

1 数据介绍

1.1 PCB

- 用途

| 领域 | 应用 | 材料 | 过程 |
|------------|-------------|-----------------------|----------|
| 汽车/消费电子/工业 | 无铅焊接技术—绿色制程 | 100%无卤素阻燃剂/25%阻燃剂 | 回流 |
| 医药、消费电子/工业 | 100%无卤素 | 100%阻燃剂 | 回流 |
| 医疗 | 100%无卤素 | 100%无卤/25%阻燃剂/100%阻燃剂 | 回流/回流/回流 |
- 制程/设备

| 领域 | 应用 | 制程 | 设备 |
|---------|---------------|-------|----------|
| 医药 | 100%无卤/100%阻燃 | 回流/回流 | 回流/回流/回流 |
| 消费电子/工业 | 100%无卤 | 回流/回流 | 回流/回流/回流 |
| 消费电子/工业 | 100%无卤 | 回流/回流 | 回流/回流/回流 |

1.2 印刷

初始数据

自创2个缺陷和
可能会出现异物影响
增加缺陷的出现频率

工艺问题/改进

1 PCB工艺

[illegible]

一共57个不完全
相同的
缺陷-原因-措施

2 印刷工艺

① 少度以阳工艺中有强漏大面缺少的缺陷, 其会导致数据部的阻塞, 造成数据提升速度变慢, 与其对应的快速修复措施是时间 (光驱盘修复数据), 使几次后再次访问。

技术文档

工艺问题/改进

1 PCB工艺

【例 1】甲乙两村共有 10 亩山林，因产生林木所有权归属发生纠纷诉至法院，法院因甲村人口众多而决定委托，与村内的该宗族长辈和不动产产品生产者协商处理，

【例 2】甲村有 10 亩山林，因产生林木所有权归属发生纠纷，法院因甲村人口众多而决定委托，与村内的该宗族长辈和不动产产品生产者协商处理，

【例 3】甲村有 10 亩山林，因产生林木所有权归属发生纠纷，法院因甲村人口众多而决定委托，与村内的该宗族长辈和不动产产品生产者协商处理，

【例 4】甲村有 10 亩山林，因产生林木所有权归属发生纠纷，法院因甲村人口众多而决定委托，与村内的该宗族长辈和不动产产品生产者协商处理，

【例 5】甲村有 10 亩山林，因产生林木所有权归属发生纠纷，法院因甲村人口众多而决定委托，与村内的该宗族长辈和不动产产品生产者协商处理，

【例 6】甲村有 10 亩山林，因产生林木所有权归属发生纠纷，法院因甲村人口众多而决定委托，与村内的该宗族长辈和不动产产品生产者协商处理，

【例 7】甲村有 10 亩山林，因产生林木所有权归属发生纠纷，法院因甲村人口众多而决定委托，与村内的该宗族长辈和不动产产品生产者协商处理，

【例 8】甲村有 10 亩山林，因产生林木所有权归属发生纠纷，法院因甲村人口众多而决定委托，与村内的该宗族长辈和不动产产品生产者协商处理，

【例 9】甲村有 10 亩山林，因产生林木所有权归属发生纠纷，法院因甲村人口众多而决定委托，与村内的该宗族长辈和不动产产品生产者协商处理，

【例 10】甲村有 10 亩山林，因产生林木所有权归属发生纠纷，法院因甲村人口众多而决定委托，与村内的该宗族长辈和不动产产品生产者协商处理，

2 印刷工艺

②少引粉工艺中有磨屑大面积缺失的缺陷,其会导致漏磨的后果,造成原因是研磨速度,与其对应的改善措施是调整研磨速度。

技术文档

基于RAFT

基于脚本

Craft_QA_miniled

包含4个无关的
context作为干扰项

微调时增强LLM的RAG能力

Reports_1000_sample

基于频率假设了
1000个出现缺陷的
样品

2 已有工作介绍

- 微调llama
 - 基于peft库的LoRA完成微调——数据集为RAFT的QA数据

```

What are some potential defects that can cause LEDs to not light up properly? , # input
, # output - leave this blank for generation!

], return_tensors = "pt").to("cuda")

outputs = model.generate(**inputs, max_new_tokens = 512, use_cache = True)
tokenizer.batch_decode(outputs)

Unused kwargs: ['load_in_4bit', 'load_in_8bit', 'quant_method']. These kwargs are not used in <class 'transformers.utils.quantization_config.BitsAndBytesConfig'>.
==((====))= Unsloth: Fast Llama patching release 2024.4
\\ // GPU: Tesla T4. Max memory: 14.748 GB. Platform = Linux.
0'o/ \_// Pytorch: 2.2.1+cu121. CUDA = 7.5. CUDA Toolkit = 12.1.
\_-__ Bfloat16 = FALSE. Xformers = 0.0.25.post1. FA = False.
Free Apache license: http://github.com/unslothai/unsloth
Special tokens have been added in the vocabulary, make sure the associated word embeddings are fine-tuned or trained.
Setting 'pad_token_id' to 'eos_token_id':128001 for open-end generation.
[<|begin_of_text|>Below is an instruction that describes a task, paired with an input that provides related or disturbing contexts. Please give the most likely reason
and the corresponding improvement strategy briefly without analysis step by step.\n\n### Instruction:\nAnswer the Question in input. Please give the most likely reason
properly?\n\n### Response:\nStep 1: Identify the key elements in the question. The question is asking about potential defects that can cause LEDs to not light up
properly.\n\nStep 2: Look for relevant information in the input and the provided context. The input mentions "short circuit defects" and "solid mounting errors".\n\nStep 3: Formulate the
concise.\n\n<ANSWER>: The potential defects that can cause LEDs to not light up properly are short circuit defects and solid mounting errors.<|end_of_text|>"]

```

You can also use Hugging Face's . Only use this if you do not have a GPU, and Unsloth's inference is 2x faster. AutoModelForCausalLM.from_pretrained("unslothai/llama-3.1-8b", torch_dtype=torch.float16, device_map="auto").

```

[] if False:
    # I highly do NOT suggest - use Unsloth
    from peft import AutoPeftModelForCausalLM
    from transformers import AutoTokenizer
    model = AutoPeftModelForCausalLM.from_pretrained("unslothai/llama-3.1-8b", torch_dtype=torch.float16, device_map="auto")
    tokenizer = AutoTokenizer.from_pretrained("unslothai/llama-3.1-8b")
    model = AutoPeftModelForCausalLM.from_pretrained(model.model_name_or_path, torch_dtype=torch.float16, device_map="auto")
    tokenizer = AutoTokenizer.from_pretrained(model.model_name_or_path)

```

Below is an instruction that describes a task, paired with an input that provides related or disturbing contexts. Please give the most likely reason and the corresponding improvement strategy briefly without analysis step by step.

Input: What are some potential defects that can cause LEDs to not light up properly?

Response:

Step 1: Identify the key elements in the question. The question is asking about potential defects that can cause LEDs to not light up properly.

Step 2: Look for relevant information in the input and the provided context. The input mentions "short circuit defects" and "solid mounting errors".

Step 3: Formulate the concise.

ANSWER: The potential defects that can cause LEDs to not light up properly are short circuit defects and solid mounting errors.

Below is an instruction that describes a task, paired with an input that provides related or disturbing contexts. Please give the most likely reason and the corresponding improvement strategy briefly without analysis step by step.

Input: What are some potential defects that can cause LEDs to not light up properly?

Response:

Step 1: Identify the key elements in the question. The question is asking about potential defects that can cause LEDs to not light up properly.

Step 2: Look for relevant information in the input and the provided context. The input mentions "short circuit defects" and "solid mounting errors".

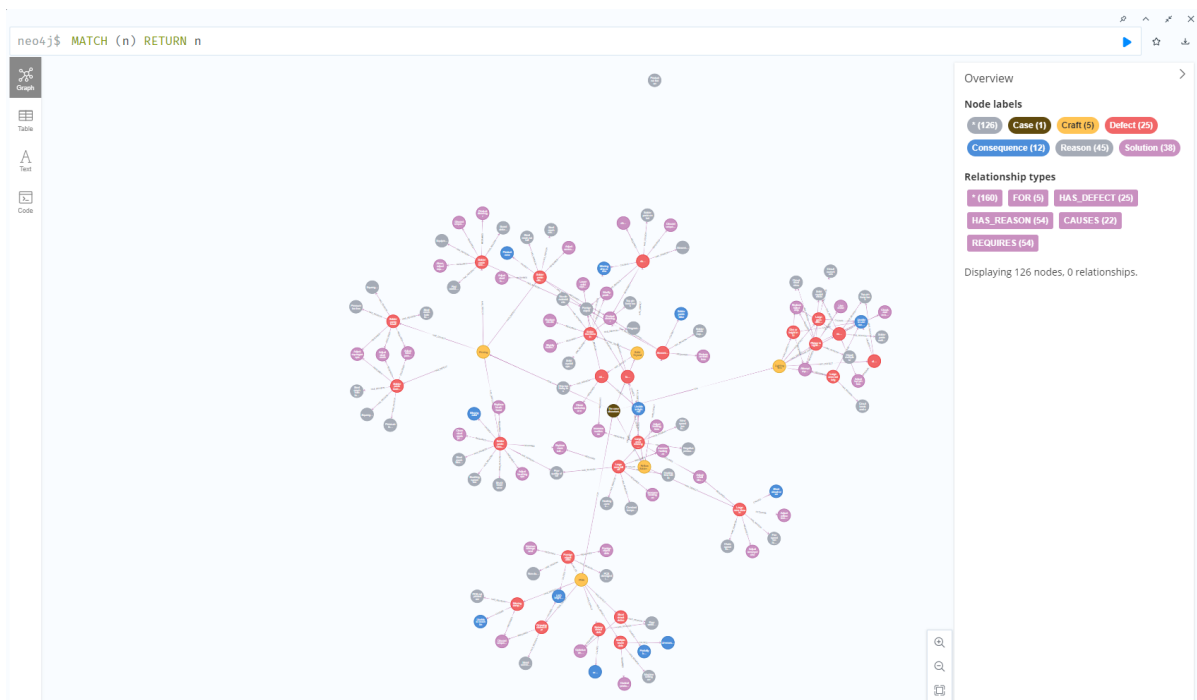
Step 3: Formulate the concise.

ANSWER: The potential defects that can cause LEDs to not light up properly are short circuit defects and solid mounting errors.

- RAG构建向量库

- Graph知识图谱

- 通过技术文档构建知识图谱——neo4j数据库 + gpt-4-32k提取关系



- 调研介绍

- llama3大部分工作都是微调，少有缝合或加模块

llama-pro通过文本修改的方式增加block

```

23     output = {}
24     for i in range(args.original_layers):
25         for k in ckpt:
26             if ('layers.' + str(i) + '.') in k:
27                 output[k.replace(('layers.' + str(i) + '.'), ('layers.' + str(layer_cnt) + '.'))] = ckpt[k]
28         layer_cnt += 1
29         if (i+1) % split == 0:
30             for k in ckpt:
31                 if ('layers.' + str(i) + '.') in k:
32                     if 'down_proj' in k or 'o_proj' in k:
33                         output[k.replace(('layers.' + str(i) + '.'), ('layers.' + str(layer_cnt) + '.'))] = torch.zeros_1
34                     else:
35                         output[k.replace(('layers.' + str(i) + '.'), ('layers.' + str(layer_cnt) + '.'))] = ckpt[k]
36
37         layer_cnt += 1
38
39     assert layer_cnt==args.layers
40     for k in ckpt:
41         if not 'layers' in k:
42             output[k] = ckpt[k]
43
44     torch.save(output, args.output_path)
45

```

- **评估指标**——RAGAs自带的4个检索相关指标，参考论文lawyer-llama是否有相关领域的指标

lawyer llama

其工作与我们相似，微调+RAG，不同的是其做了很多组微调（毕竟数据够），也就是先大微调加入领域知识，再小微调加入推理或检索能力等。

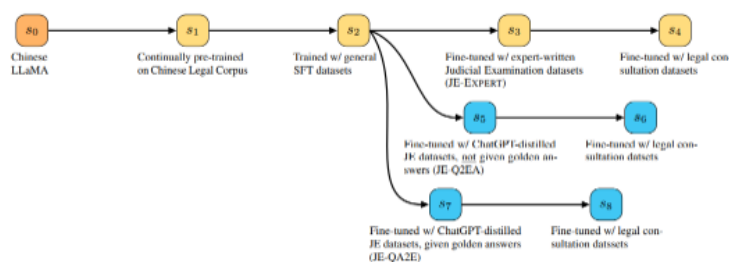


Figure 2: The training process of Lawyer LLaMA, where each node s_i represents the i -th training stage.

论文做了以下实验：

- 对比不同微调阶段的性能——其中涉及领域的**专业评价指标**和普遍的评价指标
- 计算了模型幻觉的占比
- 展示了专家以及其他模型输出的排名(哪个更好)
- 其他就是做了大量的讨论，展示了prompt和response

上下文精度、上下文召回率、事实一致性、答案相关性

3 未来计划

- 设计知识图谱检索的实验——展示多跳图谱的检索效果
- 设计其余的LLM实验——时间、幻觉占比.....