DL Project Update report

Note: AI tools were used to better organize data for this report

- Problem: Provide a concrete problem statement and formulation.
- Method: You should have already implemented a basic version of your algorithm / model and show that it's working (for your problem).
- Dataset: Finished data preprocessing (if applicable). If you need to collect new data, you should have already done that.
- Experiment: If you want to improve on a model, you should already have the results on the baseline (the method you are based off of).

Problem

Problem Statement:
Given an incident description (e.g., "Machine is not working right and the pressure is low, machine red light on, temperature warning"), retrieve the most relevant Standard Operating Procedure (SOP) and generate a structured, executable plan with validated safety constraints.

Formulation:

- **Input:** Natural language incident text q (e.g., incident description)
- **Output:** JSON plan with structure:

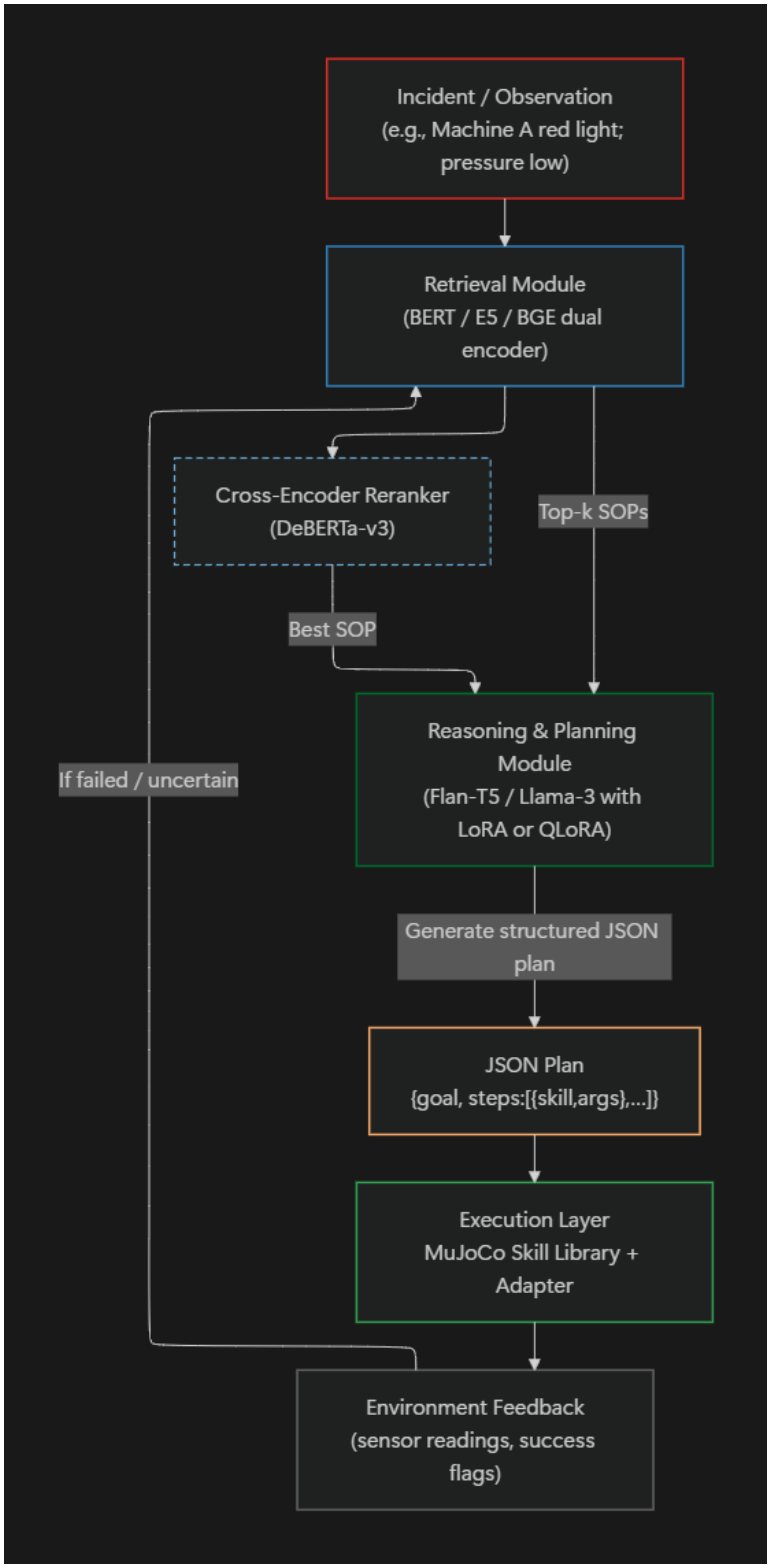
$$\text{Plan} = \{\text{goal,steps}[\{\text{skill,args}\}],\text{fallback}[\cdot]\}$$

Components:

1. **Retrieval Stage:** Find most relevant SOP s^* from 100 SOPs using dual-encoder BERT with InfoNCE loss (Still experimenting and researching into what loss function is the best for my case)
2. **Planning Stage:** Generate structured plan from incident + retrieved SOP using Flan-T5 + LoRA
3. **Validation Stage:** Ensure all skills are in whitelist of 8 allowed skills for the humanoid robot ()

Method

Architecture:



Component	Model	Loss Function	Notes
Retrieval	Dual-Encoder BERT	InfoNCE (Contrastive)	Separate query/passage encoders with batch negatives
Reranker	DeBERTa-v3-base	Cross-entropy	Precision reranking of top-k candidates
Planning	Flan-T5-base + LoRA	Seq2Seq (Teacher Forcing)	Low-rank adaptation: $r = 32$, $\alpha = 64$, dropout=0.1
Execution	MuJoCo Skill API	N/A	Dummy adapter (extensible to real MuJoCo)

Key Algorithm Details:

- **Dual-Encoder Retrieval:** Normalizes embeddings $\vec{q}, \vec{p}$ via L2, computes similarity scores via dot-product, and applies InfoNCE contrastive loss with in-batch negatives.
- **LoRA Fine-tuning:** Adapts Flan-T5 with low-rank matrices on Q, V, O attention modules; minimal parameter overhead
- **Safety Validation:** All generated plans checked against 8-skill whitelist before execution

Dataset

Data Preprocessing Completed:

1. **100 Standard Operating Procedures**
    - **SOP-001 to SOP-020:** Machine control (pressure, alarms, startup, shutdown, diagnostics)
    - **SOP-021 to SOP-040:** Table 1 manipulation (pick, place, sort, retrieve)
    - **SOP-041 to SOP-060:** Table 2 manipulation (tool management, parts, transfers)
    - **SOP-061 to SOP-100:** Complex workflows (multi-step, conditional logic, dual-object operations)
2. **100 Incident Examples**
    - Diverse incidents sampled from SOP conditions and contextual variations
    - Ground-truth SOP mappings provided via labels

Example this is SOP 001:

{"sop_id":"SOP-001","title":"Low Pressure Warning","condition":"Blue indicator is flashing","steps":["Walk to Machine","Read pressure_sensor","If below threshold, press blue_diagnostic_button","Wait 3 seconds","Re-read pressure_sensor","Notify tech if unchanged"],"equipment":["machine","pressure_sensor","blue_diagnostic_button"]}

Example this is Incident 001:

{"incident_id":"INC-001","text":"Blue indicator on the main machine is flashing and pressure looks low; run the low-pressure diagnostic and confirm sensor readings.", "labels":{"sop_id":"SOP-001"}}

Train/Validation Split: 80/10/10 (100 SOPs → 80 train, 10 val, 10 test)

Experiment

Baseline & Current Implementation:

Metric	Baseline (10 SOPs)	Current (100 SOPs)
Retriever Training	5 epochs, batch=2, r=8	50 epochs, batch=8, r=32
Planner Training	3 epochs, batch=2	10 epochs, batch=4
Training Time	~1 min retriever + ~30s planner	~2-5 min retriever + ~1-2 min planner

And below attached pictures of the terminal after training the dual encoder and the results from the training where the models correctly identify the SOP.

```
             Batch size: 8                                  build_dual_encoder.py:260
             Learning rate: 5e-05                           build_dual_encoder.py:261
[18:43:37] Loading tokenizer: bert-base-uncased             build_dual_encoder.py:91
[18:43:38] Loading model (this may download weights on      build_dual_encoder.py:94
           first run)...
[18:43:41] Epoch 1/50: avg_loss = 1.3163                    build_dual_encoder.py:168
           Epoch 2/50: avg_loss = 1.1903                    build_dual_encoder.py:168
[18:43:42] Epoch 3/50: avg_loss = 1.0255                    build_dual_encoder.py:168
           Epoch 4/50: avg_loss = 1.0343                    build_dual_encoder.py:168
[18:43:43] Epoch 5/50: avg_loss = 0.9392                    build_dual_encoder.py:168
           Epoch 6/50: avg_loss = 0.9481                    build_dual_encoder.py:168
           Epoch 7/50: avg_loss = 0.8947                    build_dual_encoder.py:168
[18:43:44] Epoch 8/50: avg_loss = 0.8708                    build_dual_encoder.py:168
           Epoch 9/50: avg_loss = 0.9218                    build_dual_encoder.py:168
[18:43:45] Epoch 10/50: avg_loss = 0.9727                   build_dual_encoder.py:168
           Epoch 11/50: avg_loss = 0.8645                   build_dual_encoder.py:168
           Epoch 12/50: avg_loss = 0.8411                   build_dual_encoder.py:168
[18:43:46] Epoch 13/50: avg_loss = 0.8368                   build_dual_encoder.py:168
           Epoch 14/50: avg_loss = 0.9173                   build_dual_encoder.py:168
[18:43:47] Epoch 15/50: avg_loss = 0.7861                   build_dual_encoder.py:168
           Epoch 16/50: avg_loss = 0.8485                   build_dual_encoder.py:168
           Epoch 17/50: avg_loss = 0.9321                   build_dual_encoder.py:168
[18:43:48] Epoch 18/50: avg_loss = 0.8084                   build_dual_encoder.py:168
           Epoch 19/50: avg_loss = 0.8473                   build_dual_encoder.py:168
[18:43:49] Epoch 20/50: avg_loss = 0.8168                   build_dual_encoder.py:168
           Epoch 21/50: avg_loss = 0.8500                   build_dual_encoder.py:168
           Epoch 22/50: avg_loss = 0.8051                   build_dual_encoder.py:168
[18:43:50] Epoch 23/50: avg_loss = 0.8235                   build_dual_encoder.py:168
           Epoch 24/50: avg_loss = 0.8166                   build_dual_encoder.py:168
[18:43:51] Epoch 25/50: avg_loss = 0.7805                   build_dual_encoder.py:168
           Epoch 26/50: avg_loss = 0.8312                   build_dual_encoder.py:168
           Epoch 27/50: avg_loss = 0.8154                   build_dual_encoder.py:168
[18:43:52] Epoch 28/50: avg_loss = 0.7869                   build_dual_encoder.py:168
           Epoch 29/50: avg_loss = 0.8036                   build_dual_encoder.py:168
           Epoch 30/50: avg_loss = 0.8119                   build_dual_encoder.py:168
[18:43:53] Epoch 31/50: avg_loss = 0.8182                   build_dual_encoder.py:168
           Epoch 32/50: avg_loss = 0.7905                   build_dual_encoder.py:168
[18:43:54] Epoch 33/50: avg_loss = 0.8119                   build_dual_encoder.py:168
           Epoch 34/50: avg_loss = 0.8068                   build_dual_encoder.py:168
           Epoch 35/50: avg_loss = 0.7864                   build_dual_encoder.py:168
[18:43:55] Epoch 36/50: avg_loss = 0.7822                   build_dual_encoder.py:168
           Epoch 37/50: avg_loss = 0.7896                   build_dual_encoder.py:168
[18:43:56] Epoch 38/50: avg_loss = 0.8071                   build_dual_encoder.py:168
           Epoch 39/50: avg_loss = 0.7943                   build_dual_encoder.py:168
           Epoch 40/50: avg_loss = 0.8028                   build_dual_encoder.py:168
[18:43:57] Epoch 41/50: avg_loss = 0.7734                   build_dual_encoder.py:168
           Epoch 42/50: avg_loss = 0.7872                   build_dual_encoder.py:168
[18:43:58] Epoch 43/50: avg_loss = 0.7870                   build_dual_encoder.py:168
           Epoch 44/50: avg_loss = 0.7683                   build_dual_encoder.py:168
           Epoch 45/50: avg_loss = 0.8059                   build_dual_encoder.py:168
[18:43:59] Epoch 46/50: avg_loss = 0.7832                   build_dual_encoder.py:168
           Epoch 47/50: avg_loss = 0.7880                   build_dual_encoder.py:168
[18:44:00] Epoch 48/50: avg_loss = 0.8034                   build_dual_encoder.py:168
           Epoch 49/50: avg_loss = 0.7743                   build_dual_encoder.py:168
           Epoch 50/50: avg_loss = 0.7863                   build_dual_encoder.py:168
[18:44:01] Saved dual encoders.                             build_dual_encoder.py:174
[18:44:01] Indexing 10 SOP texts                                    index_utils.py:76
           Looking for trained models in:                           index_utils.py:88
           C:\Users\sikka\Documents\2025_IMP\Courses\Fall_25\DL
           \Project\artifacts\retriever_bert
           SOP encoder path:                                        index_utils.py:89
           C:\Users\sikka\Documents\2025_IMP\Courses\Fall_25\DL
           \Project\artifacts\retriever_bert\sop_encoder
           Tokenizer path:                                          index_utils.py:90
           C:\Users\sikka\Documents\2025_IMP\Courses\Fall_25\DL
           \Project\artifacts\retriever_bert\tokenizer
           SOP encoder exists: True                                 index_utils.py:91
           Tokenizer exists: True                                   index_utils.py:92
           Loading trained SOP encoder and tokenizer                index_utils.py:95
           Loading tokenizer and model (may download on first       index_utils.py:103
           run)...
Embedding (BERT) ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 100% 0:00:00
[18:44:02] Built embeddings with BERT.                              index_utils.py:117
[18:44:03] FAISS index saved to                                     index_utils.py:135
           C:\Users\sikka\Documents\2025_IMP\Courses\Fall_25\D
           L\Project\artifacts\retriever_bert\index\index.fais
           s
[18:44:03] Artifacts saved under                            build_dual_encoder.py:183
           C:\Users\sikka\Documents\2025_IMP\Courses\Fa
           ll_25\DL\Project\artifacts\retriever_bert

(SOP) C:\Users\sikka\Documents\2025_IMP\Courses\Fall_25\DL\Project>
```

```
[19:27:33] ✓ Index built with 100 SOPs                    demo.py:30

(SOP) C:\Users\sikka\Documents\2025_IMP\Courses\Fall_25\DL\P
roject>python src/cli/demo.py retrieve --q "Machine pressure
 is low"
[19:31:22] Looking for trained models in:          index_utils.py:178
           C:\Users\sikka\Documents\2025_
           IMP\Courses\Fall_25\DL\Project
           \artifacts\retriever_bert
           Incident encoder path:                  index_utils.py:179
           C:\Users\sikka\Documents\2025_
           IMP\Courses\Fall_25\DL\Project
           \artifacts\retriever_bert\inci
           dent_encoder
           Incident encoder exists: True           index_utils.py:180
           Loading trained incident                index_utils.py:183
           encoder
{
    'sop_id': 'SOP-001',
    'score': 0.8501064777374268,
    'text': 'Low Pressure Warning. Blue indicator is
flashing. Steps: Walk to Machine ; Read pressure_sensor ; If
below threshold, press blue_diagnostic_button ; Wait 3
seconds ; Re-read pressure_sensor ; Notify tech if
unchanged'
}
{
    'sop_id': 'SOP-018',
    'score': 0.766273558139801,
    'text': 'Flow Rate Error. Low flow rate indicator
active. Steps: Walk to Machine ; Read flow_sensor ; Press
flow_reset_button ; Wait 2 seconds'
}
{
    'sop_id': 'SOP-064',
    'score': 0.7444010972976685,
    'text': 'Replace Machine Intake Cover. Machine intake
cover missing. Steps: Walk to Table 2 ; Pick intake_cover ;
Walk to Machine ; Place cover on intake_port'
}
{
    'sop_id': 'SOP-017',
    'score': 0.3972323536872864,
    'text': 'System Heartbeat Lost. Heartbeat/status LED
dark. Steps: Walk to Machine ; Press system_wake_button ;
Wait 5 seconds ; Confirm heartbeat LED on'
}
{
    'sop_id': 'SOP-014',
    'score': 0.3545655310153961,
    'text': 'Cooling Fan Check. Cooling fan speed low
warning. Steps: Walk to Machine ; Read fan_speed_sensor ;
Press fan_boost_button ; Wait 3 seconds ; Check sensor
again'
}

(SOP) C:\Users\sikka\Documents\2025_IMP\Courses\Fall_25\DL\P
roject>
```