Siri for Network Configuration

Ze Yang, Ziheng (Jack) Chen, Shengkun Cui

Motivation

- Network configuration is crucial to cloud system:
  - Connectivity: Enables communication between different components.
  - Reliability: Provides performance and availability that meet user SLO.
  - Security: Secures network traffics and prevents unauthorized access.
- Network configuration (routers, interfaces, subnets, switches) validation and correction is complicated and error-prone.
  - Complicated: "Configuration-as-code" approach, rigorous testing requires static validation, configuration testing, and manual review and approval.
  - Error prone: Current solution (over 90% of participants) is to use manually generated python/JSON/vendor-specific network scripts with ansible playbooks [1].

[1] https://blog.networktocode.com/post/state-network-operations-netdevops-survey-2019/

Problem Statement

OpenAI
GPT - 4

- LLM as a network configuration validation tool
  - LLM-aided network management has started to emerge, which proves that LLM can generate code for network traffic analysis and network topology manipulation Graph Manipulation [2].
  - **LLM, with code generation ability, fits the "configuration-as-code" paradigm.**
  - **LLMs offer advantages such as automatic scalability, generalization, and reasoning abilities, making them suitable for misconfiguration detection.**

- Problem statement: **Characterize the SOTA LLM (GPT-4 Turbo)'s performance and failure modes in detecting and correcting network misconfigurations.**

[2] https://arxiv.org/pdf/2308.06261.pdf

Technical Challenges

- General LLM models are not fine-tuned to network configuration tasks.

- Large number of configurations do not fit into LLMs context size. (Improved GPT-4 Turbo offers a 128,000-token context window)

- LLM can hallucinate and generate incorrect or unspecific outputs.

- LLM corrected configurations can be grammarly correct but functionally invalid or harmful, e.g., network storm, incorrect firewall rules.

Key Contributions

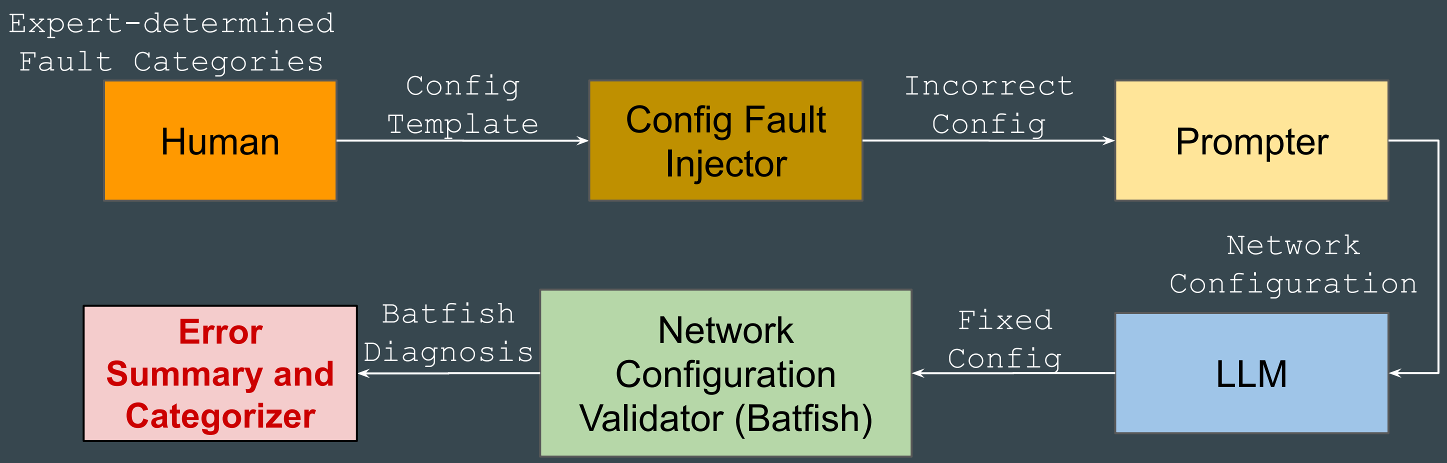
- Used validation-based post-checker for LLM corrected network configuration.

- Created network config injector to programmatically inject single or multiple faults in network configurations by 5 diverse (sub)-categories.
    - Invalid symbol, invalid subnet, invalid range, invalid interface, functional error.
    - Based on previous work (ISSTA 2021 [3]) on configuration error characterization.

- Characterization of SOTA LLMs on network configuration error correction in these five categories.

[3] https://dl.acm.org/doi/10.1145/3460319.3464799

Approach: Overview



1. Human sends a correct net config to an fault injector, specifying the fault category to be injected as well as the number of faults.
2. Fault injector automatically generates net configurations with altered content and send it to prompter.
3. Prompter sets the context for the LLM by providing system and user prompt and ask LLM to find the error(s).
4. LLM completes the config and pass it to the validator (Batfish) that provides correctness checks.
5. Batfish outputs validation results, and result is summarized.

Approach: Fault Categories and Dataset

Category	Sub-Category	Explanation	Example	Number
Syntax	Invalid Symbol	Randomly inserting ["@", "%", "^", "&", "(", ")"] at the beginning or end of a line	service timestamps log datetime msec -> service timestamps log datetime msec^	25
Syntax	Invalid Subnets	Randomly inserting ["@", "%", "^", "&", "(", ")"] in subnet expressions	2.128.0.0/9 -> 2.128.0.0//9	10
Syntax	Invalid Interfaces	Randomly inserting ["@", "%", "^", "&", "(", ")"] in interface expressions	interface GigabitEthernet0/0 -> interface Gigab%itEthernet0/0	20
Range	IP out-of-range	Randomly changing a part of the IP out of range (only 1 IP address is changed)	neighbor 2.1.2.322 activate	5
Range	IP out-of-range	Randomly changing a part of the IP out of range (multiple IP address is changed)	neighbor 2.1.2.322 activate neighbor 2.322.2.200 activate	20
Functionality	Incorrect Permissions	Randomly swapping a permission keyword ['permit', 'deny', 'remark'] with another	ip community-list expanded as1_community permit _1: -> ip community-list expanded as1_community remark _1:	20

Approach: Prompt Formatting

- System Prompt:
  - You are an network configuration operator. And you will find error(s) and guarantees the correctness of planned or current network configurations.
- User Prompt:
  - The network configuration is below: {Config}
  - Pls help me find the most possible error(s) in the configuration and help me correct it.
  - The answer format should be a json like below and pls only return json:
    - Correctness
      - Yes, if there is any error and No, if there is no error
    - RootCause:
      - The top five error(s) with their line number, content and explanation of the error(s).
      - If there are less than five errors, list as many as you have.
  - Example: Provided an example…(omit for space)

Results

RQ1: How does GPT-4 perform overall?

Category	Sub-Category	Total #	Error found in top-1 solution	Error found in top-5 solution
			Count (Percentage)	Count (Percentage)
Syntax	**Invalid Symbol**	**25**	**13 (52%)**	**16 (64%)**
Syntax	Invalid Subnets	10	7 (70%)	9 (90%)
Syntax	Invalid Interfaces	20	19 (95%)	19 (95%)
Range	**IP out-of-range**	**25**	**24 (96%)**	**25 (100%)**
Functionality	**Incorrect Permissions**	**20**	**0 (0%)**	**1 (5%)**
Total		**100**	**63 (63%)**	**70 (70%)**

Results

RQ2: How does GPT-4 perform on finding multiple errors in a file?

- Varying the number of faults per file in the "Range Error" category from 1-5 faults per file.

# of faults per file	Average success rate per file
1	100%
2	100%
3	87%
4	95%
5	88%

Results

RQ3: Can domain-specific context improves GPT-4's performance?

- In-context learning:
    - `You may encounter errors like invalid range for IP, invalid syntax, invalid character insertion, functional misconfig and so on.`
    - `For invalid character insertion, some characters like @, %, ^, &, (, ), / may be mistyped into the config.`
    - `e.g., "%" is mistyped into the config "%version 15.2" "/" is mistyped into "ip prefix-list inbound_route_filter seq 5 deny 2.0.0.0/8 le 32`

Results

RQ3: Can domain-specific context improves GPT-4's performance?

Category	Sub-Category	Total #	Error found in top-1 solution	Error found in top-5 solution
			Count (Percentage)	Count (Percentage)
Syntax	Invalid Symbol	25	13 (52%) → 21 (84%)	16 (64%) → 25 (100%)
Syntax	Invalid Subnets	10	7 (70%)	9 (90%)
Syntax	Invalid Interfaces	20	19 (95%)	19 (95%)
Range	IP out-of-range	25	24 (96%)	25 (100%)
Functionality	Incorrect Permissions	20	0 (0%) → 1 (5%)	1 (5%) → 7 (35%)
Total		100	63 (63%) → 72 (72%)	70 (70%) → 85 (85%)

Discussion

- What is the broader impact of your work?
  - GPT itself can help with testing of some types of configurations.
  - GPT with in-context learning can boost Network Configuration testing without any model tuning.
    - UniLog: Automatic Logging via LLM and In-Context Learning (ICSE'24).

- What are the limitations and areas for future improvements?
  - The amount of our experiments is not sufficient.
  - Automatic in-context learning example selection.

- Suggest ideas for how your idea could be extended?
  - Fine tune Network LLM vs In-Context Learning.
    - Xpert: Empowering Incident Management with Query Recommendations via Large Language Models (ICSE'24).

Statement of individual contributions

- Ze Yang: Responsible for network configuration pipeline implementation and prompt designing.
- Shengkun Cui and Ziheng (Jack) Chen: Batfish verification pipeline and config fault generation, and in-context learning example design.
- We all work on the evaluation and report writing.