UNIVERSIDAD ANDRÉS BELLO
FACULTAD DE INGENIERÍA

INGENIERÍA CIVÍL INFORMÁTICA

RecSysUnab: Sistema de recomendación de ejercicios de programación basado en un modelo de dos torres

EDGAR ALEJANDRO RAMOS VIVENES
Profesor guía: Pablo Schwarzenberg

SANTIAGO - CHILE

Diciembre - 2024

Contents

List of Figures

List of Tables

RecSysUnab: Sistema de recomendación de ejercicios de programación basado en un modelo de dos torres

EDGAR ALEJANDRO RAMOS VIVENES, Universidad Andrés Bello, Chile

Objetivo Desarrollar e implementar un sistema de recomendación de ejercicios de programación para estudiantes de la Universidad Andrés Bello, utilizando el método de "dos torres“ con el propósito ofrecer una orientación personalizada que se adapte a la diversidad de niveles de habilidad de los estudiantes y a sus características individuales, en relación con los ejercicios establecidos en el curso de "Introducción a la Programación“.

Participantes Edgar Alejandro Ramos Vienes

Métodos de Estudio El método de estudio de este proyecto se dividirá en siete secciones principales: introducción al tema, metodología a implementar, criterios a seguir, experimentos realizados, resultados obtenidos, conclusiones propuestas y una expresión de agradecimiento por la realización de este proyecto. En la sección de metodología, se empleó la metodología CRISP-DM para el desarrollo del proyecto, lo que permitió una mayor flexibilidad en el análisis de datos, la preparación de los mismos, el desarrollo y evaluación de modelos, así como la implementación del mejor modelo obtenido hasta el momento para proporcionar una solución efectiva al problema planteado.

Additional Key Words and Phrases: Sistema de recomendación, Recomendadores, Inteligencia Artificial, Programación, Universidad Andrés Bello, Aprendizaje Personalizado, Aprendizaje Profundo, Educación, E-Learning, Sistemas de Recomendación de Dos Torres, Aprendizaje Supervisado

ACM Reference Format:
Edgar Alejandro Ramos Vivenes. 2024. RecSysUnab: Sistema de recomendación de ejercicios de programación basado en un modelo de dos torres. *ACM Trans. Comput. Educ.* 1, 1, Article 1 (December 2024), 22 pages. https://doi.org/xx.xxxx/xxxxxxx

Author's address: Edgar Alejandro Ramos Vivenes, e.ramosvivenes@uandresbello.edu, Universidad Andrés Bello, Chile.

1946-6226/2024/12-ART1 $00.00
https://doi.org/xx.xxxx/xxxxxxx

1 INTRODUCCIÓN

En la era actual de la información, caracterizada por la masiva proliferación de sitios web especializados que ofrecen una diversidad abrumadora de productos y servicios, los consumidores enfrentan un desafío creciente: navegar entre innumerables opciones para tomar decisiones informadas. Este exceso de información, conocido como "sobrecarga informativa", no solo complica la toma de decisiones, sino que también genera insatisfacción y pérdida de eficiencia en la búsqueda de soluciones adecuadas.

En respuesta a este desafío, los sistemas de recomendación han emergido como herramientas clave para filtrar, personalizar y priorizar la información según las necesidades específicas de los usuarios. Estos sistemas tienen la capacidad de analizar datos sobre las preferencias, comportamientos e intereses de los usuarios, y a partir de ellos, generar recomendaciones personalizadas que optimicen la experiencia del usuario, ya sea en el ámbito del comercio, el entretenimiento o la educación.

En los últimos años, la investigación en sistemas de recomendación ha avanzado significativamente, trascendiendo el simple emparejamiento de usuarios con productos. Ahora, el énfasis se encuentra en la personalización avanzada y el diseño de modelos adaptativos que integren enfoques interdisciplinarios de áreas como inteligencia artificial, minería de datos, experiencia de usuario e interacción persona-computadora. Estos avances han permitido el desarrollo de sistemas que no solo comprenden las preferencias explícitas de los usuarios, sino que también infieren patrones complejos para anticipar sus necesidades.

En este contexto, este proyecto se centra en el diseño e implementación de un sistema de recomendación adaptativo para estudiantes de la Universidad Andrés Bello, específicamente orientado a mejorar su aprendizaje en el área de programación. Este sistema busca ofrecer recomendaciones personalizadas de ejercicios, considerando las características individuales y el nivel de habilidad de cada estudiante, con el objetivo de apoyar su desarrollo académico y aumentar su efectividad en la resolución de problemas.

A lo largo de este trabajo, se explorarán los fundamentos teóricos y prácticos de los sistemas de recomendación, se justificará el uso de un enfoque adaptativo basado en modelos avanzados como los sistemas de dos torres, y se analizarán los resultados obtenidos. Finalmente, se presentarán conclusiones que destacan el impacto y las implicaciones de esta investigación en el ámbito académico y tecnológico de la Universidad Andrés Bello.

1.1 Revisión de la bibliografía

Los sistemas de recomendación son herramientas esenciales diseñadas para filtrar información masiva y proporcionar contenido relevante y personalizado a los usuarios. Su principal propósito es abordar la sobrecarga de información que dificulta la identificación de recursos adecuados, facilitando la toma de decisiones en dominios como el comercio electrónico, el entretenimiento y, especialmente, la educación [11] [16] [1].

En el ámbito educativo, los sistemas de recomendación buscan mejorar la experiencia de aprendizaje al ofrecer recomendaciones adaptadas a las necesidades específicas de los estudiantes. Esto incluye cursos, materiales y actividades que se alineen con sus estilos de aprendizaje y objetivos académicos [11] [5] [3]. Además, los docentes pueden beneficiarse al utilizar estas herramientas para planificar estrategias pedagógicas más efectivas, ajustadas a las necesidades de sus alumnos [3] [13]. Sin embargo, persiste una brecha en la investigación, ya que muchas evaluaciones de sistemas de recomendación se centran en métricas como la precisión, ignorando su impacto pedagógico [3] [6] .

Los métodos utilizados en sistemas de recomendación incluyen técnicas tradicionales como el filtrado colaborativo, el filtrado basado en contenido y modelos híbridos, estos últimos son los más destacados en contextos educativos recientes debido a su capacidad para combinar fortalezas de múltiples enfoques [10] [3]. Con la incorporación de aprendizaje profundo, los sistemas de recomendación han avanzado significativamente, permitiendo captar relaciones complejas entre usuarios y recursos educativos, lo que mejora la calidad y personalización de las recomendaciones [16] [1]. Además, los enfoques basados en aprendizaje por refuerzo, aunque prometedores, enfrentan retos como la escalabilidad y los altos costos computacionales [1].

En términos de impacto, los sistemas de recomendación optimizan la eficiencia del aprendizaje al reducir el tiempo y esfuerzo necesarios para encontrar recursos relevantes, personalizando las recomendaciones según las habilidades, intereses y objetivos de los estudiantes. Esto no solo mejora el rendimiento académico, sino que también fomenta la motivación al proporcionar contenido alineado con las preferencias individuales [5] [3] [6] .

A pesar de estos beneficios, los sistemas de recomendación enfrentan problemas tecnológicos como el inicio en frío, la escasez de datos y la escalabilidad. Además, en el ámbito educativo, es crucial incorporar factores éticos como la equidad y la diversidad para evitar sesgos y garantizar que las recomendaciones sean inclusivas y alineadas con los principios pedagógicos [1] [3] [18]. Las tendencias actuales apuntan hacia sistemas más adaptativos, con un enfoque en la inteligencia artificial para superar limitaciones tradicionales y evaluar su impacto en el aprendizaje de manera más holística [11] [3] [18].

Los sistemas de recomendación basados en la arquitectura de dos torres (two-tower) son ampliamente reconocidos por su eficacia en entornos de gran escala debido a su capacidad para separar eficientemente la representación de usuarios y elementos recomendados en espacios vectoriales. Esta separación permite precomputar las representaciones de los elementos, optimizando así el tiempo de inferencia mediante algoritmos de búsqueda de vecinos más cercanos (ANN, por sus siglas en inglés), lo que es esencial para escenarios donde los tiempos de respuesta son crítico [2] [15] [12] [17].

En el contexto educativo, los sistemas de recomendación de dos torres pueden aportar ventajas significativas al facilitar la personalización de contenido para estudiantes. Por ejemplo, estos modelos pueden integrar características de cursos y comportamientos de los usuarios para ofrecer recomendaciones adaptadas a sus necesidades

específicas, superando problemas comunes como la sobrecarga de información y la escasez de datos [7] [4].

Además, al abordar problemas como el "inicio en frío", donde hay poca interacción inicial entre usuarios y elementos, los sistemas de dos torres utilizan características adicionales, como descripciones de cursos, intereses profesionales o datos demográficos, para mejorar la precisión de las recomendaciones incluso con datos limitados [15] [14].

En términos de impacto en la educación, estas capacidades permiten a los sistemas adaptarse dinámicamente a los estilos de aprendizaje individuales, mejorando la retención y motivación de los estudiantes. Además, al reducir la carga cognitiva asociada con la selección de recursos, los estudiantes pueden enfocarse en objetivos de aprendizaje más claros y efectivos [7] [12].

1.2 Trabajos Relacionados

En 2021, investigadores en Corea se enfrentaron al problema de la sobrecarga de información en plataformas de educación en línea, donde los estudiantes tenían dificultades para encontrar cursos relevantes entre miles de opciones. Este desafío se intensificaba por el problema del inicio en frío, donde nuevos usuarios carecían de un historial suficiente para generar recomendaciones personalizadas.

Para resolver esta situación, desarrollaron el sistema DECOR, basado en una arquitectura de dos torres. Una torre modelaba las características de los estudiantes, como sus intereses profesionales y objetivos académicos, mientras que la otra representaba atributos de los cursos, incluyendo descripciones y niveles de popularidad. Estas representaciones se integraban mediante módulos avanzados que capturaban interacciones complejas entre estudiantes y cursos, complementados con datos contextuales para mitigar el inicio en frío.

Las pruebas realizadas en datasets educativos de universidades coreanas mostraron que DECOR superó a métodos tradicionales como el filtrado colaborativo en precisión y tasas de clics. En particular, logró una mejora del 18% en la tasa de clics y un aumento del 12% en la precisión de las recomendaciones. Los hallazgos demostraron que DECOR no solo personalizó eficazmente la experiencia de aprendizaje, sino que también permitió a los estudiantes encontrar cursos alineados con sus necesidades y metas individuales [7].

En 2020, un equipo de investigadores se propuso resolver un problema recurrente en sistemas de recomendación a gran escala: cómo identificar candidatos relevantes entre millones de opciones, especialmente en escenarios de inicio en frío. Este desafío era crítico en plataformas como Google Play, donde los usuarios y elementos nuevos carecían de datos suficientes para generar predicciones confiables.

Para abordar esta dificultad, se propuso un modelo de dos torres combinado con una técnica innovadora de entrenamiento llamada Mixed Negative Sampling (MNS). Este enfoque optimizó la selección de datos negativos durante el entrenamiento, permitiendo que el modelo diferenciara entre elementos irrelevantes (negativos fáciles) y aquellos más complejos (negativos duros). Esta estrategia mejoró la calidad de las representaciones de usuarios y elementos, incluso en condiciones de datos limitados.

Los experimentos realizados en el sistema de recomendaciones de Google Play mostraron mejoras significativas: un incremento del 15% en la precisión de las recomendaciones y un aumento del 20% en la tasa de instalaciones de alta calidad. Los resultados confirmaron que MNS es altamente efectivo para resolver el inicio en frío y que su implementación en plataformas educativas podría mejorar la personalización al ofrecer contenido relevante, incluso para cursos o estudiantes nuevos [15].

En 2022, en el contexto de sistemas de recomendación industriales con grandes volúmenes de datos, se identificó un problema clave en los modelos tradicionales de dos torres: la falta de interacciones tempranas entre usuarios y elementos. Esto afectaba su capacidad para capturar relaciones complejas, especialmente en dominios como la educación, donde es esencial alinear las características de los estudiantes con los atributos de los recursos educativos desde el inicio.

Para abordar esta limitación, los investigadores desarrollaron IntTower, un modelo de dos torres que introduce un módulo de interacción temprana (FE-Block) para capturar señales detalladas desde las primeras etapas del proceso. Este enfoque se complementó con técnicas de regularización diseñadas para evitar el sobreajuste, mejorando la capacidad de modelado y garantizando una mayor precisión en las recomendaciones.

Los resultados experimentales en datasets industriales fueron notables: IntTower mostró un incremento del 22% en la precisión de las recomendaciones en comparación con modelos estándar y un ahorro del 15% en tiempo de inferencia gracias a su optimización computacional. Estas mejoras posicionaron a IntTower como una herramienta ideal para aplicaciones educativas, permitiendo recomendaciones en tiempo real y personalizadas que transforman la experiencia de aprendizaje al adaptarse dinámicamente a las necesidades e intereses de los estudiantes [8].

1.3 Problema

En la Universidad Andrés Bello se imparte la asignatura "Introducción a la Programación" para estudiantes de primer año de carreras relacionadas con programación, informática y computación. Como parte de la asignatura, a cada estudiante se le otorga acceso a una plataforma web de ejercicios de programación. Esta herramienta tiene como objetivo principal permitir que los estudiantes practiquen ejercicios, ya sea de manera voluntaria o como parte de los requisitos obligatorios de la asignatura, para prepararse adecuadamente para las evaluaciones del semestre. Además, busca que los estudiantes adquieran los conocimientos necesarios para superar los exámenes y, en última instancia, aprobar la asignatura.

Sin embargo, se han identificado varios problemas que podrían estar afectando el proceso de estudio y dificultando que algunos estudiantes logren los objetivos esperados en la asignatura. Entre los factores más relevantes se encuentran:

- La amplia variedad de ejercicios disponibles en la plataforma dificulta que los estudiantes identifiquen cuáles son los más adecuados para sus necesidades y objetivos de aprendizaje específicos.

- Los estudiantes carecen de orientación personalizada que les permita avanzar de manera efectiva en el aprendizaje de la programación.
- La heterogeneidad en los niveles de habilidad de los estudiantes, que varían desde principiantes absolutos hasta estudiantes más avanzados, genera dificultades. La falta de adaptación de los ejercicios y recursos a estos diferentes niveles puede conducir a insatisfacción, desmotivación y un progreso académico limitado.

1.4 Hipótesis

Implementar un sistema de recomendación basado en la arquitectura de dos torres permitiría ofrecer a los estudiantes de "Introducción a la Programación" recomendaciones personalizadas de ejercicios, adaptadas a sus características individuales y nivel de habilidad. Al seguir las recomendaciones personalizadas, los estudiantes podrían mejorar significativamente sus conocimientos en programación, optimizando su ritmo de aprendizaje facilitando la resolución de los ejercicios propuestos.

1.5 Objetivo General

Desarrollar e implementar un sistema de recomendación de ejercicios de programación para estudiantes de la Universidad Andrés Bello, utilizando el método de "dos torres“ con el propósito ofrecer una orientación personalizada que se adapte a la diversidad de niveles de habilidad de los estudiantes y a sus características individuales, en relación con los ejercicios establecidos en el curso de "Introducción a la Programación“.

1.6 Objetivos Específicos

- Diseñar e implementar modelos que utilicen datos como el nivel de habilidad, el historial de ejercicios resueltos y otras características individuales de los estudiantes para personalizar las sugerencias de ejercicios de programación.
- Construir y evaluar modelos de recomendación basados en la arquitectura de "dos torres", identificando el enfoque más efectivo para mejorar el aprendizaje y desempeño de los estudiantes en programación.
- Garantizar que el sistema de recomendación proporcione a todos los estudiantes una variedad de ejercicios adaptados a sus características individuales.

2 METODOLOGÍA

La metodología utilizada para desarrollar este proyecto fue CRISP-DM, que son las siglas de Cross-Industry Standard Process for Data Mining, una metodología orientada específicamente a proyectos de minería de datos. Esta metodología es flexible y puede personalizarse fácilmente según las necesidades del proyecto. Por tanto, este trabajo se estructuró en seis fases fundamentales, descritas a continuación:

(1) **Comprensión del problema:** Se definió claramente el objetivo del proyecto y los problemas a resolver, asegurando una alineación completa con las necesidades de un grupo específico de estudiantes.

(2) **Comprensión de los datos:** Se recopilaron, exploraron y analizaron los datos disponibles para evaluar su calidad, estructura y relevancia en relación con el problema planteado.
(3) **Preparación de los datos:** Los datos fueron limpiados, transformados y estructurados en un formato adecuado para el modelado. Se seleccionaron las variables más relevantes y se manejaron valores faltantes o inconsistencias.
(4) **Modelado:** Se seleccionaron y aplicaron técnicas de modelado adecuadas para construir modelos que resolvieran el problema. Se ajustaron los parámetros de los modelos para optimizar los resultados.
(5) **Evaluación:** Se analizó el desempeño de los modelos con respecto a los objetivos planteados, verificando su precisión, relevancia y aplicabilidad antes de proceder con la implementación.
(6) **Implementación:** Se guardó el mejor modelo desarrollado para su uso futuro en algún entorno de producción.

3 CRITERIOS

3.1 Inclusión y Exclusión

Este proyecto estuvo dirigido a estudiantes universitarios de la Universidad Andrés Bello. Por lo tanto, la población objetivo se centró en aquellos estudiantes cuya malla académica incluyera la asignatura "Introducción a la Programación", independientemente de si aprobaron o no dicha asignatura durante el período en el que se recopilaron los datos.

Por otro lado, se excluyeron del estudio a los estudiantes que cumplían con alguna de las siguientes condiciones:

- Estar cursando grados superiores al nivel de pregrado.
- Pertenecer a una carrera que no estuviera relacionada con la asignatura "Introducción a la Programación" o con el ámbito de la programación en general.
- No haber tenido ninguna interacción durante el semestre, ya sea en la realización de ejercicios o en las pruebas asociadas a la asignatura.

3.2 Características de los participantes

En los datos recopilados, se excluyó variables como la edad, el sexo, el RUT y el nombre del estudiante. Por lo tanto, las características se centraron exclusivamente en los resultados obtenidos por cada estudiante durante un semestre natural cursando la asignatura. Las características del estudiante se pueden dividir en dos segmentos principales: la prueba diagnóstica y el registro de interacciones durante el semestre.

La **prueba diagnóstica** fue una evaluación de rendimiento diseñada para medir los conocimientos previos en programación de los estudiantes al inicio de las clases. Consistió en una serie de ejercicios prácticos que cubrían 6 temas distintos. El estudiante podía intentar resolver los ejercicios y obtener una serie de puntajes según el nivel de conocimientos demostrado. Esta prueba tuvo un enfoque práctico y no fue utilizada como instrumento de evaluación formal del semestre; su propósito fue investigativo más que evaluativo.

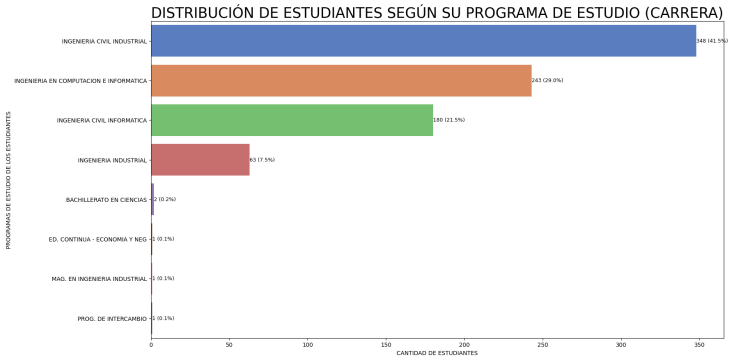

Fig. 1. Evidencia de la participación de estudiantes con estudios no relacionados o grados superiores a pregrado. Conjunto de datos: `data_a_2021.csv`.

Por otro lado, se incluyeron los **resultados obtenidos durante el semestre**, que reflejan el desempeño de los estudiantes en pruebas, controles y tareas variadas. Este conjunto de características de los estudiantes recopila sus puntajes totales y clasifica a los estudiantes en dos categorías según su estado final: aprobado (A) o reprobado (R).

3.3 Procedimientos de muestreo

Se optó por un método de selección simple para incluir a estudiantes que hayan cursado la asignatura "Introducción a la Programación" y que cumplieran con las condiciones previamente especificadas. Además, se estableció una colaboración con el jefe de carrera de Ingeniería Civil Informática de la Universidad Andrés Bello para la obtención de los datos, otorgando permiso para el uso de los registros de los estudiantes, garantizando siempre su privacidad y evitando la divulgación de información personal.

No se realizaron pagos a los participantes en el marco de este proyecto, incluyendo su implementación, desarrollo y prueba del sistema de recomendación. Todos los datos fueron entregados previamente saneados y cumplieron con las normativas éticas y legales proporcionadas por los directivos, específicamente por el jefe de carrera de Ingeniería Civil Informática del campus Antonio Varas de la Universidad Andrés Bello.

3.4 Tamaño de Muestra, Potencia y Precisión

Dentro de los conjuntos de datos, se contó con un total de 1306 registros de estudiantes participantes que cursaron la asignatura "Introducción a la Programación" durante el período establecido entre 2020 y 2021.

Posteriormente, tras aplicar las cláusulas de exclusión y realizar el proceso de limpieza de datos, quedaron disponibles 1092 registros de estudiantes para el desarrollo de este proyecto. De estos, 766 estudiantes aprobaron la asignatura, lo que representa aproximadamente el 70% del total de datos disponibles.

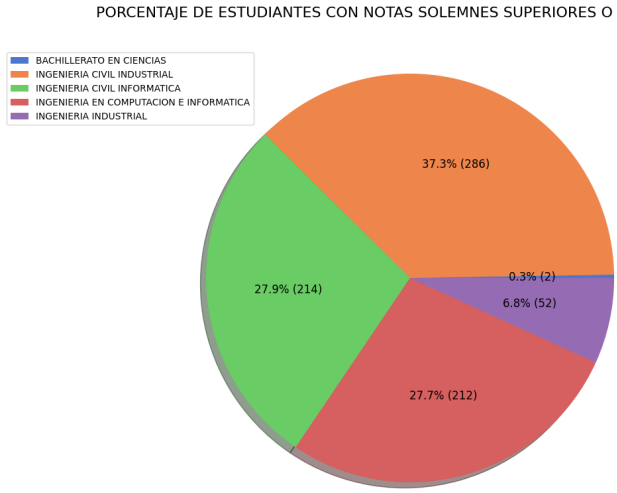

Fig. 2. Distribución de los datos de estudiantes aprobados para entrenamiento, validación y testeo de los modelos.

3.5 Recopilación de datos

La recopilación de datos se realizó a partir de los registros generados por estudiantes que utilizaron una plataforma web diseñada para la práctica y el aprendizaje mediante ejercicios planteados. Esta plataforma fue utilizada como una herramienta de apoyo para que los estudiantes adquirieran los conocimientos necesarios y enfrentaran las pruebas solemnes con mayor preparación.

La mayoría de los registros provienen del campus Antonio Varas de la Universidad Andrés Bello, con un contexto histórico que abarca principalmente el período comprendido entre los años 2020 y 2021. Gracias a ello, fue posible obtener tanto datos históricos como notas de los estudiantes, así como información más específica relacionada con los ejercicios, tales como inputs enviados, cantidad de envíos, resets, ejercicios realizados, entre otros.

Todos los datos recopilados fueron entregados y autorizados por el jefe de carrera de Ingeniería Civil Informática del campus Antonio Varas de la Universidad Andrés Bello.

3.6 Diagnóstico de datos

Según los criterios previamente descritos, se consideraron los criterios de exclusión para ciertos grupos de estudiantes que no aportaban información significativa para el desarrollo del sistema de recomendación. Estos incluían estudiantes que no interactuaron con la plataforma de ejercicios, que no aprobaron la asignatura o que no

pertenecían a un ramo relacionado con "Introducción a la Programación". También se excluyeron estudiantes que estaban cursando estudios superiores al nivel de pregrado.

En cuanto a los procedimientos, se empleó la fase tres de la metodología CRISP-DM, utilizando herramientas y técnicas de ciencia de datos para la limpieza y análisis de los datos recolectados.

Se identificaron datos faltantes en un grupo de estudiantes, principalmente en las notas correspondientes a las solemnes 2, 3 y 4, donde no se registraron datos. Sin embargo, estas variables se usaron únicamente como filtros y no tuvieron una relación directa con los resultados obtenidos en los experimentos.

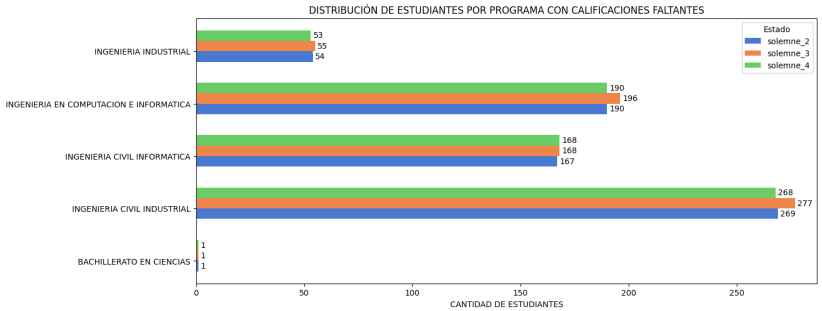

Fig. 3. Distribución de estudiantes cuyas notas de las solemnes 2, 3 y 4 no estaban registradas.

Por otro lado, entre los datos recopilados, se identificaron dos variables categóricas principales: el programa del estudiante y su estado de aprobación. El estado de aprobación solo fue relevante para un grupo específico de estudiantes y se utilizó principalmente en el análisis exploratorio. Sin embargo, la variable del programa de estudio fue esencial para los experimentos. Para procesar esta variable categórica, se utilizó una técnica de codificación que permitió transformar sus valores categóricos en valores numéricos.

4 EXPERIMENTOS

4.1 Enfoque de modelado

Para este proyecto se utilizó la arquitectura de dos torres para abordar la falta de recomendaciones personalizadas en la asignatura "Introducción a la Programación". Esta arquitectura, ampliamente usada en sistemas de recomendación a gran escala, representa separadamente a los usuarios (estudiantes) y los elementos (ejercicios) en espacios vectoriales, permitiendo calcular su similitud.

Cada torre emplea redes neuronales para generar embeddings que capturan las características de usuarios y elementos. Estos embeddings, al combinarse, identifican los elementos más relevantes para cada usuario. Además, la precomputación de los embeddings de los elementos reduce los tiempos de inferencia, lo que es clave para plataformas educativas que requieren recomendaciones rápidas y precisas.

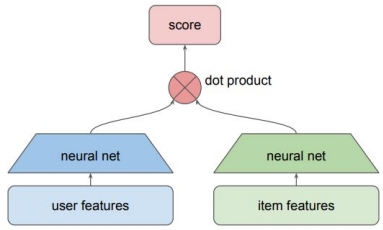

Fig. 4. Arquitectura del modelo de dos torres, que utiliza una torre para modelar usuarios y otra para modelar artículos. El modelo genera un puntaje mediante el cálculo del producto punto entre los embeddings de ambas torres. Fuente: [9].

4.2 Arquitecturas de las torres

Una vez establecida la arquitectura global del sistema, es necesario describir la arquitectura interna que tendrá cada torre, incluyendo los modelos a implementar. Las torres deben contar con arquitecturas similares para mantener la consistencia y el balance en la representación de usuarios y elementos recomendados.

La simetría en las torres ofrece varias ventajas. En primer lugar, permite generar embeddings comparables en un espacio vectorial compartido, lo que facilita el cálculo de similitud entre estudiantes y ejercicios. Además, asegura una convergencia más estable durante el entrenamiento, al evitar que una torre domine el proceso de aprendizaje.

En este proyecto se trabajó de manera modular, separando tanto el modelo como las torres, lo que permitió realizar cambios y comparar resultados de forma más eficiente. En este caso, se construyeron tres versiones de las torres, tanto para los estudiantes como para los ejercicios a recomendar.

Table 1. Arquitecturas de las torres por versiones

Versión	Arquitectura por Torre	Tamaño de Embedding	Dropout
V1	1 Capa Lineal [e, 128] 1 Función de activación (ReLU) 1 Capa Dropout 1 Capa Lineal [128, e]	64	0.5
V2	1 Capa Lineal [e, 256] 1 Función de activación (ReLU) 1 Capa Lineal [256, 128] 1 Función de activación (ReLU) 1 Capa Dropout 1 Capa Lineal [128, e]	64	0.5
V3	1 Capa Lineal [e, 256] 1 Función de activación (LeakyReLU) 1 Capa Dropout 1 Capa Lineal [256, 128] 1 Función de activación (LeakyReLU) 1 Capa Dropout 1 Capa Lineal [128, e]	64	0.5

Otro beneficio importante es que simplifica la implementación del modelo, ya que las arquitecturas simétricas son más fáciles de replicar y depurar. Asimismo, evita desbalances en la calidad de las representaciones, garantizando que ambas torres posean la misma capacidad de modelado.

4.3 Modelos de prueba

En un nivel superior a las torres, se encuentra el proceso de cálculo que analiza ambos embeddings en un espacio vectorial compartido. En este proyecto, se implementaron cinco modelos con diferentes técnicas para calcular un puntaje, el cual se transforma mediante una función sigmoid para producir valores en el rango [0, 1]. En este rango, un valor cercano a 1 indica que el ejercicio debe recomendarse, mientras que un valor cercano a 0 sugiere lo contrario.

Table 2. Descripción de los modelos de prueba

Modelo	Técnica para calcular el score	Función de salida
TwoTowerModel	Producto punto entre embeddings de usuario e ítem	sigmoid
TwoTowerModelV1	Similitud coseno entre embeddings de usuario e ítem	sigmoid
TwoTowerModelV2	Concatenación de embeddings y paso por capas lineales	sigmoid
IntTower	Concatenación de embeddings y paso por capas lineales	sigmoid
SparseTwoTower	Interacción cruzada disperza entre embeddings y capa lineal	sigmoid

A partir de la elección del modelo, se pueden configurar las torres que se utilizarán dentro de su arquitectura, permitiendo adaptar las características y enfoques según las necesidades del problema. Este diseño modular proporciona flexibilidad, facilitando la experimentación y optimización de los componentes individuales para mejorar el rendimiento del sistema. Se mantuvo el enfoque binario debido a que los datos originales estaban estructurados de esta forma, asegurando consistencia en el procesamiento y tratamiento de la información.

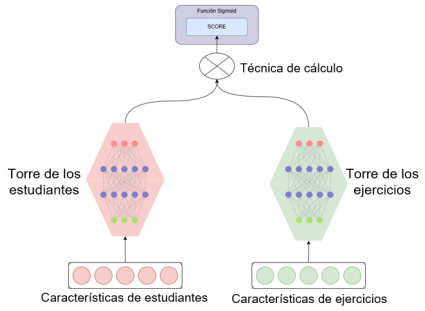

Fig. 5. Arquitectura modular de torres utilizada para la configuración e implementación de los modelos.

5 RESULTADOS

5.1 Métricas

Las métricas utilizadas para evaluar los modelos de recomendación de dos torres fueron: **Precisión@k**, **Recall@k**, **NDCG@k**, **Cobertura**, **Novedad** y **Sesgo de Popularidad**. Estas se calcularon para diferentes valores de k ($k = 5, 10, 15$), con el objetivo de analizar la calidad de las recomendaciones generadas por cada modelo, evaluando tanto su relevancia como su diversidad. A continuación, se describen las fórmulas empleadas para cada métrica:

- **Precisión@k**: Mide la proporción de ejercicios relevantes entre los k ejercicios recomendados.

$$\text{Precisión@k} = \frac{\text{Número de ejercicios relevantes en los top-}k}{k \text{ ejercicios recomendados}}$$

- **Recall@k**: Representa la proporción de ejercicios relevantes que son recuperados dentro de los k ejercicios recomendados.

$$\text{Recall@k} = \frac{\text{Número de ejercicios relevantes en los top-}k}{\text{Total de ejercicios relevantes disponibles}}$$

- **NDCG@k**: Evalúa la calidad del ranking de los ejercicios relevantes. Se basa en el **Discounted Cumulative Gain (DCG)** y su versión ideal, el **IDCG (Ideal DCG)**. Donde Relevancia_i indica la relevancia del ejercicio en la posición i, y IDCG@k es el valor máximo de DCG@k que se alcanzaría con un ranking ideal.

$$\text{DCG@k} = \sum_{i=1}^{k} \frac{\text{Relevancia}_i}{\log_2(i+1)}$$

$$\text{NDCG@k} = \frac{\text{DCG@k}}{\text{IDCG@k}}$$

- **Cobertura**: Indica la proporción de ejercicios únicos recomendados en relación con el total de ejercicios disponibles.

$$\text{Cobertura} = \frac{\text{Número de ejercicios únicos recomendados}}{\text{Total de ejercicios disponibles}}$$

- **Novedad**: Evalúa la diversidad de las recomendaciones, penalizando la aparición de ejercicios altamente populares. Donde N es el total de recomendaciones realizadas, y Popularidad_i es la frecuencia de interacción del ejercicio i en los datos históricos.

$$\text{Novedad} = -\frac{1}{N} \sum_{i \in \text{Ejercicios Recomendados}} \log_2(\text{Popularidad}_i)$$

- **Sesgo de Popularidad**: Representa la propensión del sistema a recomendar ejercicios populares.

$$\text{Sesgo de Popularidad} = \frac{\sum_{s \in \text{Estudiantes}} \sum_{i \in \text{Ejercicios Recomendados}} \text{Popularidad}_i}{\text{Total de Recomendaciones}}$$

Donde Popularidad$_i$ es el número de veces que el ejercicio i ha sido interactuado por cualquier estudiante.

5.2 Análisis de resultados

Las métricas obtenidas para los diferentes modelos de recomendación implementados permitió evaluar el desempeño de los modelos en dos dimensiones principales: relevancia de las recomendaciones y variabilidad de las mismas. A continuación, se presenta los resultados obtenidos en el desarrollo de este proyecto:

Table 3. Evaluación de modelos: Métricas obtenidas con el conjunto de validación

Modelo	Versión de Torres	K=5			K=10			K=15		
		Presición@5	Recall@5	NDCG@5	Presición@10	Recall@10	NDCG@10	Presición@15	Recall@15	NDCG@15
TwoTowerModel	V1	0.6472	0.2353	0.6247	0.6325	0.5088	0.6539	0.6705	0.8221	0.7473
	V2	0.6878	0.2609	0.7208	0.6764	0.5362	0.6789	0.6705	0.8221	0.7440
	V3	0.7366	0.3272	0.7598	0.6935	0.5724	0.7791	0.6705	0.8221	0.8267
TwoTowerModelV1	V1	0.6976	0.3093	0.7336	0.6951	0.5705	0.7824	0.6705	0.8221	0.8006
	V2	0.6878	0.2609	0.7130	0.6894	0.5655	0.7053	0.6705	0.8221	0.7428
	V3	0.7789	0.3580	0.8240	0.7325	0.6137	0.8223	0.6705	0.8221	0.8411
TwoTowerModelV2	V1	0.6846	0.2584	0.6960	0.6732	0.5376	0.6961	0.6705	0.8221	0.7424
	V2	0.5642	0.2070	0.5061	0.6285	0.4682	0.5860	0.6705	0.8221	0.6934
	V3	0.7171	0.2749	0.7113	0.6285	0.4682	0.6507	0.6705	0.8221	0.7332
IntTower	V1	0.6748	0.3003	0.6846	0.7138	0.5909	0.7790	0.6705	0.8221	0.7933
	V2	0.6472	0.2353	0.6247	0.5764	0.4229	0.6041	0.6705	0.8221	0.7432
	V3	0.6244	0.2390	0.5687	0.6285	0.4682	0.5735	0.6705	0.8221	0.6845
SparseTwoTower	V1	0.7447	0.2981	0.7856	0.7358	0.6064	0.7911	0.6705	0.8221	0.8096
	V2	0.7577	0.3187	0.8029	0.6732	0.5319	0.7612	0.6705	0.8221	0.8258
	V3	0.7675	0.3404	0.8046	0.7431	0.6178	0.8201	0.6705	0.8221	0.8338

Table 4. Variabilidad de las recomendaciones según cada modelo

Modelo	Versión de Torres	Cobertura	Novedad	Sesgo de Popularidad
TwoTowerModel	V1	0.5472	0.1393	227.6097
	V2	0.6981	-5.4864	252.9448
	V3	0.6981	-3.4390	245.5201
TwoTowerModelV1	V1	0.5472	0.6624	222.4896
	V2	0.5472	0.9971	218.2727
	V3	0.5472	0.7159	222.5818
TwoTowerModelV2	V1	0.4151	2.1740	41.3195
	V2	0.7925	-0.4719	218.4058
	V3	0.6604	0.6820	195.3013
IntTower	V1	0.6226	6.4810	97.7864
	V2	0.3962	12.6402	0.8052
	V3	0.7736	-0.9043	137.0468
SparseTwoTower	V1	0.6038	-2.5314	102.1416
	V2	0.2264	13.3014	0.0000
	V3	0.7925	-0.2940	216.1578

En términos de relevancia, el modelo **SparseTwoTower (V3)** destacó significativamente. Este modelo alcanzó la mejor Precisión@10 con un valor de 0.7431, Recall@10

de 0.6178 y un NDCG@10 de 0.8201. Además, logró un NDCG@15 de 0.8338, lo que indica que no solo identifica ejercicios relevantes de manera eficiente, sino que también prioriza correctamente aquellos que son más útiles para el usuario. Esto lo posiciona como uno de los modelos más efectivos en términos de relevancia.

Por otro lado, **TwoTowerModel (V3)** también mostró un desempeño competitivo, logrando un NDCG@15 de 0.8267. Aunque sus métricas de Precisión@10 de 0.6935 y Recall@10 de 0.5724 fueron ligeramente inferiores a las de **SparseTwoTower (V3)**, este modelo se mantiene como una opción sólida para organizar recomendaciones relevantes.

En tanto, el modelo **TwoTowerModelV1 (V3)** demostró un rendimiento sobresaliente, alcanzando el NDCG@15 más alto de todos los modelos evaluados con un valor de 0.8411 y una Precisión@10 de 0.7325, posicionándose como una alternativa robusta en escenarios donde el orden de las recomendaciones es prioritario.

En cuanto a la diversidad y balance, las métricas de Cobertura, Novedad y Sesgo de Popularidad de la Tabla 4 permitieron evaluar cómo los modelos manejan la propensión a depender de ejercicios populares y su capacidad de diversificar las recomendaciones.

El modelo **SparseTwoTower (V3)** mantuvo una alta Cobertura de 0.7925, aunque mostró una ligera caída en Novedad con un valor de -0.2940 y un Sesgo de Popularidad moderado de 216.1578. Esto refleja que, aunque el modelo es equilibrado, podría mejorar en su capacidad para recomendar ejercicios menos comunes.

Por otro lado, el modelo **IntTower (V2)** sobresalió en términos de Novedad, logrando un valor de 12.6402, junto con el menor Sesgo de Popularidad de 0.8052 entre todos los modelos evaluados. Esto indica que evita recomendaciones dominadas por ejercicios populares, fomentando una mayor diversidad en las sugerencias. Sin embargo, su baja Cobertura de 0.3962 limita su capacidad para abarcar una mayor cantidad de ejercicios únicos, lo que afecta su aplicabilidad en contextos más amplios.

Por último, **TwoTowerModelV1 (V3)** presentó una Novedad de 0.7159 y un Sesgo de Popularidad moderado de 222.5818, lo que refleja un buen equilibrio entre relevancia y diversidad, aunque no alcanza el nivel de **SparseTwoTower (V3)**. Por su parte, **TwoTowerModelV2 (V2)** obtuvo la Cobertura más alta de 0.7925, pero su Novedad de -0.4719 y su Sesgo de Popularidad de 218.4058 indican una dependencia significativa hacia ejercicios populares, lo que limita su capacidad de generar recomendaciones diversas.

5.3 Mejor modelo

Tras un análisis de los resultados obtenidos, se determinó que el modelo **SparseTwoTower (V3)** se posiciona como la opción más robusta y equilibrada entre los evaluados. Este modelo destaca por su capacidad para generar recomendaciones altamente relevantes y con una amplia cobertura, lo que lo convierte en una herramienta eficaz para ofrecer sugerencias personalizadas que maximizan tanto el aprendizaje como la exploración de ejercicios. Sin embargo, se identificaron oportunidades de mejora, particularmente en la recomendación de ejercicios menos populares y en el incremento de la diversidad en las sugerencias generadas.

Con el modelo equilibrado para generar recomendaciones personalizadas, se llevaron a cabo dos tipos de pruebas. En primer lugar, se realizaron recomendaciones para un estudiante seleccionado aleatoriamente del conjunto de datos de prueba, representando a un estudiante que ya había aprobado la asignatura. En segundo lugar, se generaron recomendaciones para un estudiante elegido aleatoriamente del conjunto de estudiantes que reprobaron la asignatura.

Esto tiene como objetivo medir las diferencias entre los valores promedio de las características de los ejercicios recomendados y los valores promedio de los ejercicios previamente realizados por los estudiantes. Por tanto, se buscó evaluar si el modelo era adapto sus recomendaciones en función de las características individuales y necesidades de cada estudiante.

La figura 6 muestra las diferencias medias entre los ejercicios recomendados y aquellos con los que ya había interactuado un estudiante aprobado, en este caso el estudiante 922, al usar el modelo **SparseTwoTower (V3)**. En este caso, se observa que la mayor diferencia positiva corresponde a la variable skill, mientras que la mayor diferencia negativa está en score_a. Esto indica que el sistema recomienda ejercicios con mayores valores de skill, pero tiende a penalizar o evitar ejercicios asociados a score_a para el estudiante 922.

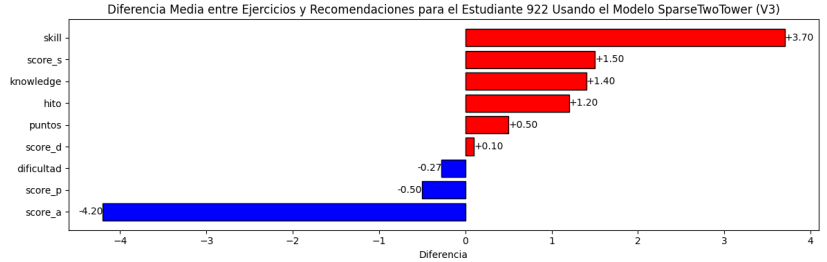

Fig. 6. Diferencia media entre ejercicios y recomendaciones del estudiante 922 usando el modelo SparseTwoTower (V3).

Por el otro lado, en la figura 7 se presenta las diferencias medias para un estudiante reprobado, en este caso el estudiante 816, bajo el mismo modelo. Aquí, las diferencias positivas más altas corresponden a las variables score_d y score_s, lo que sugiere que el sistema está priorizando ejercicios que potencian la descomposición y la algoritmia. Las diferencias negativas son menos pronunciadas, lo que implica que el sistema está más equilibrado en las recomendaciones, pero sigue evitando ejercicios relacionados con valores bajos en score_a, es decir, ejercicios con bajo nivel de abstracción.

En conjunto, las gráficas de las figuras 7 y 6 reflejan que el modelo ajusta las recomendaciones de manera diferente dependiendo del desempeño previo del estudiante, con un enfoque más exploratorio para estudiantes aprobados y uno más enfocado en reforzar habilidades para estudiantes reprobados.

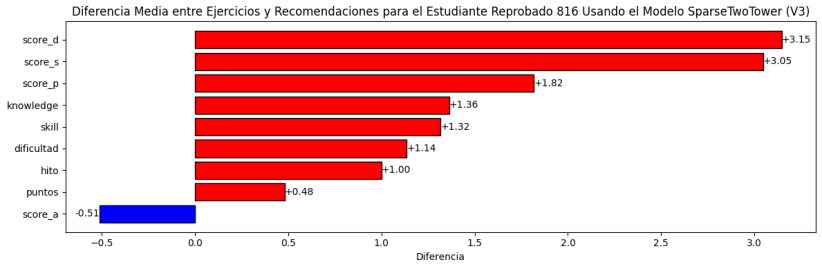

Fig. 7. Diferencia media entre ejercicios y recomendaciones del estudiante reprobado 816 usando el modelo SparseTwoTower (V3).

6 CONCLUSIONES

El desarrollo del sistema de recomendación de ejercicios de programación basado en un modelo de dos torres, realizado en el contexto de la asignatura "Introducción a la Programación" en la Universidad Andrés Bello, ha demostrado ser una solución efectiva para mejorar el aprendizaje personalizado de los estudiantes. Este proyecto logró implementar un sistema que ofrece recomendaciones ajustadas a las características individuales de los usuarios, abordando desafíos comunes como la sobrecarga informativa y las diferencias en niveles de habilidad entre estudiantes.

El modelo SparseTwoTower (V3) se destacó como el más robusto y equilibrado entre los evaluados. Sus resultados reflejan una alta precisión y relevancia en las recomendaciones, con métricas destacadas como un NDCG@15 de 0.8338, que evidencia su capacidad para identificar y priorizar ejercicios útiles. Además, el modelo mostró una notable adaptabilidad, ajustando las recomendaciones según las necesidades específicas de los estudiantes. Por ejemplo, en el caso de estudiantes con bajo desempeño, las recomendaciones se enfocaron en reforzar habilidades clave, mientras que, para estudiantes con mejor rendimiento, se priorizó la exploración de nuevos desafíos.

A pesar de los logros obtenidos, el sistema enfrenta ciertas limitaciones. Una de las principales es su dependencia de los datos de entrenamiento, que condicionan el desempeño del modelo a la calidad y cantidad de los datos disponibles. Asimismo, se identificó un sesgo hacia ítems populares, lo que afecta la diversidad de las recomendaciones y limita el descubrimiento de opciones menos comunes. También se observó una falta de diversidad en las interacciones capturadas, lo que podría influir negativamente en la personalización de las recomendaciones. Otro desafío relevante es el problema del cold start, tanto para nuevos estudiantes como para ejercicios sin datos históricos, que dificulta la generación de recomendaciones precisas en esos casos. La representación de las características también fue una limitación, ya que las variables utilizadas no capturaron toda la complejidad de las preferencias y habilidades de los estudiantes. Además, el sistema no consideró explícitamente el contexto temporal, lo que podría haber mejorado la relevancia de las recomendaciones en función de patrones estacionales o de progreso académico. Por último, la escalabilidad y la rigidez

del diseño del modelo plantean desafíos a futuro, especialmente si se planea expandir el sistema a conjuntos de datos más grandes o dominios educativos más amplios.

En cuanto a los trabajos futuros, se propone abordar estas limitaciones mediante diversas estrategias. En primer lugar, sería fundamental incorporar técnicas de regularización del sesgo para mitigar la dependencia hacia ítems populares y fomentar una mayor equidad en las recomendaciones. Asimismo, la optimización de la métrica de novedad podría mejorar la diversidad, incentivando recomendaciones que incluyan ejercicios menos comunes. También se recomienda explorar alternativas de modelado, como otras arquitecturas híbridas o con enfoques basados en aprendizaje por refuerzo, que permitan una mayor adaptabilidad y eficiencia.

El análisis del contexto temporal es otra área prometedora; integrar información sobre el progreso académico o la dinámica temporal de las interacciones podría enriquecer las recomendaciones. Además, una representación más compleja de las características, incluyendo datos demográficos, estilo de aprendizaje o preferencias declaradas por los estudiantes, permitiría construir un modelo más robusto y preciso. Finalmente, trabajar en la flexibilidad del diseño del sistema para hacerlo más modular y escalable será esencial si se busca implementarlo en otros contextos educativos o expandir su uso a poblaciones estudiantiles más amplias.

En conclusión, este proyecto no solo cumplió con los objetivos planteados, sino que también evidenció el potencial de los sistemas de recomendación basados en modelos avanzados como el de dos torres para transformar la experiencia educativa. Los resultados alcanzados, junto con las oportunidades de mejora y expansión, posicionan esta investigación como una contribución significativa al desarrollo de tecnologías educativas personalizadas en un entorno académico. Las propuestas para trabajos futuros abren nuevas oportunidades para optimizar y ampliar este sistema, fortaleciendo su impacto en la educación personalizada.

7 EXPRESIONES DE GRATITUD

Deseo expresar mi más profundo agradecimiento a mis hermanas, Lizabeth y Vithdiley, cuyo apoyo incondicional y generosidad hicieron posible que tuviera la oportunidad de cursar mis estudios universitarios. Su esfuerzo y confianza en mí han sido fundamentales para alcanzar este logro, y siempre estaré en deuda con ustedes por ello.

A mis padres, Edgar y Verso, les agradezco profundamente por inculcarme valores sólidos que han sido guía y fortaleza en cada etapa de mi vida. Su amor, enseñanzas y ejemplo me han permitido forjar el carácter necesario para enfrentar los desafíos que este camino ha presentado.

A todos ustedes, mi más sincera gratitud. Este logro no habría sido posible sin su apoyo constante y su influencia en mi vida.

REFERENCES

[1] Afsar, M. M., Crump, T., and Far, B. H. Reinforcement learning based recommender systems: A survey. *CoRR abs/2101.06286* (2021).

[2] Covington, P., Adams, J., and Sargin, E. Deep neural networks for youtube recommendations. In *Proceedings of the 10th ACM Conference on Recommender Systems* (New York, NY, USA, 2016), RecSys

'16, Association for Computing Machinery, p. 191–198.

[3] da Silva, F. L., Slodkowski, B. K., da Silva, K. K. A., and Cazella, S. C. A systematic literature review on educational recommender systems for teaching and learning: research trends, limitations and opportunities. *Education and Information Technologies 28*, 3 (March 2023), 3289–3328.

[4] Fan, G., Zhang, C., Wang, K., and Chen, J. Mv-han: A hybrid attentive networks based multi-view learning model for large-scale contents recommendation. In *Proceedings of the 37th IEEE/ACM International Conference on Automated Software Engineering* (New York, NY, USA, 2023), ASE '22, Association for Computing Machinery.

[5] George, G., and Lal, A. M. Review of ontology-based recommender systems in e-learning. *Computers Education 142* (2019), 103642.

[6] Khanal, S. S., Prasad, P. W. C., Alsadoon, A., and Maag, A. A systematic review: machine learning based recommendation systems for e-learning. *Education and Information Technologies 25*, 4 (July 2020), 2635–2664.

[7] Li, Q., and Kim, J. A deep learning-based course recommender system for sustainable development in education. *Applied Sciences 11*, 19 (2021).

[8] Li, X., Chen, B., Guo, H., Li, J., Zhu, C., Long, X., Li, S., Wang, Y., Guo, W., Mao, L., Liu, J., Dong, Z., and Tang, R. Inttower: The next generation of two-tower model for pre-ranking system. In *Proceedings of the 31st ACM International Conference on Information & Knowledge Management* (New York, NY, USA, 2022), CIKM '22, Association for Computing Machinery, p. 3292–3301.

[9] Ma, J., Zhao, Z., Yi, X., Yang, J., Chen, M., Tang, J., Hong, L., and Chi, E. H. Off-policy learning in two-stage recommender systems. In *Proceedings of The Web Conference 2020* (New York, NY, USA, 2020), WWW '20, Association for Computing Machinery, p. 463–473.

[10] Pinho, P. C. R., Barwaldt, R., Espíndola, D., Torres, M., Pias, M., Topin, L., Borba, A., and Oliveira, M. Developments in educational recommendation systems: a systematic review. In *2019 IEEE Frontiers in Education Conference (FIE)* (2019), pp. 1–7.

[11] Rivera, A. C., Tapia-Leon, M., and Lujan-Mora, S. Recommendation systems in education: A systematic mapping study. In *Proceedings of the International Conference on Information Technology & Systems (ICITS 2018)* (Cham, 2018), Á. Rocha and T. Guarda, Eds., Springer International Publishing, pp. 937–947.

[12] Su, L., Yan, F., Zhu, J., Xiao, X., Duan, H., Zhao, Z., Dong, Z., and Tang, R. Beyond two-tower matching: Learning sparse retrievable cross-interactions for recommendation. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval* (New York, NY, USA, 2023), SIGIR '23, Association for Computing Machinery, p. 548–557.

[13] Urdaneta-Ponte, M. C., Mendez-Zorrilla, A., and Oleagordia-Ruiz, I. Recommendation systems for education: Systematic review. *Electronics 10*, 14 (2021).

[14] Wang, J., Zhu, J., and He, X. Cross-batch negative sampling for training two-tower recommenders. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval* (New York, NY, USA, 2021), SIGIR '21, Association for Computing Machinery, p. 1632–1636.

[15] Yang, J., Yi, X., Zhiyuan Cheng, D., Hong, L., Li, Y., Xiaoming Wang, S., Xu, T., and Chi, E. H. Mixed negative sampling for learning two-tower neural networks in recommendations. In *Companion Proceedings of the Web Conference 2020* (New York, NY, USA, 2020), WWW '20, Association for Computing Machinery, p. 441–447.

[16] Zhang, S., Yao, L., Sun, A., and Tay, Y. Deep learning based recommender system: A survey and new perspectives. *ACM Comput. Surv. 52*, 1 (Feb. 2019).

[17] Zhang, Y., Dong, X., Ding, W., Li, B., Jiang, P., and Gai, K. Divide and conquer: Towards better embedding-based retrieval for recommender systems from a multi-task perspective, 2023.

[18] Zhao, Y., Wang, Y., Liu, Y., Cheng, X., Aggarwal, C., and Derr, T. Fairness and diversity in recommender systems: A survey, 2024.