DEVNEURON

Software Engineer Intern Assessment

Congratulations on making it to the assessment round of DevNeuron!

Objective

This assessment is designed to evaluate your ability to learn new concepts, solve problems, handle ambiguity, and deliver results. For this purpose, you will develop adversarial attacks targeting AI models, as well as assess the robustness of those models when subjected to such attacks. The tasks will focus on both theoretical understanding and practical implementation of adversarial machine learning techniques.

This assessment also evaluates your technical abilities to:

- Build and expose a machine learning–powered API with **FastAPI**
- Develop a simple **Next.js frontend** that consumes the API
- Deploy the system on **AWS Free Tier** (Amplify, Lambda, or EC2)
- Communicate results clearly and professionally

General Instructions:

- **Preferred Programming Language**: Python, JavaScript
- You are encouraged to use any libraries or frameworks that you deem appropriate for completing the tasks. We would like you to use Next.js for the frontend, FASTAPI for the backend, and PyTorch for Machine Learning related tasks.
- You are expected to attempt **all questions to the best of your ability**.
- The total time required to complete this assessment may not exceed **7 hours**. However, we have given you 7 days to work at your own time.

- Please include **screenshots of the execution output** for each implementation step or deliverable.

- Submit all files that are **explicitly mentioned as required** in the assessment description. Submit the URL of the exposed API and the frontend application

- Concepts and technologies introduced in this assessment may be discussed further in a follow-up interview. Therefore, we strongly encourage you to complete the assignment independently and gain a solid understanding of the techniques and tools used.

- If you refer to external resources, open-source code, or third-party implementations, please ensure proper attribution by providing **references or links** to the source.

Tasks

Part 1 — Backend (FastAPI + ML)

Goal: Implement Fast Gradient Sign Method (FGSM) in a REST API.

1. Study the Fast Gradient Sign Method (FGSM):
   Begin by reviewing the original paper by Goodfellow et al., where the FGSM adversarial attack was introduced. Familiarize yourself with the core intuition and mathematical formulation of the attack.

2. Implement FGSM in PyTorch:
   Implement the FGSM attack function in PyTorch and save the implementation in a script titled `fgsm.py`. Encapsulate this function into the `Attack` Class.

3. Model Development and Attack Evaluation:

    - Evaluate the robustness of any pretrained model (e.g., MNIST model) by applying `fgsm.py` attacks.

    - Record the output (e.g., accuracy drop due to an attack).

    - Submit only the resulting **output files and screenshots**. **Do not submit any datasets**.

4. Create a FastAPI service with one endpoint in `app_fgsm.py`:

- `POST /attack`:
    - Input: uploaded image (PNG/JPEG) and epsilon (default = 0.1).
    - Output (JSON):
        - Clean Prediction: Prediction of the model on the original image
        - Adversarial Prediction: Prediction of the model on adversarial FGSM Image
        - Base64 adversarial image
        - Attack Success Status

5. Keep the implementation modular and documented.

Part 2 — Frontend (Next.js)

Goal: Build a UI for demonstration. Feel free to be creative here if your interest lies in frontend.

- A single-page app with:
    - File upload for an image
    - Numeric input or slider for epsilon
    - Button to run the attack
    - Display of attack success, clean vs adversarial predictions, and both images side-by-side

---

Part 3 — AWS Deployment

Goal: Deploy both frontend and backend using AWS Free Tier. Feel free to choose any. The following are our recommended options:

Options:

- **Frontend**:
    - Deploy Next.js static site with **Amplify Hosting** (Free Tier includes 1,000 build minutes/month, 5 GB stored, 15 GB served for 12 months).
- **Backend**:
    - Option A: Deploy FastAPI with **AWS Lambda + API Gateway** (serverless, covered under Always Free tier).
    - Option B: Deploy FastAPI on **EC2 t2.micro** (covered under 12-month Free Tier).

Document which option you chose and why. If you are unable to use AWS for any reason, use the Render free tier for deployment. Please note that the Render deployment will only get half marks on this task. If you are unable to deploy, submit localhost screenshots to show the working of the application.

Part 4 — Documentation

Provide a **README.md** including:

- How to run locally
- Deployed URLs (frontend + backend)
- Explanation of FGSM (1–2 paragraphs, in your own words)
- Observations: How did predictions change? Did increasing epsilon make attacks stronger?

Deliverables

- Zip everything and include:
    - `backend/` (FastAPI code + FGSM + requirements)
    - `frontend/` (Next.js app)

    - `README.md` with setup and deployment instructions

- **Deployed app links** on AWS Free Tier
- Relevant Screenshots of server-side deployments
- Relevant screenshots to show that each task is running on your end and on the server.
- Do not include heavy model, environment, and data files in your submissions.
- Upload the zip file to Google Drive and send it to hr@devneuron.com