Standard Operating Procedure: Offline LLM Dataset Generation

Version: 1.0 **Target Environment:** Isolated / Air-Gapped Systems (Ship-Board) **Security Classification:** Unclassified / Internal Use Only

1. Executive Summary

This protocol defines the procedure for generating high-quality "Question-Answer" training datasets from raw PDF technical manuals within a strictly offline (air-gapped) environment.

The system utilizes a locally hosted Neural Network (T5-Base Transformer) to analyze text segments and reverse-engineer natural language questions, eliminating the need for external APIs (e.g., OpenAI, Gemini) or internet connectivity during the generation phase.

2. System Architecture

- **Model:** mrm8488/t5-base-finetuned-question-generation-ap (Local Hugging Face Model).
- **Input:** Technical PDF Documents.
- **Output:** JSON formatted dataset (train_offline.json) ready for LLM fine-tuning.
- **Dependencies:** PyTorch, Transformers, PyMuPDF, SentencePiece, Protobuf.

3. Phase I: Shore-Side Preparation

Status: Internet Connection REQUIRED. **Objective:** Securely download model weights and software libraries for transfer.

Step 3.1: Create Workspace

On your internet-connected terminal, create a project folder:

Bash

```
mkdir Defense_LLM_Gen

cd Defense_LLM_Gen
```

Step 3.2: The Asset Downloader Script

Save the following code as **download_assets.py**. This script automates the retrieval of the AI model and the specific .whl installation files required for offline use.

Python

```
# FILE: download_assets.py
```

```
import os
import subprocess
import sys


def install_prereqs():
    """Installs libraries needed for the download process itself."""
    print(">>> Checking prerequisites...")
    subprocess.check_call([sys.executable, "-m", "pip", "install", "transformers", "sentencepiece", "torch"])


def download_model():
    """Downloads the T5 Model and Tokenizer to a local folder."""
    try:
        from transformers import AutoTokenizer, AutoModelForSeq2SeqLM
    except ImportError:
        install_prereqs()
        from transformers import AutoTokenizer, AutoModelForSeq2SeqLM

    model_name = "mrm8488/t5-base-finetuned-question-generation-ap"
    save_directory = "./offline_model"

    print(f"\n>>> Downloading AI Model: {model_name}")
    tokenizer = AutoTokenizer.from_pretrained(model_name)
    model = AutoModelForSeq2SeqLM.from_pretrained(model_name)

    print(f">>> Saving Model to '{save_directory}'...")
    tokenizer.save_pretrained(save_directory)
    model.save_pretrained(save_directory)
    print("✅ Model Download Complete.")
```

```
def download_libs():
    """Downloads .whl files for offline installation."""
    libs_dir = "./offline_libs"
    if not os.path.exists(libs_dir):
        os.makedirs(libs_dir)

    print(f"\n>>> Downloading Offline Installers to '{libs_dir}'")
    # Exact list of required libraries
    packages = ["torch", "transformers", "sentencepiece", "protobuf", "pymupdf"]

    # Download commands
    subprocess.check_call([sys.executable, "-m", "pip", "download"] + packages + ["-d", libs_dir])
    print("✅ Library Download Complete.")

if __name__ == "__main__":
    download_model()
    download_libs()
    print("\n[SUCCESS] PREPARATION COMPLETE.")
    print("You may now transfer this entire folder to the secure drive.")
```

Step 3.3: Execution

Run the script to fetch all assets:

Bash

```
python download_assets.py
```

4. Phase II: Transfer Protocol

Objective: Migrate assets to the isolated system.

Ensure the external drive contains the following **exact directory structure**:

Plaintext

```
/Defense_LLM_Gen/
│
├── offline_model/     <-- (Folder) Contains pytorch_model.bin, config.json, etc.
├── offline_libs/      <-- (Folder) Contains .whl files (torch, transformers, etc.)
├── st_notes.pdf       <-- (File) Your target PDF document.
└── ship_generator.py  <-- (File) The generation script (Code provided in Phase III).
```

5. Phase III: Ship-Side Deployment

Status: Strictly OFFLINE. **Objective:** Install environment and generate dataset.

Step 5.1: The Generator Script

Save the following code as **ship_generator.py** on the drive. This is the executable logic for the ship's system.

Python

```
# FILE: ship_generator.py
import fitz  # PyMuPDF
import torch
from transformers import AutoTokenizer, AutoModelForSeq2SeqLM
import json
import re
import sys
import os

# CONFIGURATION
MODEL_PATH = "./offline_model"
PDF_PATH = "st_notes.pdf"
OUTPUT_FILE = "train_offline.json"

def load_offline_model():
    """Loads model strictly from local files."""
    print(f"Loading AI Model from {MODEL_PATH}...")
    if not os.path.exists(MODEL_PATH):
        print(f"CRITICAL ERROR: Directory '{MODEL_PATH}' missing.")
        sys.exit(1)

    try:
        # local_files_only=True ensures no internet connection is attempted
        tokenizer = AutoTokenizer.from_pretrained(MODEL_PATH, local_files_only=True)
        model = AutoModelForSeq2SeqLM.from_pretrained(MODEL_PATH, local_files_only=True)
        return tokenizer, model
    except Exception as e:
        print(f"Error loading model: {e}")
        print("Ensure 'protobuf' and 'sentencepiece' are installed.")
        sys.exit(1)

def extract_text(pdf_path):
    """Parses PDF content."""
    try:
        doc = fitz.open(pdf_path)
        text = ""
        for page in doc:
            text += page.get_text() + " "
        return text
    except Exception as e:
        print(f"Error reading PDF: {e}")
        sys.exit(1)

def clean_and_chunk(text, chunk_size=150):
    """Cleans text and groups it into context windows."""
    text = re.sub(r'\s+', ' ', text).strip()
    sentences = re.split(r'(?<=[.!?])\s+', text)

    chunks = []
    current_chunk = []
    current_count = 0

    for sentence in sentences:
        current_chunk.append(sentence)
        current_count += len(sentence.split())
        if current_count >= chunk_size:
            chunks.append(" ".join(current_chunk))
            current_chunk = []
            current_count = 0
    if current_chunk:
        chunks.append(" ".join(current_chunk))
    return chunks

def generate_question(tokenizer, model, context, answer):
    """AI Generation Logic."""
    input_text = f"answer: {answer}  context: {context}"
    inputs = tokenizer(input_text, return_tensors="pt", max_length=512, truncation=True)

    # Generation parameters
    outputs = model.generate(
        inputs["input_ids"],
        max_length=64,
        num_beams=4,
        early_stopping=True
    )

    question = tokenizer.decode(outputs[0], skip_special_tokens=True)
    return question.replace("question:", "").strip()

def main():
    # 1. User Interface
    try:
        limit = int(input("Enter number of questions to generate: "))
    except ValueError:
        limit = 10

    # 2. Initialization
    tokenizer, model = load_offline_model()
    raw_text = extract_text(PDF_PATH)
    chunks = clean_and_chunk(raw_text)

    dataset = []
    print(f"\n>>> Processing {len(chunks)} text segments for {limit} Q&A pairs...")

    # 3. Processing Loop
    for i, chunk in enumerate(chunks):
        if len(dataset) >= limit:
            break

        sentences = re.split(r'(?<=[.!?])\s+', chunk)
        # Filter for substantial answers (exclude short headers)
        valid_answers = [s.strip() for s in sentences if len(s.split()) > 10]
        if not valid_answers:
            continue

        target_answer = valid_answers[0]
        generated_q = generate_question(tokenizer, model, chunk, target_answer)

        if len(generated_q) > 10:
            print(f"[{len(dataset)+1}] Generated: {generated_q}")
            dataset.append({
                "instruction": generated_q,
                "output": target_answer
            })

    # 4. Save Output
    with open(OUTPUT_FILE, "w", encoding="utf-8") as f:
        json.dump(dataset, f, indent=2, ensure_ascii=False)

    print(f"\n[SUCCESS] Dataset saved to {OUTPUT_FILE}")

if __name__ == "__main__":
    main()
```

Step 5.2: Offline Library Installation

On the secure machine, open the terminal in the folder and run:

Bash

```
pip install --no-index --find-links=./offline_libs torch transformers sentencepiece protobuf pymupdf
```

Note: The --no-index flag forces pip to look ***only*** *in the local folder, preventing any attempt to connect to the internet.*

Step 5.3: Operation

Run the generator:

Bash

```
python ship_generator.py
```

Follow the on-screen prompt to specify the number of Q&A pairs required.

6. Troubleshooting

Error: ImportError: T5Converter requires the protobuf library **Cause:** The protobuf library was not installed. **Fix:** Ensure you ran the installation command in Step 5.2 exactly as written. Verify the protobuf .whl file exists inside the offline_libs folder.

Error: OSError: Can't find model ... **Cause:** The offline_model folder is missing or named incorrectly. **Fix:** Ensure the folder is named exactly offline_model and is in the same directory as the Python script.