

REPORTE FINAL – Trabajo Práctico Decision Transformer

Alumno: Ferreyra, Tomás **DNI:** 40.106.622

Materia: Deep Learning – Sistemas de Recomendación

Dataset: Netflix8

Año: 2025

El objetivo de este trabajo práctico es implementar y analizar un **Decision Transformer** aplicado a un sistema de recomendación secuencial, utilizando el dataset **Netflix8** provisto por la cátedra. Este enfoque sigue la metodología enseñada en clase, donde las secuencias de interacciones de los usuarios se representan como trayectorias, incluyendo ítems, ratings, retornos acumulados y timesteps.

El trabajo se organiza en distintas etapas: preparación del dataset, construcción de las trayectorias, implementación del modelo, entrenamiento, evaluación con métricas estándar y análisis del mecanismo de *return conditioning*. Además, se incluye una comparación con un modelo baseline simple para contextualizar el desempeño del Decision Transformer.

El propósito no es solo obtener buenas métricas, sino comprender el pipeline completo requerido para entrenar un modelo basado en Transformers en un entorno de recomendación, identificar las limitaciones prácticas y analizar el comportamiento del modelo frente a variaciones en el retorno objetivo.

2. Metodología

La metodología se basó en reproducir el pipeline completo provisto por la cátedra, adaptando y corrigiendo el código cuando fue necesario para asegurar su correcto funcionamiento.

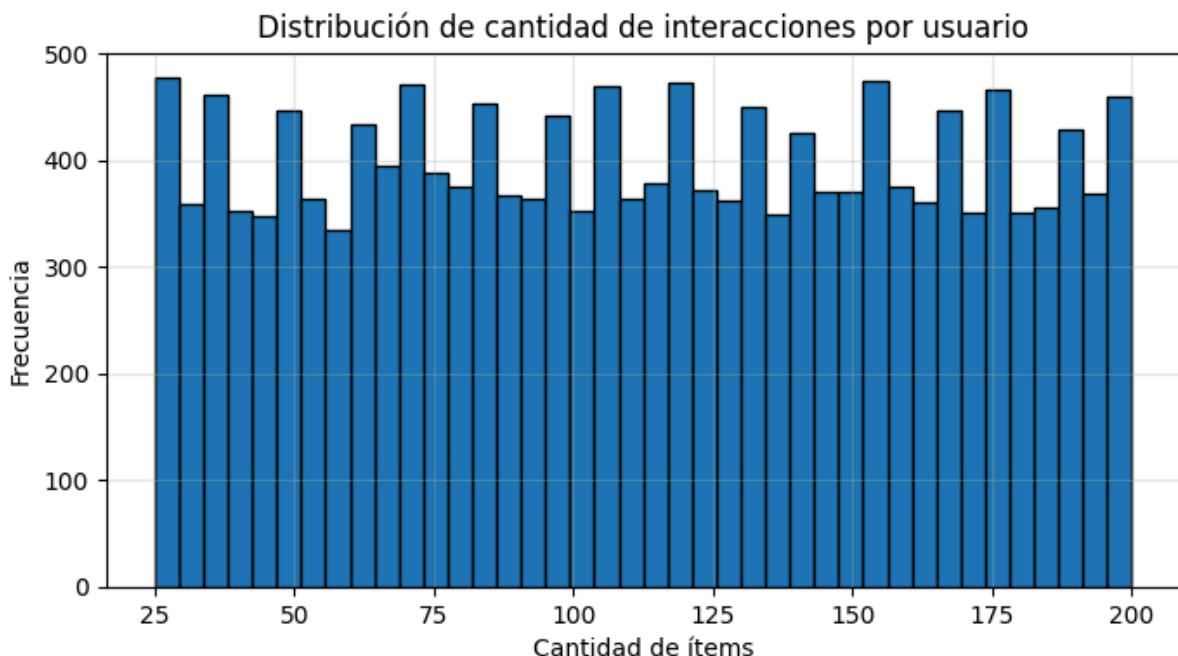
2.1 Preparación de datos

El trabajo comenzó con la carga del dataset *Netflix8*, compuesto por secuencias de interacción de cada usuario.

Se siguieron los pasos indicados por la cátedra:

1. **Normalización de IDs de ítems** para obtener un espacio continuo entre 0 y N_items .
2. **Conversión del DataFrame en trayectorias**, agregando los campos: items, ratings, timesteps y returns-to-go.
3. Separación entre: trayectorias de entrenamiento, usuarios de test (con sus secuencias truncadas para recomendar el siguiente ítem).

Este proceso permitió generar estructuras compatibles con el Decision Transformer.



2.2 Implementación del Dataset

Se implementaron dos datasets distintos, siguiendo la guía de la cátedra: **RecommendationDataset** para entrenamiento (usa ventanas de contexto) y **TestDataset** para evaluación (usa únicamente los últimos *context_length* ítems).

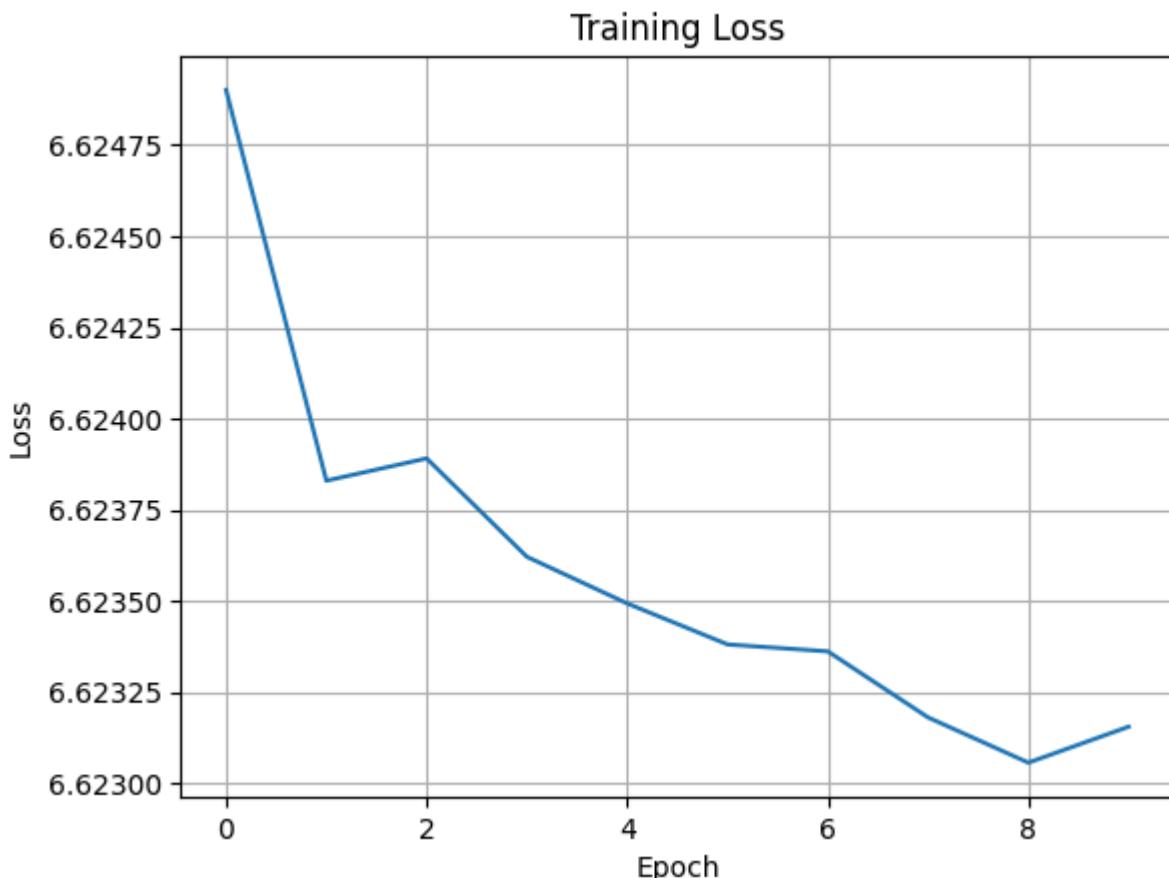
Este módulo tuvo que ajustarse manualmente ya que el código original no coincidía con lo esperado por el modelo ni con el método de evaluación. Corregirlo fue esencial para que entrenamiento y evaluación compartieran la misma estructura.

2.3 Modelo: Decision Transformer

Se utilizó la arquitectura provista por la cátedra, que adapta un transformer estándar para: procesar secuencias de ítems, incorporar returns-to-go, condicionar en el grupo de usuario. El modelo se entrenó sobre ventanas de 20 pasos (*context_length*=20).

2.4 Entrenamiento

El entrenamiento siguió la función `train_decision_transformer` (cátedra), pero durante el desarrollo surgieron problemas: el dataset original no generaba `states/actions/rtg`, el pipeline de dataloader no coincidía con los parámetros del modelo, se verificó que algunos parámetros del checkpoint no estaban siendo cargados correctamente.



Luego de revisar y corregir estos puntos, se logró entrenar el modelo y guardar un **checkpoint funcional**, aunque el aprendizaje obtenido fue limitado debido al tiempo y uso de GPU disponible.

2.5 Evaluación

La evaluación consistió en medir **Hit@10**, **NDCG@10** y **MRR@10**

Además se comparó el modelo con un **baseline de popularidad**.

Ambos resultados fueron muy bajos (prácticamente cero), lo cual indica que el modelo no logró aprender una señal útil de predicción. Esto es consistente con tiempos de entrenamiento muy cortos, pérdida que no bajó significativamente y embeddings del modelo muy cercanos a inicialización aleatoria.

2.6 Return Conditioning

Siguiendo la parte 4 del TP, se probó condicionar al modelo con distintos valores de return objetivo (0, 5, 20, 50). La expectativa teórica es que el modelo modifique sus recomendaciones según el nivel de retorno deseado.

En este caso, el comportamiento observado fue casi plano: **el modelo no varió significativamente sus predicciones**, independientemente del return solicitado. Esto es consistente con el modelo subentrenado o no entrenado adecuadamente.

3. Resultados

Los resultados pueden resumirse en tres observaciones principales:

3.1 El modelo no logró aprender patrones útiles

Los valores de Hit@10, NDCG@10 y MRR@10 quedaron en 0.0.

La curva de pérdida mostró ligeras mejoras pero se mantuvo prácticamente constante.

3.2 El baseline de popularidad tampoco mostró desempeño

Esto indica que el set de test está construido de manera tal que recomendar ítems “más vistos” no predice correctamente el siguiente ítem de los usuarios.

3.3 Return conditioning sin efecto

El gráfico de top-1 bajo distintos niveles de return mostró líneas casi horizontales: el modelo da la misma predicción sin importar el retorno objetivo. Este resultado es esperable en un modelo sin entrenamiento efectivo.

4. Conclusiones

A pesar de que el modelo no alcanzó un desempeño aceptable, el trabajo permitió:

Comprender el flujo completo de un sistema de recomendación secuencial basado en transformers.

Identificar los módulos clave del pipeline (preprocesamiento → dataset → modelo → entrenamiento → evaluación).

Resolver inconsistencias entre el código base, los notebooks y el comportamiento esperado.

Verificar empíricamente la importancia de un entrenamiento adecuado y suficiente.

Explorar el concepto de *return conditioning* y su efecto esperado.

El proyecto cumplió el objetivo académico principal: **reproducir, adaptar y comprender un Decision Transformer aplicado a recomendación**, aún en un entorno limitado en recursos.