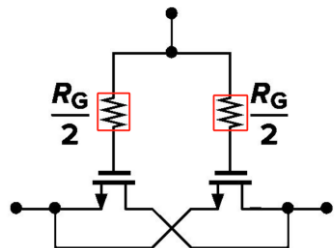
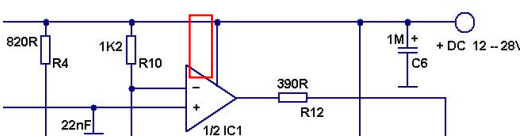
[illegible]

The diagram shows a differential pair of NMOS transistors, M_1 and M_2 , with their gates connected to an input V_{in} . Their sources are connected to a common tail node, which is also the source of a PMOS current mirror transistor M_{11} connected to ground. The tail node is biased by a current source consisting of a PMOS transistor M_8 (gate to V_{DD} , source to tail node) and an NMOS transistor M_{10} (gate to tail node, source to ground). The output nodes of the differential pair are connected to PMOS load transistors M_7 and M_6 (gates to V_{DD}) and NMOS load transistors M_5 and M_4 (gates to the output nodes). The gates of M_5 and M_4 are biased by a common-mode feedback circuit consisting of a PMOS transistor M_3 (gate to V_{DD} , source to tail node) and an NMOS transistor M_9 (gate to tail node, source to ground). The gates of M_3 and M_9 are connected to a common-mode feedback node V_{out} , which is also the source of a PMOS current mirror transistor M_{12} connected to V_{DD} . The gates of M_3 and M_9 are also connected to bias voltages V_{b1} and V_{b2} respectively. The output of the differential pair is taken differentially between the nodes connected to M_7 and M_6 .

[illegible]

Voltage	Voltage_1	Positive, Negative	
	Voltage_2	Positive, Negative	
Current	current	Positive, Negative	
NPN	BJT_NPN	Base, Emitter, Collector	
	BJT_NPN_cross	Base, Emitter, Collector	

Siso_amp	Siso_amp	In, Out	
Dido_amp	Dido_amp	InN, InP, OutN, OutP	
Cap	Capacitor	Pos, Neg	
Gnd	gnd	port	
Ind	inductor	Pos, Neg	

Res	Resistor_1	Pos, Neg	
	Resistor_2	Pos, Neg	

(二) 标注示例

本次赛题的标注工具采用的是 label studio，最终提供的标注电路图与标注结果示例如下：

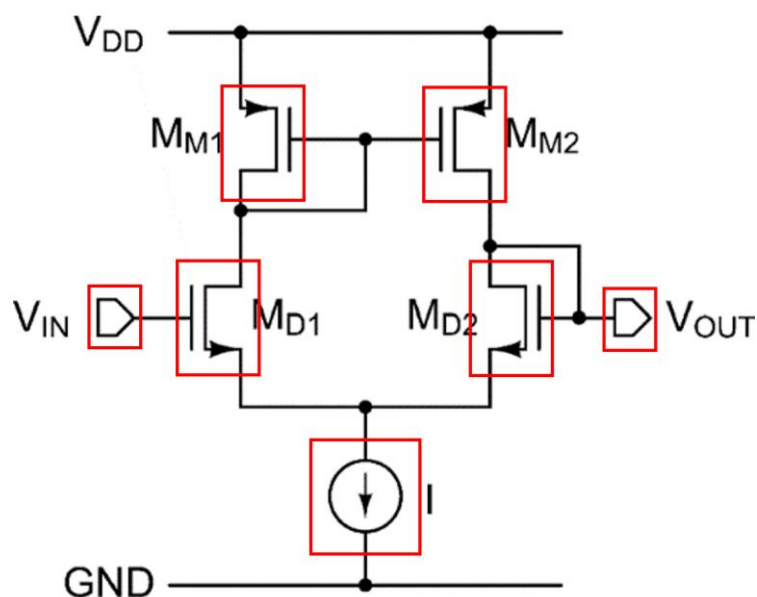


图3. 图片标注示例

代码 1. 图片标注结果

```
[
  {
    "label": "port",
    "points": [
      [
        69.8544,
        246.3727
      ],
      [
        108.6007,
        283.0
      ]
    ]
  },
  {
    "label": "pmos",
    "points": [
      [
        195.8154,
        71.1878
      ],
      [
        247.0,
        140.1066
      ]
    ]
  },
  {
    "label": "current",
    "points": [
      [
        276.0714,
        387.3971
      ],
      [
        341.1636,
        452.4718
      ]
    ]
  }
]
```

```
]
},
{
  "label": "pmos",
  "points": [
    [
      369.2658,
      72.257
    ],
    [
      420.4567,
      141.1797
    ]
  ]
},
{
  "label": "nmos",
  "points": [
    [
      153.4384,
      233.9859
    ],
    [
      202.0,
      299.7065
    ]
  ]
},
{
  "label": "port",
  "points": [
    [
      523.3775,
      246.0732
    ],
    [
      562.1272,
      282.7187
    ]
  ]
}
```

```

},
{
  "label": "nmos",
  "points": [
    [
      414.6906,
      232.9038
    ],
    [
      465.7018,
      300.8026
    ]
  ]
}
]

```

(三) 电路功能识别类型

表 3.需要识别的电路类型

中文名	标签
差分输入单端输出放大器	DISO-Amplifier
差分输入双端输出放大器	DIDO-Amplifier
单端输入单端输出放大器	SISO-Amplifier
带隙基准	Bandgap
稳压器	LDO
比较器	Comparator

（四） 输出格式

算法应基于 python 进行开发，每识别一道题时应该输出这道题对应的识别结果，格式为 python 中的字典类型变量。

该字典型变量应包括电路功能预测与电路连接关系预测两个部分：

```
Dict {"ckt_type" : <The predicted circuit's type>,  
      "ckt_netlist" : <The predicted circuit's netlist>,  
      }
```

其中“ckt_type”为预测的电路类型，对应的 value 为字符串类变量；“ckt_netlist”为预测的电路网表，对应的 value 为字典类变量组成的列表，其中每个字典为识别出的各个器件与每个端口的连接网络：

```
Dict {"component_type" : <The component's type>,  
      "port_connection" : <The component ports' connection>  
      }
```

其中每个端口的连接网络需为字典类型变量，字典类型的 key 为该器件的端口（见附录（一）），其 value 为对应的网络端口，为字符串类型变量。以一个预测出为电容的器件为例，则该器件对应的字典类型变量为：


```
Dict {"component_type" : "capacitor",  
      "port_connection" : {  
          "positive" : "net0",  
          "negative" : "net1"  
      }  
}
```

（五）注意事项

1. 对于 MOSFET 而言，如果 body 端没有明确的连接说明，请默认与其 source 端相连；
2. 无源器件，包括电阻、电容、电感，的正负极在判断时不进行区分，即无源器件的端口极性识别反了在评分时也是可通过的；
3. 由于 MOS 器件的对称性，MOSFET 的 Source 和 Drain 在判断时不进行区分，即 NMOS 和 PMOS 的 Source 和 Drain 端口识别反了在评分时也是可通过的；
4. 出题方会提供如附录（二）中格式的标注图片数据约 2000 张。如参赛队伍额外引入了新的电路图以及标注数据，需要在最后的提交实验报告材料时以附件的形式提交。
5. 本次赛题关注的是网表准确率为电路拓扑结构准确率，所以对端口名称不进行检测。
6. 如果遇到一个器件有多种端口可能的，如附录（一）

中的 PMOS 与 NMOS，最后的输出结果中的器件类型（"component_type"）仍需自行统一为输出名称，在端口连接里可依据具体器件细分类型对应的端口来进行区分。

7. 一些非常规器件（如三端口电感、可变电阻等）在本次赛题中不会出现。

（六）网表转图（graph）方案

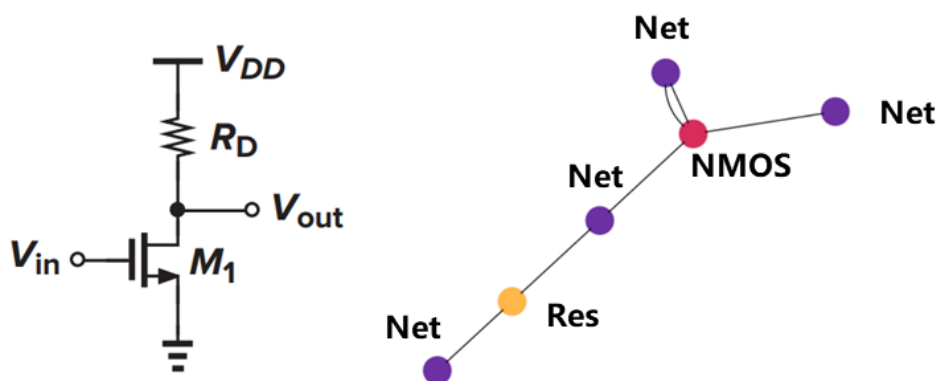


图4. 图片转 graph 示意

本次比赛采用异质图来抽象表示电路图的拓扑结构。定义无向异质图 $G = \{V, E\}$ ，其中 V 和 E 分别代表节点集合和边集合。每个节点 $v \in V$ 和每条边 $e \in E$ 都与它们的映射函数 $\phi(v) \rightarrow A$ 和 $\varphi(e) \rightarrow R$ 相关联。 A 和 R 分别表示节点类型集合与边类型集合。在本方案定义的异质图中，节点类型集合 A 的元素为各元器件（如 PMOS, NMOS, Capacitor, Resistor 等见，附录（一））以及网络（net），边类型集合 R 的元素为各元器件端子到网络的

连接（如 PMOS 漏端到网络的连接、PMOS 源端到网络的连接、Resistor 到网络的连接等）。以图 1 为例，左侧的原理图中有 1 个 NMOS、1 个电阻、4 个网络，分别对应于右侧异构图中的 1 个 NMOS 类节点、1 个 Resistor 类节点和 4 个 net 类节点。右图中共有五种类型的边，分别是 2 条从 Resistor 到 net 的边、从 NMOS 源极、漏极、栅极和衬底到 net 的边各 1 条。

本方案将从参数队伍提交的网表文件中获取电路的拓扑结构并表示成异质图的形式，计算其与标准电路异质图之间的图编辑距离（GED）。GED 的计算是通过调用 NetworkX 工具包提供的 graph_edit_distance 函数实现的。两个节点或两条边被认为是相同的，当且仅当它们的类型和拓扑关系均相同。所有操作类型的成本均为 1，GED 的值即为最少操作次数。计算时间上限设置为 5 分钟，超时后，函数将返回当前的最佳 GED。关于该函数的更多信息详见[6]。