Práctica Deep Learning – Predicción de Engagement en POIs Turísticos

Leticia Cabañas Morales
KeepCoding – Junio 2025

3. Metodología

Reproducibilidad y configuración inicial

Para garantizar la reproducibilidad de los experimentos, se establecieron semillas aleatorias para los distintos entornos (random, numpy y torch), además de fijar los parámetros de CUDNN para obtener resultados deterministas. También se comprobó automáticamente la disponibilidad de GPU para aprovechar la aceleración por hardware durante el entrenamiento del modelo.

Se han instalado de dependencias con requirements.txt. Se han importado las librerías clave, y se ha cargado de module_utils.py las funciones auxiliares utilizadas en la práctica.

3.1 Preparación y Análisis de Datos

Carga y exploración inicial del dataset

Se cargó el dataset poi_dataset.csv, compuesto por 1569 registros y 14 columnas. Se verificó que no existen valores nulos. Posteriormente, se eliminaron las columnas id, name y shortDescription por no aportar valor directo o requerir tratamiento NLP avanzado.

Se llevó a cabo un análisis estadístico y visual de las variables numéricas, identificando distribuciones sesgadas y escalas heterogéneas. Por este motivo, se aplicaron transformaciones logarítmicas a las métricas de engagement (Visits, Likes, Dislikes y Bookmars) antes de definir la variable objetivo, y se normalizaron las variables numéricas.

También se observó que la variable *Visits* presenta muy poca dispersión en los datos. Por el contrario, *Likes*, *Dislikes* y *Bookmarks*, sí presentan más variabilidad, esto nos haría pensar, a priori, que podrían tendrán más peso en nuestra variable objetivo.

Análisis y preprocesamiento de variables categóricas

Las columnas categories y tags contenían listas representadas como cadenas. Se transformaron a listas reales para su posterior procesamiento. El análisis mostró:

categories: La mayoría de los POIs tiene exactamente 3 categorías. Se aprecia un desbalanceo evidente.
tags: Hay un número muy elevado de etiquetas diferentes. Presentan una distribución bastantes desbalanceada, la gran mayoría de los POIs tienen varios tags, especialmente 10 y 13. Se observaron algunas listas vacías en ambas variables, pero que no fueron tratadas de ninguna manera.

Dado que muchos POIs presentan múltiples categorías, se optó por una representación mediante multi-hot encoding, generando vectores binarios por muestra donde cada posición indica la presencia o ausencia de

una categoría o etiqueta específica. Esta codificación resulta especialmente útil en tareas de clasificación donde la semántica contextual juega un papel importante.

En el caso de los tags, debido a su alta cardinalidad y distribución desbalanceada, se seleccionaron únicamente los 10 más frecuentes para su representación individual. El resto de etiquetas se agruparon en una columna adicional tag_other. Esta estrategia permite capturar las etiquetas más representativas sin incrementar excesivamente la dimensionalidad del modelo, mejorando así la eficiencia y capacidad de generalización.

Distribución y correlaciones de variables numéricas

La variable tier se decidió utilizar como una variable ordinal, ya que sus valores reflejan distintos niveles de popularidad. Se verificó que el valor 1 se asociaba con un mayor número de Likes, indicando una mayor relevancia, mientras que el valor 4 correspondía a los POIs menos populares.

Las variables *xps* aunque es discreta, tiene un valor cuantitativo que el modelo puede explotar directamente, podemos tratarla como numérica.

Se evaluaron la distribución y relación entre métricas de engagement: Visits, Likes, Dislikes, y Bookmarks: Vemos como *Likes* y *Bookmarks* están fuertemente correlacionadas, como era de esperar, pero teniendo en cuenta, como hemos visto en la estadística, que los *Bookmarks* son mucho menos frecuentes, implican mayor intención.

También vemos como *Dislikes* tiene una correlación negativa moderada con respecto a *Likes* y *Bookmarks*. Deberá penalizar en la métrica de engagemet.

Y *Visits* prácticamente no tiene correlación con ninguna de las demás métricas. Parece que el número visitas no está directamente relacionado con que el POI guste o se guarde.

Se aplicó una transformación logarítmica a estas variables y se definió una métrica compuesta engagement_score, con pesos que reflejan la relevancia e intención detrás de cada acción. Se asignó mayor peso a Bookmarks (acción más comprometida), y se penalizó con fuerza con los Dislikes. Dado que la variable resultante tenía una distribución multimodal, se ha planteado como un problema de clasificación multiclase (engagement_level), dividiendo el score en tres nieveles de engagement (bajo=0, medio=1, alto=2) usando percentiles (Q1 y Q3). Esto permite una interpretación más clara y podría facilitar el aprendizaje del modelo. Se observó algo de desbalanceo en los tres niveles, y s decidió hacer una división de los datos estratificada para compensar.

Preprocesamiento de imágenes y metadatos

Las imágenes fueron redimensionadas a 224×224 píxeles para ser compatibles con arquitecturas preentrenadas como ResNet18, que es la red neuronal convolucional que finalmente se ha usado. Además, se normalizaron con las medias y desviaciones estándar de ImageNet.

Durante el entrenamiento, se aplicó data augmentation sobre las imágenes, incluyendo rotación aleatoria y alteración de color, con el objetivo de mejorar la robustez y generalización del modelo.

Las variables numéricas xps, locationLat y locationLon se estandarizaron con StandardScaler para facilitar el aprendizaje del modelo y prevenir que variables con escalas mayores dominen en el entrenamiento.

División del dataset y creación de los loaders

Se realizó una división estratificada del dataset en conjuntos de entrenamiento (70%), validación (15%) y test (15%) para preservar la distribución de clases.

Se definió una clase POIDataset personalizada para integrar tanto las imágenes como los metadatos en un mismo flujo de entrenamiento. Se validó su correcto funcionamiento con ejemplos individuales.

Se crearon los DataLoaders correspondientes con un batch size de 256. Además, se incorporaron class weights en la función de pérdida CrossEntropyLoss para mitigar el desbalance de clases observado en la variable objetivo (engagement_level).

3.2 Arquitectura del Modelo

Diseño modular y enfoque multimodal

El modelo final está compuesto por dos ramas principales: • Rama visual: Se utilizó una CNN preentrenada (ResNet18), eliminando su capa final de clasificación. Esta red permite extraer representaciones visuales profundas de las imágenes sin necesidad de entrenar desde cero. • Rama metadatos: Para procesar los metadatos estructurados, se definió un bloque MLP (Multilayer Perceptron) con dos capas fully connected, activación ReLU, normalización BatchNorm1d para estabilizar y acelerar el entrenamiento, y Dropout para evitar sobreajuste.

Las salidas de ambas ramas se concatenan y se pasan por un bloque adicional de clasificación (fusion_branch) compuesto por otra capa oculta, normalización y dropout, finalizando en una capa de salida de 3 unidades (correspondiente a las clases del engagement).

Prevención de data leakage

Se tuvo especial cuidado en evitar fuga de información entre los conjuntos de entrenamiento, validación y prueba. Las transformaciones de los metadatos (escalado) se ajustaron exclusivamente sobre el conjunto de entrenamiento, y se aplicaron posteriormente al resto.

El modelo fue probado con un batch del conjunto de entrenamiento para verificar la correcta integración de entradas y funcionamiento de la red. La salida fue una predicción en forma de tensor de 3 clases y un batch size de 256, como se esperaba.

3.3 Entrenamiento y Optimización

Se implementó una función POI_train_model que encapsula el pipeline de entrenamiento, evaluación y early stopping. Se entrenaron múltiples configuraciones del modelo variando hiperparámetros clave como el learning rate, dropout rate, número de épocas, función de activación y weight decay. También se aplicó la técnica de early stopping con paciencia de 3 a 5 épocas.

Se utilizó la función de pérdida CrossEntropyLoss ponderada con class weights para manejar el desbalance de clases. El optimizador Adam fue empleado en todos los entrenamientos.

Se documentaron cuidadosamente los resultados y métricas de cada entrenamiento, organizándolos en una tabla resumen que permite comparar el impacto de cada cambio. Finalmente, se seleccionó el modelo con mayor accuracy en validación y curvas de aprendizaje más estables.

Resumen de experimentos y configuración

ID	Dropout	Activation	Epochs	LR	WD	Mejor Val Acc	Comentario breve
1	0.3	ReLU	10	1e-3	1e-3	79.00%	Inicio base, picos en validación
2	0.3	ReLU	10	5e-4	1e-3	83.00%	Mejora por menor LR
3	0.3	ReLU	15	5e-4	1e-3	85.50%	Más epochs mejora estabilidad
4	0.4	ReLU	20	5e-4	1e-3	86.00%	Mayor regularización (dropout)
5	0.4	ELU	25	5e-4	1e-3	87.50%	Cambio de activación mejora rendimiento
6	0.5	ELU	25	5e-4	1e-3	**88.00%**	Mejor resultado, balance estabilidad y precisión

3.4 Evaluación y Análisis

El modelo seleccionado fue evaluado sobre el conjunto de test, obteniendo un accuracy del **87.29%**. El análisis de métricas muestra una buena precisión y recall en todas las clases, con especial solidez en la clase 0 (bajo engagement) y ligera menor sensibilidad en la clase 1.

Informe de clasificación:

```
              precision    recall  f1-score   support

           0      0.855     0.946     0.898       112
           1      0.879     0.785     0.829        65
           2      0.907     0.831     0.867        59

    accuracy                          0.873       236
   macro avg      0.881     0.854     0.865       236
weighted avg      0.875     0.873     0.872       236
```

Desde el punto de vista práctico, estos resultados implican que el modelo es capaz de discriminar entre distintos niveles de engagement con bastante precisión. La clase de mayor compromiso (2) se predice de forma precisa, lo cual es especialmente relevante para la toma de decisiones estratégicas en el contexto turístico.

Adicionalmente, se visualizaron predicciones reales del modelo sobre imágenes del conjunto de test, observándose 7 aciertos y 2 errores entre 9 ejemplos. Esto confirma que el modelo no sólo rinde bien a nivel cuantitativo, sino que también muestra una coherencia cualitativa en sus predicciones.

En trabajos futuros, podría explorarse la interpretabilidad del modelo (e.g., Grad-CAM sobre imágenes o SHAP para metadatos) y realizar un análisis de errores más exhaustivo para entender los casos de difícil predicción.