Desarrollo de un Modelo de Deep Learning para la Estimación del Engagement en Puntos de Interés Turísticos

1. Introducción

1.2. Objetivo

Este proyecto tiene como objetivo desarrollar un modelo de deep learning capaz de estimar el nivel de engagement potencial de diferentes puntos de interés (POIs) turísticos.

El principal desafío radica en predecir el grado de interacción que un POI generará en función de sus características visuales y sus metadatos. Esta capacidad resulta de gran valor estratégico para Artgonuts, permitiendo:

- Optimizar la selección y priorización de contenido para maximizar la interacción del usuario.
- Identificar patrones visuales y características correlacionadas con mayor engagement.
- Mejorar la experiencia del usuario destacando contenido más relevante y atractivo.
- Proporcionar insights data-driven para la toma de decisiones sobre nuevos POIs.

2. Metodología

2.1. Adquisición y Procesamiento de Datos

2.1.1. Dataset

El dataset utilizado incluye información sobre puntos de interés turísticos, incluyendo:

- Imágenes principales de los POIs organizadas en carpetas por ID.
- Metadatos (descripción, categorías, etc.).
- Métricas de *engagement* (visitas, *likes*, *dislikes*, *bookmarks*).

Los datos provienen directamente de la plataforma Artgonuts, asegurando su relevancia y realismo. Las imágenes las imágenes utilizadas han sido procesadas para

esta práctica, y sus originales proceden de terceros, entre ellos el portal de datos abiertos de la Comunidad de Madrid.

2.1.2. Preprocesamiento

Se aplicaron diversas técnicas de limpieza y transformación de datos:

- Combinación de campos equivalentes.
- Estandarización de textos, etiquetas y categorías (conversión a minúsculas, eliminación de caracteres especiales).
- Tokenización de textos.
- Indexación de tokens.
- Eliminación de palabras irrelevantes (stopwords).
- Escalado MinMax para delimitar los límites de clasificación la métrica de *engagement* en una escala conocida.
- Codificación Multiple Hot para variables categóricas.
- Data Augmentation
- Normalización

Data Augmentation

Se aplicaron las siguientes transformaciones:

- volteo horizontal,
- rotación de hasta 90º
- escalado
- desplazamiento,
- modificaciones de brillo y contraste,
- modificaciones de saturación,
- y modificaciones de tono.

La diversidad de transformaciones se debe la identificación de imágenes con ángulos de rotación de hasta 90º, así como de imágenes sobredimensionadas que han perdido sus bordes. También se observaron fotografías tanto oscuras como claras, enfocadas y desenfocadas, con colores intensos y apagados, entre otras variaciones. Por lo tanto, el modelo debe ser capaz de aprender a procesar imágenes bajo diversas condiciones.

Dado que el dataset presentaba un desbalance significativo en las muestras de engagement, se implementó un proceso de clonación para balancear las categorías:

- Engagement bajo: 963 muestras (sin clonación). Todo este set de muestras sufrió las transformaciones mencinaddas con un 50% de probabilidad.
- Engagement neutro: 9 muestras (clonadas 7 veces, generando 1152 muestras repartidas en 128 datasets). Los datasets fueron transformados de la siguiente forma:

#Dataset	Volteo horizontal	Brillo y contraste	Saturación	Rotación	Traslación	Escalado	Tono
Original	No	No	No	No	No	No	No
Clon 1	No	No	No	No	No	No	Sí
Clon 2	No	No	No	No	No	Sí	No
Clon 3	No	No	No	No	No	Sí	Sí
Clon 4	No	No	No	No	Sí	No	No
Clon 5	No	No	No	No	Sí	No	Sí
Clon 6	No	No	No	No	Sí	Sí	No
...			...	...	...	...	...
Clon 127	Sí	Sí	Sí	Sí	Sí	Sí	Sí

- Engagement alto: 32 muestras (clonadas 5 veces, generando 1024 muestras repartidas en 32 datasets). Los datasets fueron transformados de la siguiente forma:

#Dataset	Brillo y contraste	Saturación	Rotación	Traslación	Escalado
Original	No	No	No	No	No
Clon 1	No	No	No	No	Sí
Clon 2	No	No	No	Sí	No
Clon 3	No	No	No	Sí	Sí
Clon 4	No	No	Sí	No	No
Clon 5	No	No	Sí	No	Sí
Clon 6	No	No	Sí	Sí	No
...	...	...	...	...	...
Clon 31	Sí	Sí	Sí	Sí	Sí

A su vez, a cada uno de esos datasets se aplicó, con un 50 % de probabilidad, una transformación de volteo horizontal y una de cambio de tono.

2.2. Definición de la Métrica Objetivo

Dado que la interacción del usuario es más representativa en métricas como *Likes*, *Dislikes* y *Bookmarks* en comparación con las visitas, se definió la siguiente métrica de *engagement*:

$$Engagement = \frac{Likes + Bookmarks - Dislikes}{Visitas}$$

Esta métrica es una aproximación a la métrica de *engagement* de Redes Sociales en Marketing:

$$Engagement\ de\ RR.SS. = \frac{Likes + Comentario + Compartir}{Número\ de\ seguidores}$$

La clasificación del *engagement* se hizo siguiendo también una aproximación con la clasificación de los clientes en Marketing según el NPS (*Net Promoter Score*).

La métrica de NPS son puntuaciones enteras de 0 a 10, donde las puntuaciones de 0 a 6 son clientes detractores, de 7 u 8 son neutros, y de 9 o 10 son promotores.

Análogamente, definimos lo siguiente para el engagement:

- Un engagement inferior al 65% equivaldrá a uno bajo,
- uno mayor o igual que 65% y menor que 85% será neutro,
- y uno mayor o igual que el 85% será alto.

La definición de la métrica presentaba la necesidad de escalado de datos. Dado que los parámetros de escalación deben depender únicamente de los datos de entrenamiento y no de los de test, la definición de la métrica debía realizarse necesariamente después de la división del conjunto completo de datos en datos para entrenamiento, validación y test. Como consecuencia, las muestras no pudieron estratificarse durante la división de conjuntos.

2.3. Definición del Modelo y Parámetros

Se diseñó una red neuronal que recibe tres tipos de datos:

- **Imágenes** normalizadas en formato tensor de 3 canales y de tamaño 128 x 128.
- **Categorías** en formato Multiple Hot, en un tensor de 13 características.
- **Tokens indexados** en un tensor de tamaño 48.

La arquitectura incluye:

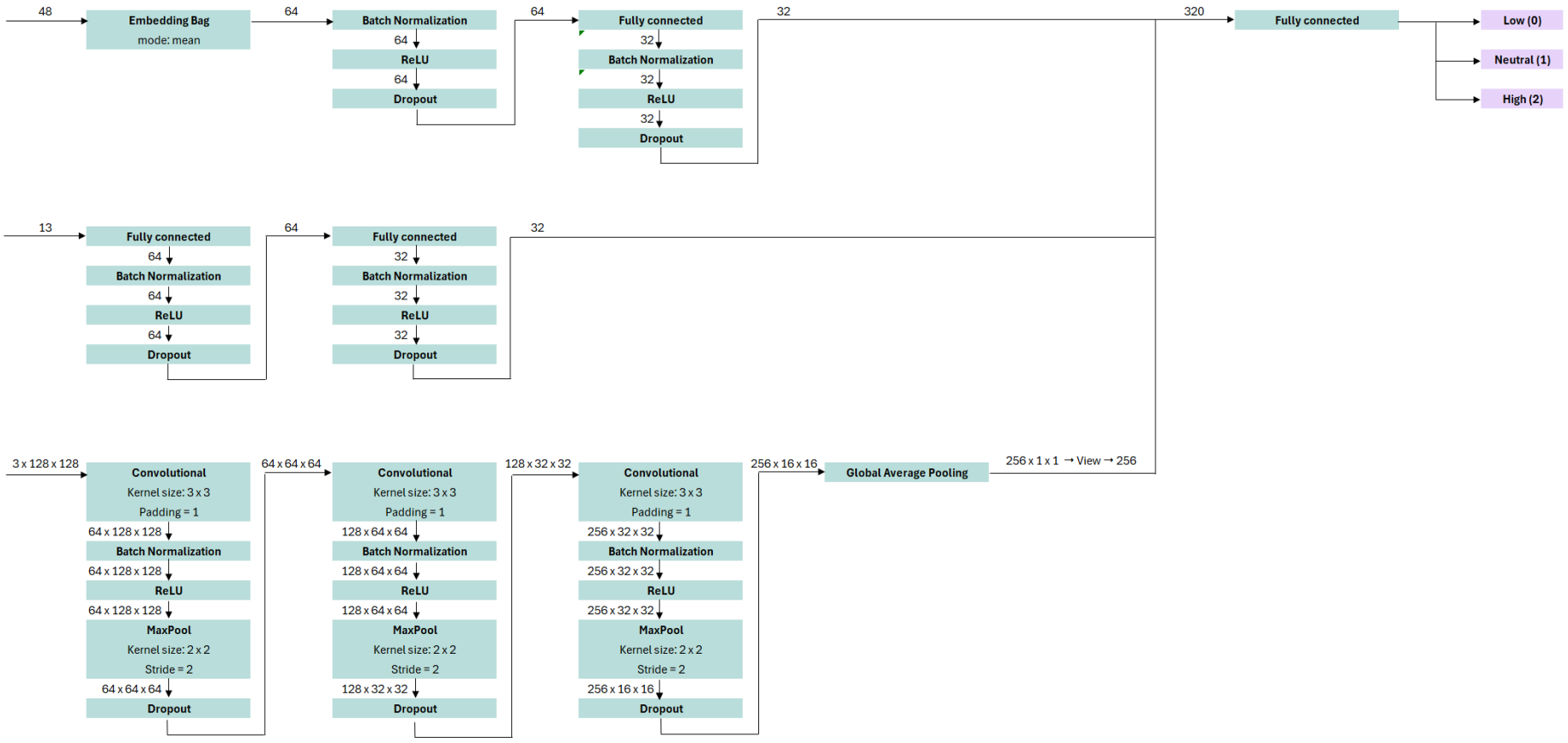
- Tres capas convolucionales con activación ReLU y MaxPooling. A su vez, para evitar el overfitting, cuentan con Batch Normalization y Dropout. Después se aplica Global Average Pooling para reducir el número en el proceso hacia la capa Fully Collected de clasificación final.
- Dos capas ReLU, ambas con Dropout y Batch Normalization para evitar el overfitting.
- Una capa de Bag Embedding mediante el modo “promedio”, seguida de dos activaciones ReLU con Batch Normalization y Dropout para evitar el overfitting.

- Batch Normalization y Dropout para prevenir overfitting.
- Global Average Pooling antes de la capa densa final.
- Una capa Fully Connected que une las tres ramas mencionadas anteriormente y realiza la clasificación fina.

Todos ratios de Dropout serán el mismo, siendo este un parámetro a optimizar.

En el punto donde se unen las 3 ramas (convolucional, simple y embedding), los datos provenientes de las imágenes tendrán un tamaño superior al de los datos provenientes de las categorías y los tokens (256 frente a 32). De esta forma, se pretende dar más importancia a las imágenes, ya que las estas presentan mayor variedad de datos (gracias a las transformaciones de Data Augmentation), mientras que los datos provenientes de las categorías y los tokens fueron clonados (cuando el engagement era alto o neutro) sin ser transformados posteriormente.

En la siguiente página se puede ver un esquema de la red neuronal y de las dimensiones de entrada y salida de los datos en cada capa.


48
Embedding Bag
mode: mean
64
Batch Normalization
64
ReLU
64
Dropout
64
Fully connected
32
Batch Normalization
32
ReLU
32
Dropout
32
320
Fully connected
Low (0)
Neutral (1)
High (2)
13
Fully connected
64
Batch Normalization
64
ReLU
64
Dropout
64
Fully connected
32
Batch Normalization
32
ReLU
32
Dropout
32
3 x 128 x 128
Convolutional
Kernel size: 3 x 3
Padding = 1
64 x 128 x 128
Batch Normalization
64 x 128 x 128
ReLU
64 x 128 x 128
MaxPool
Kernel size: 2 x 2
Stride = 2
64 x 64 x 64
Dropout
64 x 64 x 64
Convolutional
Kernel size: 3 x 3
Padding = 1
128 x 64 x 64
Batch Normalization
128 x 64 x 64
ReLU
128 x 64 x 64
MaxPool
Kernel size: 2 x 2
Stride = 2
128 x 32 x 32
Dropout
128 x 32 x 32
Convolutional
Kernel size: 3 x 3
Padding = 1
256 x 32 x 32
Batch Normalization
256 x 32 x 32
ReLU
256 x 32 x 32
MaxPool
Kernel size: 2 x 2
Stride = 2
256 x 16 x 16
Dropout
256 x 16 x 16
Global Average Pooling
256 x 1 x 1 → View → 256

El modelo fue optimizado con Adam y regularización combinada L1 y L2. Se utilizó un scheduler de learning rate. El scheduler elegido fue una combinación de Exponencial con Step con los siguientes parámetros:

- Gamma = 0.5
- Decay = 0.95
- Step size = 5

Se optimizaron los siguientes hiperparámetros mediante un sampler TPE con 100 pruebas:

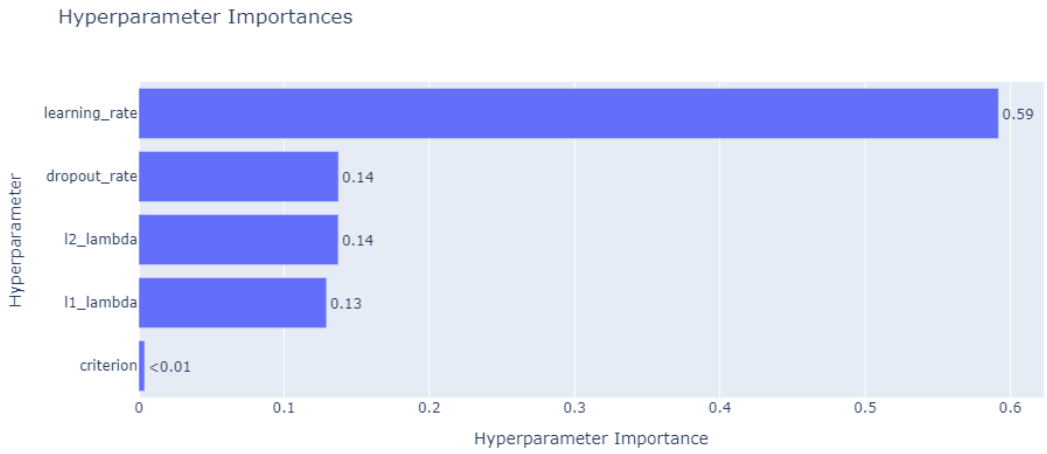
- Dropout rate, con valores entre 0.25 y 0.5
- Learning rate entre 0.001 y 0.1
- El parámetro lambda de regularización L1, con valores entre 10-6 y 0.0001
- El parámetro lambda de regularización L2, con valores entre 0.0001 y 0.01
- La métrica de la pérdida, entre las siguientes opciones:
- Cross Entropy Loss
- Cross Entropy Loss con un tensor de pesos de 0.8, 1 y 0.95 para engagement bajo, neutro y alto, respectivamente.
- Cross Entropy Loss con label smoothing de 0.1.

Los parámetros elegidos para la optimización tenían por objetivo conseguir una convergencia adecuada (learning rate) y evitar el *overfitting*.

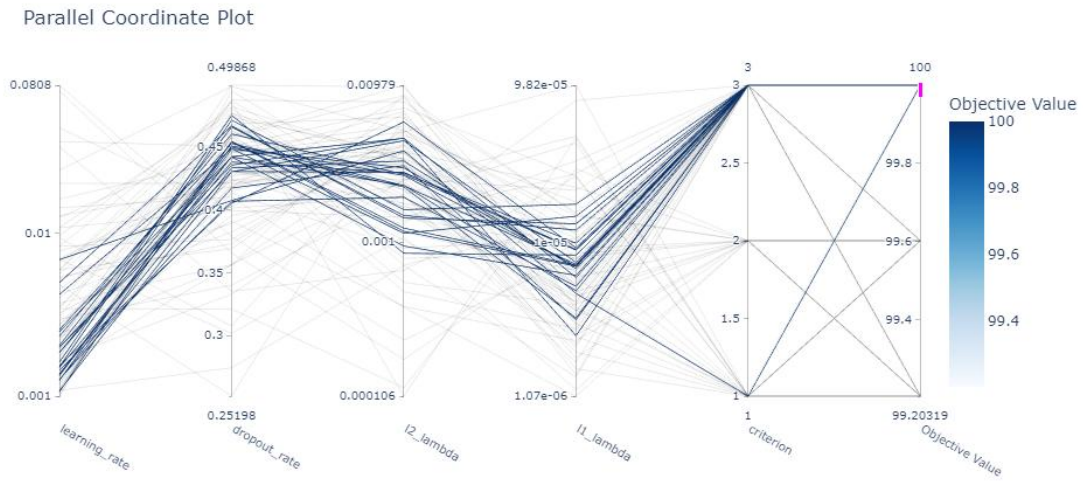
La función objetivo en los estudios realizados era el último valor de precisión.

Se fijaron, por tanto, los parámetros restantes: tamaño del lote (64), número de épocas (10) y dimensión del *embedding* (10 000).

De la evaluación de hiperparámetros, se concluyó que los parámetros más determinantes fueron, por orden, Learning rate (con una importancia relevante en comparación al resto), Dropout rate, lambda L2, lambda L1 y la función de pérdida (la cual, apenas influyó en los resultados).



En el siguiente gráfico se pueden ver los valores óptimos para conseguir la precisión máxima:



2.4. Elección y reproducción del Modelo

Numerosos *trials* consiguieron un objetivo de precisión del 100%. Sin embargo, para evitar el *overfitting* y elegir modelos en buen estado de convergencia, se eligió un trial con pérdidas bajas tanto en entrenamiento como en validación, y con precisiones altas tanto en entrenamiento como en validación:

Número de trial	Pérdida en el entrenamiento	Pérdida en la validación	Precisión en el entrenamiento	Precisión en la validación
#85	0.6513	0.3122	99.8407	100.0000

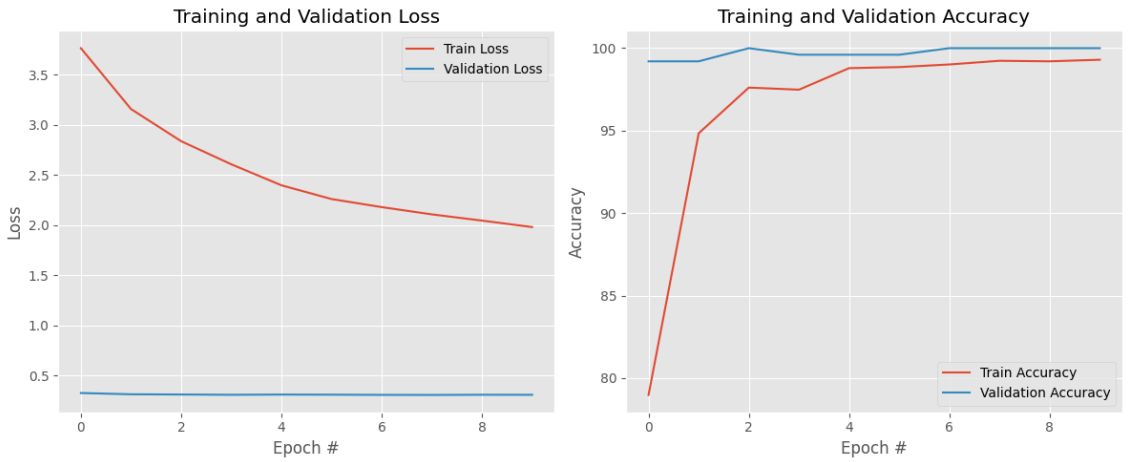
Los hiperparámetros resultantes de la optimización fueron los siguientes:

- Dropout rate: 0.4417003083584299

- Learning rate: 0.0018619371013620589
- Lambda L1: 6.197785458609027e-06
- Lambda L2: 0.00094157395640723
- Función de pérdida: Cross Entropy Loss con label smoothing de 0.1

La semilla utilizada en el entrenamiento con dichos parámetros fue 176.

El modelo se reprodujo y se consiguieron las siguientes curvas de pérdida y precisión:



Se vio que la pérdida del modelo era baja, mientras que la precisión es muy alta. Dado que parecía que el modelo aún tenía alguna muestra que aprender en entrenamiento, se probó a aumentar el número de épocas a 20. Con 20 épocas, este fue el resultado:

El modelo parecía haberse estabilizado con una precisión del 100% en validación y de un 99.27% en entrenamiento.

2.5 Evaluación del Modelo en Test

Tras la evaluación del modelo en test, se obtuvo la siguiente matriz de confusión:

		Real		
		Bajo	Neutro	Alto
Predicho	Bajo	294	0	0
	Neutro	0	0	5
	Alto	0	0	15

La precisión global fue, por tanto, del **98.408%**.

4. Conclusiones

- La precisión general es notablemente alta.
- El modelo clasifica bien los datos de *engagement* bajo.
- El modelo no clasifica del todo bien los datos de *engagement* alto, mostrando cierta incertidumbre en su clasificación entre *engagement* alto o neutro. En los ejemplos de test ha dado 5 casos de falso positivo de *engagement* neutro, o lo que es lo mismo, 5 casos de falso negativo de *engagement* alto, siendo esto un 25% de error de clasificación de los datos de *engagement* alto. Por otra parte, no observamos errores de predicción extremos (predicción de *engagement* bajo con objetivo real de *engagement* alto).
- No podemos concluir nada con certeza sobre cómo realiza la clasificación de datos con *engagement* neutro, ya que el Dataset de test no disponía de estos datos por la imposibilidad de estratificar en la división de los datos. Aunque sepamos que en validación se han clasificado bien, no podemos tomarlo como métrica, ya que evaluar el modelo final según los datos de validación sería sesgar esta evaluación. Esto se debe a que se han utilizado las métricas de validación como objetivo de optimización de hiperparámetros y han sido estas un factor clave en la elección del modelo.

Estos resultados sugieren que el modelo es robusto en cuanto a diferenciación de extremos de *engagement*. La evaluación podría mejorarse con un conjunto nuevo de test que dispusiese de datos con *engagement* neutro.