Documentación del Proceso de Desarrollo del Proyecto

Introducción

El desarrollo de este proyecto comenzó realizando una consulta inicial sobre cómo abordar la predicción del engagement de puntos de interés turísticos (POIs) utilizando imágenes y metadatos. Desde ese punto, se estableció un camino iterativo de aprendizaje, exploración y mejora continua, propio de un enfoque científico. Cada paso estuvo marcado por decisiones estratégicas, ajustes técnicos y aprendizajes significativos.

Primera Etapa: Conceptualización y Exploración

Objetivo inicial

El propósito era construir un modelo que combinara imágenes y metadatos para clasificar los POIs en dos categorías: alto o bajo engagement.

El desafío inicial fue comprender cómo estructurar un enfoque híbrido que aprovechara las fortalezas de las redes neuronales convolucionales (CNN) y técnicas de procesamiento de datos tabulares.

Primeros retos

1. **Elección de la arquitectura**:
    - Se debatió entre varias arquitecturas preentrenadas como ResNet, EfficientNet y combinaciones de ambas.
    - El primer paso fue seleccionar ResNet18 como base, por ser liviana y adecuada para transfer learning en conjuntos de datos medianos.
    - El segundo paso fue continuar con la experimentación probando EfficientNet.
2. **Definición de etiquetas**:
    - Se decidió generar etiquetas binarias a partir de las proporciones de `Likes` y `Visits`. Sin embargo, el cálculo inicial enfrentó problemas debido a divisiones por cero, lo que llevó a agregar un pequeño término (`1e-5`) para evitar errores matemáticos.

Segunda Etapa: Implementación Técnica

Diseño del dataset

Se desarrolló una clase personalizada para manejar tanto las imágenes como los metadatos.

Los primeros intentos incluyeron referencias explícitas a nombres de columnas (`label`, `image_id`), lo que generó varios **errores de clave (`KeyError`)** debido a inconsistencias en los nombres de las columnas del CSV.

Lecciones aprendidas**:**

1. **Flexibilidad en la codificación**:
    - Se implementó una lógica que detectara dinámicamente los nombres de las columnas en el archivo CSV para evitar errores futuros.
2. **Normalización de metadatos**:

- Se utilizó `StandardScaler` para escalar las características numéricas como `locationLat` y `locationLon`.

Éxitos**:**

- La clase `POIDataset` comenzó a combinar correctamente imágenes y metadatos para ser utilizadas en redes neuronales.
- Se añadió data augmentation a las imágenes mediante transformaciones como redimensionamiento, normalización y aumentos aleatorios.

Fracasos tempranos**:**

1. **Problemas de compatibilidad**:
    - Hubo dificultades al procesar imágenes debido a tamaños inconsistentes. Esto se resolvió ajustando todas las imágenes a 224x224 píxeles.
2. **Combinación de datos**:
    - El intento inicial de fusionar características de imágenes y metadatos en un único tensor generó errores dimensionales. Se corrigió concatenando las salidas procesadas de ambas fuentes antes de pasar por las capas completamente conectadas.

Tercera Etapa: Optimización del Modelo

Decisiones clave**:**

1. **Uso de ResNet18 y EfficientNet**:
    - Aunque ResNet18 era adecuada para la mayoría de los experimentos iniciales, se exploró la posibilidad de combinarla con EfficientNet para mejorar la capacidad del modelo.
    - Finalmente, se optó por mantener solo ResNet18 debido a su simplicidad y a que los resultados eran competitivos.
2. **Congelación de capas**:
    - Para reducir el tiempo de entrenamiento, se decidió congelar las capas iniciales de ResNet18, entrenando únicamente las capas finales y la subred de metadatos.
3. **Función de pérdida y optimizador**:
    - Se utilizó `BCELoss` para la clasificación binaria y Adam como optimizador por su capacidad para manejar tasas de aprendizaje dinámicas.

Éxitos**:**

- El modelo logró métricas iniciales de evaluación razonables: **F1-score: 0.5551, Accuracy: 0.6274**, aunque con un recall bajo.
- Se implementaron scripts de utilidad para dividir y procesar los datos automáticamente.

Fracasos y aprendizajes**:**

1. **Desbalance de clases**:
    - El modelo mostró un sesgo hacia la clase mayoritaria. Se discutió implementar técnicas de balanceo como oversampling o ajuste de pesos en la función de pérdida.

2. **Resultados no reproducibles**:
    - La falta de control sobre semillas aleatorias inicialmente generó resultados inconsistentes. Esto se solucionó fijando una semilla global para PyTorch, NumPy y los generadores de datos.

Cuarta Etapa: Integración y Reproducibilidad

Documentación técnica

- Se generaron scripts de preprocesamiento para normalizar los datos y transformar imágenes.
- Se desarrolló un informe técnico que justifica cada decisión, incluyendo la elección de ResNet18 y la metodología para combinar imágenes y metadatos.

Notable éxito**:**

El proyecto alcanzó una estructura modular y reproducible:

1. **Dataset personalizado**: Combina imágenes y metadatos de forma dinámica.
2. **Modelo extensible**: Permite integrar otras arquitecturas fácilmente.
3. **Evaluación robusta**: Métricas como F1-score, precisión y recall están bien implementadas.

Conclusión

Este proyecto demostró ser un ejercicio riguroso de exploración científica y aprendizaje iterativo. Aunque enfrentó varios desafíos técnicos (errores de clave, incompatibilidades de datos), cada problema condujo a mejoras significativas en la arquitectura y la metodología.

El producto final es un modelo funcional que utiliza redes neuronales convolucionales y subredes para metadatos, respaldado por un conjunto de herramientas reproducibles. Las decisiones tomadas fueron guiadas por principios de escalabilidad, simplicidad y rendimiento.

Próximos pasos:

- Implementar técnicas de ajuste de hiperparámetros (Optuna).
- Explorar arquitecturas híbridas adicionales, como Vision Transformers.
- Optimizar el modelo para su uso en producción.