AUTOMATIC MUSIC PLAYLIST GENERATION VIA SIMULATION-BASED REINFORCEMENT LEARNING

FEDERICO TOMASI, JOSEPH CAUTERUCCIO, SURYA KANORIA, KAMIL CIOSEK, MATTEO RINALDI, ZHENWEN DAI

INDICE

- Introducción
- Trabajos Relacionados
- Problema
- Solución
- Metodología
- Resultados
- Conclusión

INTRODUCCIÓN

Objetivo: Generar listas de reproducción para mejorar la satisfacción del usuario

Problema: Métodos tradicionales, presentan limitaciones que provocan desajustes entre los objetivos del modelo y las métricas de satisfacción del usuario.

Solución: Un marco basado en aprendizaje por refuerzo que utiliza entornos simulados para optimizar la generación de listas en función de la satisfacción del usuario.

TRABAJOS RELACIONADOS

Música y Sistemas de Recomendación: Desafíos únicos debido a la dificultad de modelar preferencias musicales implícitas.

Filtrado Colaborativo y Modelado Secuencial: Limitaciones para manejar la diversidad de preferencias de los usuarios en diferentes contextos.

Recomendaciones de Música Basadas en AR: Enfoques previos se limitan a la selección de una sola canción, ignorando la secuencialidad de las listas completas.

FORMULACIÓN DEL PROBLEMA

Queremos generar una lista de canciones para el usuario

- Millones de canciones
- Cada usuario tiene una experiencia distinta con la música

Se diseña un modelo de comportamiento del usuario:

- Este estima cómo un usuario responde a cada canción, de tal forma que se intenta maximizar una métrica de satisfacción del usuario.
- Para modelar a un usuario, se asume que este tiene una preferencia por canciones que pertenecen a una cierta categoría (género, tiempo del día, tipo de dispositivo)

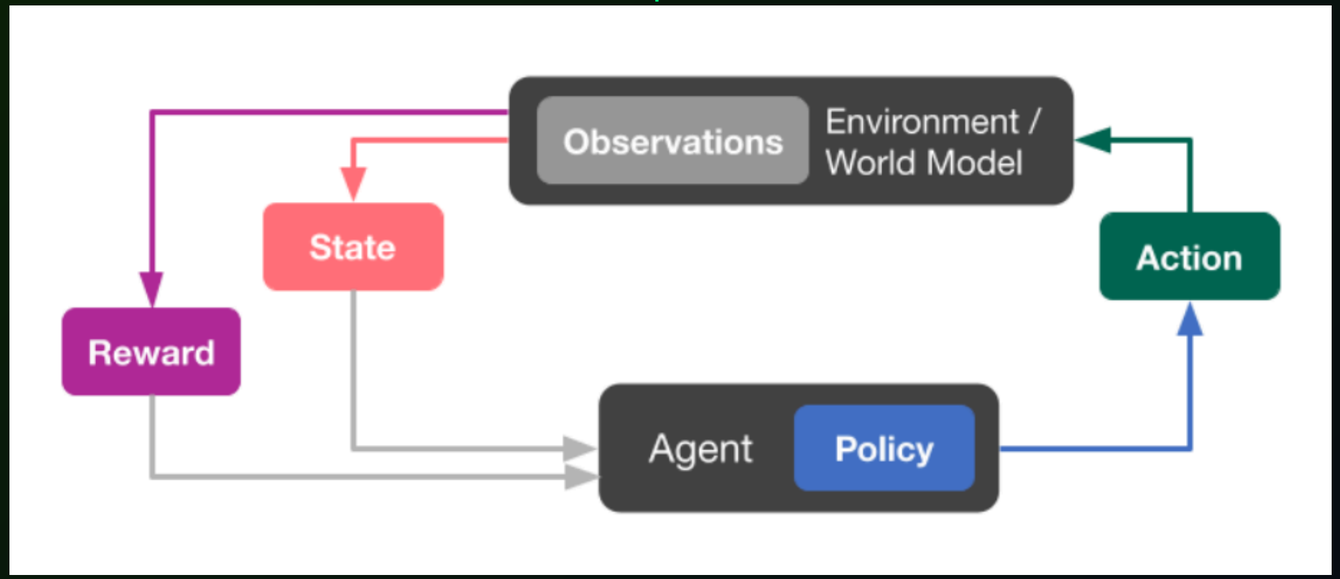
- Para reducir la cantidad de canciones posibles, se considera un subconjunto, que llamamos *candidate pool,* en base a los factores anteriores.

Con esto podemos resolver el problema de recomendación usando un framework de RL, que se explica a continuación

METODOLOGÍA

- El entorno consiste en un *world model* que modela el comportamiento del usuario en respuesta a una acción.
- El entorno modela una función de transición y una función de recompensa.
- El agente debe aprender una política, que mapea estados a acciones.
- El estado inicial se muestrea a partir de una distribución.

Model-based RL framework loop



Para la tarea de recomendación:

Action ⟶ item (canción)

- El entorno consume la acción y, usando el *world model,* determina el siguiente estado y entrega una recompensa condicionada a la acción.
- Esta recompensa es, por ejemplo, si el usuario reproduce la canción o la agrega a sus favoritos.

World model

- Contiene información como el *pool* de items candidatos, el estado del usuario y ciertos *features* de la canción (item) actual.
- Junto con datos históricos, modela la función de transición.

¿Cómo se obtiene la recompensa?

- El *world model* utiliza un *user model,* que toma la acción y algunos *features* y en base a eso hace una predicción tomando en cuenta el estado actual del entorno.
- Este modelo se entrena previamente con datos de sesiones de usuarios.

¿Cómo se elige la acción?

- Se utiliza un Deep Q-Learning (DQN) Agent.
- Este usa una red neuronal para predecir la calidad de la recomendaión (Q) de cada item.
- Cada item posible se asigna a un Q-value, y luego el item con el mayor Q-value es el seleccionado por el agente.
- La recompensa retornada por el entorno ayuda a actualizar la red neuronal.

RESULTADOS

- Evaluar las políticas de recomendación de playlists en dos entornos distintos: **simulaciones con datos públicos** y **pruebas en línea con usuarios reales.**

- Las politicas evaluadas son:
  - AH-DQN (Agente basado en aprendizaje por refuerzo).
  - CWM-GMPC (Modelo no secuencial con política greedy).
  - Política Aleatoria

RESULTADOS: EVALUACIÓN CON DATOS PÚBLICOS

- Validar el comportamiento del modelo en un entorno controlado, utilizando datos de escucha pública de Spotify (160 millones de sesiones de escucha)
- Calculado como la suma de recompensas obtenidas en cada episodio, basado en la satisfacción del usuario (tasa de canciones completadas o saltadas).

RESULTADOS: EVALUACIÓN CON DATOS PÚBLICOS

Table 1: Average return for random and agent policy on offline evaluation of the public streaming dataset.

Policy	Avg. Return	σ	CI (95%)
Action Head Policy	1.94	1.27	(0.32 | 4.18)
CWM-GMPC	2.46	1.10	(0.83 | 4.36)
Random	0.98	0.76	(0.0 | 2.54)

RESULTADOS: EVALUACIÓN CON DATOS PÚBLICOS

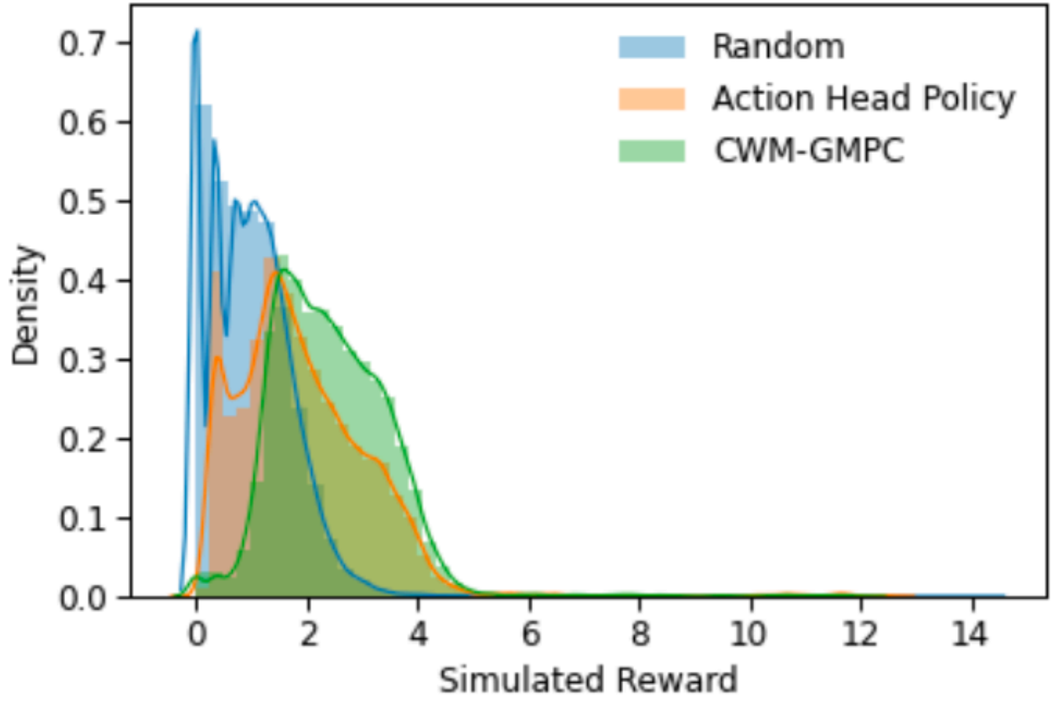


Figure 3: Reward distributions for the compared policies on the public streaming dataset as estimated by the SWM.

RESULTADOS: EXPERIMENTOS EN LÍNEA

- Evaluar cómo las políticas entrenadas en simulación se desempeñan en pruebas reales con usuarios de Spotify
- Se evaluo usando las metricas de **tasa de finalización** y **tasa de saltos**
- Se ocupo el **A/B Testing** en donde se dividieron a los usuarios en varios grupos de prueba, asignando las playlists generadas por las políticas AH-DQN, CWM-GMPC, Similitud Coseno y Política Aleatoria.

RESULTADOS: EXPERIMENTOS EN LÍNEA

Table 2: Relative percent difference between world model (CWM) policy and control on online evaluation.

Metric (Per-Session Average):	Relative % Difference:
Completion-Count	-2.9 (p: 0.88, CI: -39.38 | 33.59)
Total MSP	-8.59 (p: 0.64, CI: -44.55 | 27.36)
MSP-Per-Item	-13.97 (p: 0.05, CI: -27.89 | -0.06)
Skip-Rate	-5.49 (p: 0.42, CI: -18.77 | 7.79)
Completion-Rate	9.98 (p: 0.23, CI: -6.52 | 26.49)
Session Length (interactions)	13.05 (p: 0.31, CI: -12.02 | 38.11)

RESULTADOS: EXPERIMENTOS EN LÍNEA

Table 3: Relative percent difference between agent policy and control on online evaluation.

Metric (Per-Session Average):	Relative % Difference:
Completion-Count	10.17 (p: 0.59, CI: -26.79 | 47.13)
Total MSP	6.43 (p: 0.73, CI: -30.67 | 43.53)
MSP-Per-Item	-5.62 (p: 0.46, CI: -20.69 | 9.44)
Skip-Rate	-7.8 (p: 0.33, CI: -23.52 | 7.92)
Completion-Rate	5.39 (p: 0.6, CI: -14.8 | 25.58)
Session Length (interactions)	8.87 (p: 0.53, CI: -19.02 | 36.75)

CONCLUSION

Éxito del Enfoque Basado en Simulación:

- El entrenamiento de agentes en entornos simulados demostró ser eficaz para generar recomendaciones precisas.
- AH-DQN y CWM-GMPC superaron al control y políticas aleatorias en métricas clave, mejorando la satisfacción del usuario (más canciones completadas y menos saltos)

CONCLUSION

Ventajas del Aprendizaje por Refuerzo (AH-DQN):

- AH-DQN optimizó la generación de playlists maximizando la satisfacción del usuario.
- Los experimentos en línea validaron que AH-DQN reduce significativamente los saltos y aumenta la tasa de finalización de canciones.

Rendimiento Consistente de CWM-GMPC:

- CWM-GMPC mostró un rendimiento sólido y consistente, lo que lo convierte en una alternativa competitiva en la personalización de playlists.

CONCLUSION

Impacto en la Experiencia del Usuario:

- Ambas políticas mejoraron la experiencia del usuario al generar playlists más relevantes y satisfactorias, validando el enfoque basado en simulación.

Implicaciones Futuras:

- El uso de simulaciones para entrenar modelos de recomendación tiene un gran potencial en otros dominios, permitiendo personalizar contenidos sin comprometer la experiencia de los usuarios.