Cronograma Detalhado – Projeto Indoor Localization com GANs

Objetivo Geral do Projeto:

> Desenvolver, em Python (via Jupyter Notebook), a implementação reprodutiva do artigo "Indoor Localization Using Data Augmentation via Selective GANs", com base em dados simulados e reais (UJIndoorLoc), incluindo todas as etapas metodológicas, tabelas e gráficos do estudo.

Acompanhe as fases também por esta planilha: Planilha no Google Drive

---

Etapa 1: Configuração Inicial e Preparação de Dados

Duração estimada: 1 a 2 dias úteis

Objetivos Técnicos:

- Estruturar o projeto com diretórios organizados (`data/`, `models/`, `outputs/`, `notebooks/`).
- Criar ambiente reprodutível com `requirements.txt` ou `environment.yml`.
- Instalar bibliotecas principais (TensorFlow, Pandas, NumPy, Matplotlib).
- Baixar, explorar e filtrar o dataset **UJIndoorLoc** (foco em Building 1, Floor 2).
- Implementar código base para simulação dos vetores RSSI (modelo de propagação do sinal).

Entregáveis:

- Estrutura de projeto pronta e organizada.
- Script de download e pré-processamento do UJIndoorLoc.
- Código base de geração de dados simulados com fórmulas documentadas.
- Documento curto (Markdown) explicando a base real e parâmetros da simulação.

Etapa 2: Implementação e Treinamento do GAN

Duração estimada: 3 a 4 dias úteis

Objetivos Técnicos:

- Implementar o **Generator** e o **Discriminator** conforme artigo (1 camada escondida com 10 neurônios cada).

- Utilizar função de perda binária com otimizador Adam (lr = 0.01).
- Treinar com 1000 vetores reais simulados.
- Gerar 40.000 vetores sintéticos ao final.
- Validar graficamente e numericamente a coerência dos dados gerados.

Entregáveis:

- Script completo do GAN (treinamento, geração).
- Gráficos de perda do Discriminador e do Generator por época.
- Dispersão visual dos dados gerados versus reais.
- Salvar modelo do GAN para reuso.

Etapa 3: Pseudo-rotulação e Seleção Inteligente

Duração estimada: 2 a 3 dias úteis

Objetivos Técnicos:

- Treinar rede DNN supervisionada com os dados reais (RSSI $\rightarrow$ coordenadas).
- Aplicar pseudo-label nos vetores gerados com o modelo treinado.
- Dividir o ambiente em zonas (ex: $1m^2$) para aplicar **Critério 1 – Cobertura**.
- Avaliar score de realismo via Discriminador para aplicar **Critério 2 – Confiança**.
- Selecionar subconjunto de dados sintéticos de maior qualidade para etapa final.

Entregáveis:

- Script de pseudo-labeling com o modelo DNN.
- Lógica de seleção com logs por zona e histogramas de confiança.
- Lista de vetores gerados selecionados para o treinamento final.
- Visualizações: mapa de cobertura espacial, distribuição de confiança.

Etapa 4: Modelo Final de Localização e Avaliação

Duração estimada: 2 a 3 dias úteis

Objetivos Técnicos:

- Treinar modelo final de localização (DNN com 2 camadas ocultas: 30 e 20 neurônios).
- Comparar o desempenho com:

  - Apenas dados reais
  - Dados reais + gerados sem seleção
  - Dados reais + gerados com seleção
- Calcular métricas: erro médio de localização, erro mínimo e máximo.
- Reproduzir visualizações: CDF do erro, Tabela 2 e Figura 6 do artigo.

Entregáveis:

- Script de treinamento do modelo final com conjunto expandido.
- Tabelas de comparação de desempenho (formato `.csv` e `.png`).
- Gráficos de erro (CDF, dispersão espacial).
- Modelo salvo para deploy ou análise futura.

Etapa 5: Documentação Final e Entrega

Duração estimada: 1 a 2 dias úteis

Objetivos Técnicos:

- Consolidar todas as etapas em um único Jupyter Notebook limpo e comentado.
- Garantir reprodutibilidade com documentação de ambiente (`requirements.txt` ou `.yml`).
- Empacotar todos os arquivos (dados, modelos, resultados) em `.zip`.
- (Opcional) Criar **PDF técnico** com sumário executivo dos resultados.

Entregáveis:

- `notebook_final.ipynb` com todas as etapas documentadas.
- Arquivo `.zip` com estrutura completa e instruções de uso.
- PDF com gráficos e tabelas (se solicitado).
- Prontidão para apresentação ou demo técnica.