Sistema inteligente de recomendación de moda basado en Deep Learning

Mendoza Bravo, María Victoria y Montaño Moncada, Yandry Damián

Departamento de Ciencias de la Computación

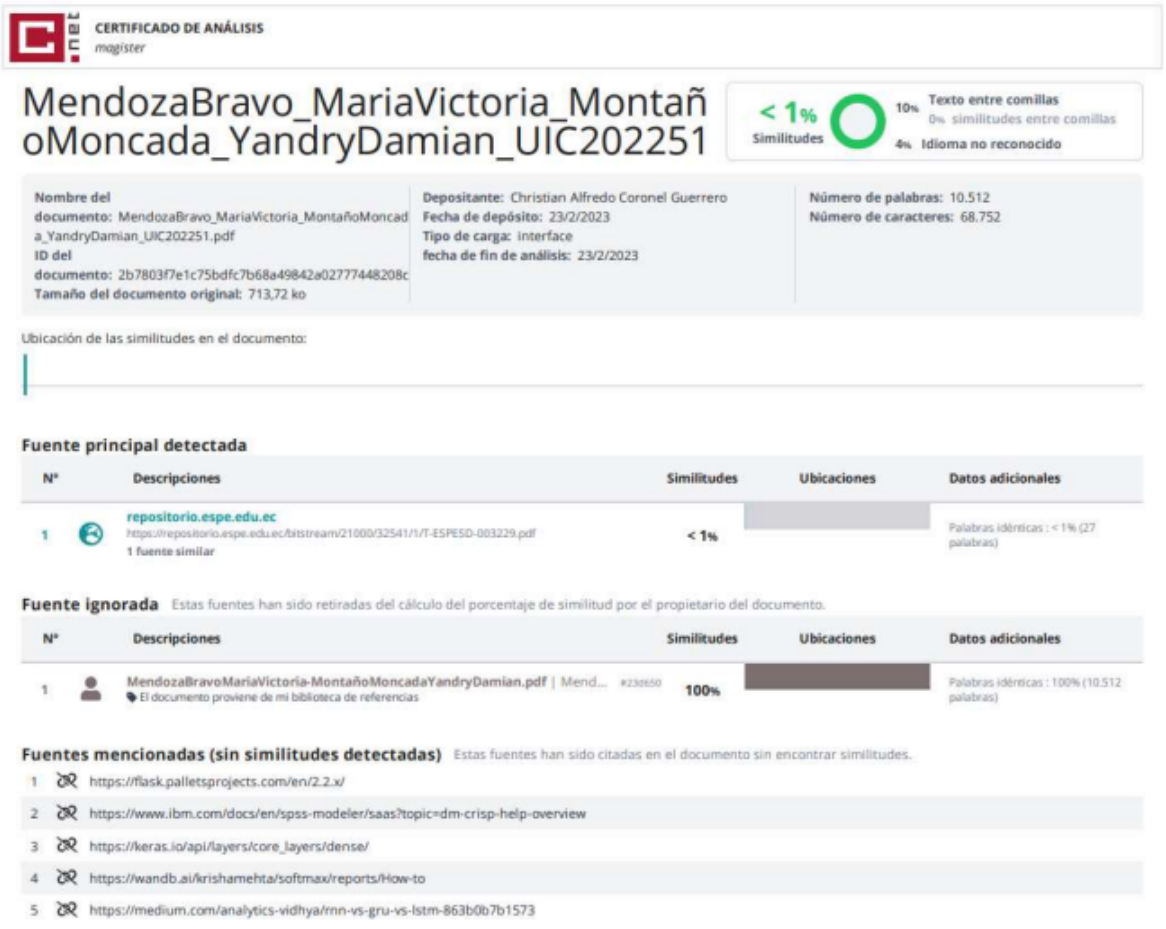
Carrera de Ingeniería en Tecnologías de la Información

Trabajo de integración curricular, previo a la obtención del título de Ingeniería en Tecnologías de la Información

Ing. Salazar Armijos, Diego Ricardo

22 de febrero de 2023

Reporte de verificación de contenido



CERTIFICADO DE ANÁLISIS
magister

MendozaBravo_MariaVictoria_MontañoMoncada_YandryDamian_UIC202251

< 1% Similitudes
10% Texto entre comillas
0% similitudes entre comillas
4% Idioma no reconocido

Nombre del documento: MendozaBravo_MariaVictoria_MontañoMoncada_YandryDamian_UIC202251.pdf
ID del documento: 2b7803f7e1c75bdfc7b68a49842a02777448208c
Tamaño del documento original: 713,72 ko

Depositante: Christian Alfredo Coronel Guerrero
Fecha de depósito: 23/2/2023
Tipo de carga: interface
fecha de fin de análisis: 23/2/2023

Número de palabras: 10.512
Número de caracteres: 68.752

Ubicación de las similitudes en el documento:

Fuente principal detectada

Nº	Descripciones	Similitudes	Ubicaciones	Datos adicionales
1	repositorio.espe.edu.ec https://repositorio.espe.edu.ec/bitstream/21000/32541/1/T-ESPESD-003229.pdf 1 fuente similar	< 1%		Palabras idénticas : < 1% (27 palabras)

Fuente ignorada Estas fuentes han sido retiradas del cálculo del porcentaje de similitud por el propietario del documento.

Nº	Descripciones	Similitudes	Ubicaciones	Datos adicionales
1	MendozaBravoMariaVictoria-MontañoMoncadaYandryDamian.pdf | Mend... #230650 El documento proviene de mi biblioteca de referencias	100%		Palabras idénticas : 100% (10.512 palabras)

Fuentes mencionadas (sin similitudes detectadas) Estas fuentes han sido citadas en el documento sin encontrar similitudes.

1 https://flask.palletsprojects.com/en/2.2.x/
2 https://www.ibm.com/docs/en/spss-modeler/saas?topic=dm-crisp-help-overview
3 https://keras.io/api/layers/core_layers/dense/
4 https://wandb.ai/krishamehta/softmax/reports/How-to
5 https://medium.com/analytics-vidhya/rnn-vs-gru-vs-lstm-863b0b7b1573

Firma

Firmado electrónicamente por:
DIEGO RICARDO SALAZAR ARMIJOS

.....
Salazar Armijos, Diego Ricardo

Director

DEPARTAMENTO DE CIENCIAS DE LA COMPUTACIÓN

CARRERA DE INGENIERÍA EN TECNOLOGÍAS DE LA INFORMACIÓN

CERTIFICACIÓN

Certifico que el trabajo de integración curricular, "**Sistema inteligente de recomendación de moda basado en Deep Learning**" fue realizado por los señores **Mendoza Bravo, María Victoria y Montaño Moncada, Yandry Damián** el cual ha sido revisado y analizado en su totalidad por la herramienta de verificación de similitud de contenido; por lo tanto cumple con los requisitos legales, teóricos, científicos, técnicos y metodológicos establecidos por la Universidad de las Fuerzas Armadas ESPE, razón por la cual me permito acreditar y autorizar para que lo sustente públicamente.

Santo Domingo de los Tsáchilas, 22 de febrero de 2023

Firma:

.....

Salazar Armijos, Diego Ricardo

C.C. 1710481027

DEPARTAMENTO DE CIENCIAS DE LA COMPUTACIÓN

CARRERA DE INGENIERÍA EN TECNOLOGÍAS DE LA INFORMACIÓN

RESPONSABILIDAD DE AUTORÍA

Nosotros, **Mendoza Bravo, María Victoria** y **Montaño Moncada, Yandry Damián,** con cedulas de ciudadanía n° 1727157743 y n° 2350461899, declaramos que el contenido, ideas y criterios del trabajo de integración curricular: **Sistema inteligente de recomendación de moda basado en Deep Learning**, es de nuestra autoría y responsabilidad, cumpliendo con los requisitos legales, teóricos, científicos, técnicos, y metodológicos establecidos por la Universidad de las Fuerzas Armadas ESPE, respetando los derechos intelectuales de terceros y referenciando las citas bibliográficas.

Santo Domingo de los Tsáchilas, 22 de febrero de 2023

Firmas:

.....

Mendoza Bravo, María Victoria

C.C.: 1727157743

.....

Montaño Moncada, Yandry Damián

C.C.: 2350461899

DEPARTAMENTO DE CIENCIAS DE LA COMPUTACIÓN

CARRERA DE INGENIERÍA EN TECNOLOGÍAS DE LA INFORMACIÓN

AUTORIZACIÓN DE PUBLICACIÓN

Nosotros, **Mendoza Bravo, María Victoria** y **Montaño Moncada, Yandry Damián,** con cédulas de ciudadanía n° 1727157743 y n° 2350461899 autorizamos a la Universidad de las Fuerzas Armadas ESPE publicar el trabajo de integración curricular: **Sistema inteligente de recomendación de moda basado en Deep Learning**, en el Repositorio Institucional, cuyo contenido, ideas y criterios son de nuestra responsabilidad.

Santo Domingo de los Tsáchilas, 22 de febrero de 2023.

Firmas:

.....

Mendoza Bravo, María Victoria

C.C.: 1727157743

.....

Montaño Moncada, Yandry Damián

C.C.: 2350461899

Dedicatoria

Dedico este trabajo principalmente a Dios por darme la vida, la oportunidad de ser quien soy y la fuerza necesaria para no rendirme.

A mis padres, por apoyarme incondicionalmente en cada paso y educarme en la disciplina de estudiar y luchar por mis sueños.

A mis hermanas, familia y amigos que creyeron en mí, por el apoyo que me brindaron en toda esta etapa.

María Victoria Mendoza Bravo

Dedicatoria

Dedico este trabajo principalmente a mis padres, por apoyarme en cada uno de los pasos que he decido dar, por todo el esfuerzo y tiempo que han dedicado en educarme y lograr mis sueños.

A mis hermanos, y amigos que creyeron en mí y me brindaron todo su apoyo de forma incondicional.

Yandry Damián Montaño Moncada

Agradecimientos

Agradecemos a nuestros padres por su apoyo incondicional, por su dedicación y esfuerzo para ayudarnos a cumplir nuestras metas y sobre todo, por ser pilares fundamentales en este camino de preparación.

Al Ing. Diego Salazar y a la Ing. Mayra Macas por su tiempo, conocimientos y por ser parte importante en el desarrollo de este proyecto.

A todos los docentes que han sido parte de nuestra carrera universitaria y han compartido con nosotros sus consejos y enseñanzas.

A nuestros compañeros, por el tiempo agradable que compartimos.

Victoria Mendoza & Yandry Montaño

Índice de contenido

Carátula 1

Reporte de verificación de contenido 2

Certificación del director 3

Responsabilidad de Autoría 4

Autorización de Publicación 5

Dedicatoria 6

Agradecimientos 8

Índice de figuras 11

Índice de tablas 11

Resumen 12

Abstract 13

Capítulo I: Descripción del proyecto 14

Introducción 14

Antecedentes 16

Justificación 19

Alcance 20

Objetivos 20

Objetivo General 20

Objetivos Específicos 20

Capítulo II: Marco Teórico 21

Deep Learning en Sistemas de Recomendación 21

Modelos de Deep Learning 22

Redes Neuronales Convolucionales (CNN) 22

Redes Neuronales Recurrentes (RNN) 23

Metodologías Orientadas a Data Mining 25

Metodología KDD 25

Metodología SEMMA 26

Metodología CRISP-DM 26

Aprendizaje no supervisado 28

Autocodificadores 29

Capítulo III: Metodología 30

Recolección de datos 30

Preprocesamiento de datos 31

Extracción de características 33

Modelado 33

Entrenamiento y evaluación 34

Despliegue 34

Capítulo IV: Experimentación 39

Configuración experimental y preprocesamiento de datos 39

Extracción de características 39

Entrenamiento y evaluación del modelo 40

Evaluación experimental 42

Capítulo V: Conclusiones y Recomendaciones 45

Conclusiones 45

Recomendaciones 46

Bibliografía 47

Índice de figuras

Figura 1 *Arquitectura de la red CNN* 23

Figura 2 *Framework para la elaboración del modelo de IA* 30

Figura 3 *Librerías requeridas para la implementación de Flask* 35

Figura 4 *Función para la lectura de imagen* 36

Figura 5 *Función principal para la ejecución del modelo* 36

Figura 6 *Código HTML de la página inicial del proyecto* 37

Figura 7 *Interfaz gráfica de la aplicación web* 38

Figura 8 *Arquitectura del modelo* 41

Figura 9 *Evaluación de los modelos* 44

Índice de tablas

Tabla 1 Comparación de las metodologías orientadas a Data Mining 27

Tabla 2 Resumen del rendimiento de los modelos 43

Resumen

Los sistemas de recomendación de productos facilitan al usuario la navegación entre toda la información que existe en internet, específicamente, en el área del comercio electrónico, entretenimiento y redes sociales. En un inicio, dichos sistemas utilizaban procesos tradicionales para realizar sugerencias de consumo, sin embargo, con el avance de la tecnología e implementaciones de inteligencia artificial, los sistemas de recomendación basados en imágenes, se vuelven más eficientes en las recomendaciones. La industria de la moda, por su parte, ha experimentado un rápido crecimiento en los últimos años y por la gran cantidad de datos que se generan en este ámbito, es fundamental la implementación de sistemas de recomendación de moda para el comercio electrónico del área textil. Por esta razón, el presente estudio propone un sistema inteligente de recomendación de moda basado en Deep Learning. Para los fines pertinentes se utiliza aprendizaje semi supervisado con redes neuronales convolucionales recurrentes que permiten la extracción de características y un clasificador compuesto de capas densas. Logrando, mediante este esquema, una precisión del 98% y una exactitud del 92% respecto a la tendencia de recomendación de prendas. El dataset utilizado contiene imágenes obtenidas de internet de diversas marcas (H&M, Zara, etc.), combinadas con un porcentaje del conjunto de datos público “Deep Fashion”. Adicionalmente, los datos son permutados para mejorar el aprendizaje de las neuronas.

Palabras clave: aprendizaje profundo, sistema de recomendación, aprendizaje semi supervisado, redes neuronales recurrentes, autocodificadores.

Abstract

Recommendation systems make it easier for the user to navigate through all the information available on the Internet, specifically in e-commerce, entertainment, and social networks. Initially, these systems used traditional processes to make consumer suggestions; however, with the advancement of technology and the implementation of artificial intelligence, image-based recommendation systems are becoming more efficient in making recommendations. The fashion industry, on the other hand, has experienced rapid growth in recent years and due to the large amount of data generated in this area, it is essential to implement fashion recommendation systems for e-commerce in the textile area. For this reason, this study proposes an intelligent fashion recommendation system based on Deep Learning. For the relevant purposes, semi-supervised learning is used with recurrent convolutional neural networks that allow feature extraction and a classifier composed of dense layers. Achieving, through this scheme, a precision of 98% and an accuracy of 92% with respect to the garment recommendation trend. The dataset used contains images obtained from the internet of various brands (H&M, Zara, etc.), combined with a percentage of the public dataset "Deep Fashion". Additionally, the data is permuted to improve the learning of the neurons.

Keywords: deep learning, recommender system, semi-supervised learning, recurrent neural networks, autoencoders.

Capítulo I: Descripción del proyecto

Introducción

Según datos de la Conferencia de las Naciones Unidas sobre Comercio y Desarrollo (UNCTAD, 2020), luego de la pandemia de Covid19, las actividades de comercio electrónico crecieron considerablemente. En la primera encuesta realizada a 1819 personas de 10 nacionalidades diferentes, para determinar el impacto de la crisis sanitaria mundial, se obtuvo como resultado que entre las 11 fuentes de consumo, donde aumentaron los compradores en línea, la moda y los accesorios se encuentran en el cuarto lugar.

El comercio electrónico respecto a las compras de moda en línea ha evidenciado grandes avances en los últimos años y se espera que sus ingresos a nivel mundial, se dupliquen en comparación con 2018, alcanzando los 872.000 millones de dólares para 2023 (Turnbull, 2022). Así mismo, en un estudio realizado en Ecuador en marzo de 2021, se determinó que la comercialización online de prendas de vestir, está dentro de las 10 categorías de productos más vendidos durante la pandemia.

A pesar de los grandes beneficios que el comercio electrónico de moda aporta, se presentan dificultades para los clientes, al momento de explorar atuendos entre la inmensa cantidad de prendas de vestir que existen en internet. Por este motivo, se considera pertinente la implementación de sistemas de recomendación de moda eficientes e inteligentes, que permita a los usuarios la clasificación, orden y transmisión de los contenidos de interés, para seleccionar de mejor forma las prendas que más se ajustan a su estilo.

Inicialmente, los sistemas de recomendación de moda se fundamentaban en opiniones explícitas de los usuarios, sobre determinados artículos. Los comentarios o calificaciones emitidos por estos, permitía deducir los artículos más solicitados de los clientes, sin embargo, lo hacían de una manera muy general, pues se basaban en la información de muestras pequeñas y no representativas de personas (Chakraborty, y otros, 2021).

Por esta razón, se inició con la automatización de esta tarea con ayuda de algoritmos de Machine Learning y específicamente, Deep Learning, para los sistemas de recomendación de moda. Estos sistemas inteligentes no solo se basan en comentarios de clientes, sino, en un estudio de los gustos de cada persona según lo que busca y la interacción con los productos que ve en internet, de manera que las recomendaciones se realizan de forma más personalizada para cada usuario.

En el presente proyecto se propone un sistema de recomendación de moda que brindará beneficios tanto a los clientes como a los comerciantes en línea, ya que no solo proporcionará objetivos claros para los clientes, sino que también los alienta a comprar un atuendo. Más específicamente, se pretende crear un sistema inteligente de recomendación de moda basado en Deep Learning, que sea capaz de devolver los artículos compatibles dado un artículo de consulta de interés. Este estudio seguirá CRISP-DM (IBM, 2021) como metodología.

Este documento está conformado por cinco capítulos estructurados de la siguiente manera: en el primer capítulo se describen los aspectos principales del proyecto como antecedentes, justificación, alcance y sistema de objetivos.

En el segundo capítulo se realiza un análisis teórico acerca de los conceptos básicos respecto al tema y la descripción del marco de trabajo a implementar.

En el tercer capítulo se detalla la implementación de la metodología de desarrollo utilizada con la descripción paso a paso.

En el cuarto capítulo se explica de manera específica la experimentación con los datos para lograr un buen resultado del modelo de aprendizaje profundo y los distintos escenarios elaborados.

Finalmente, en el quinto capítulo, se describen las conclusiones y recomendaciones luego de efectuar y comprobar la funcionalidad del modelo.

Antecedentes

Para la elaboración de sistemas de recomendación de moda, existen diferentes técnicas y métodos que pueden ser empleados. En particular, se utilizan modelos de Machine Learning tradicionales como Clustering y también, algoritmos de Deep Learning como Redes Neuronales Convolucionales (CNN), Memoria a Largo y Corto Plazo Bidireccional (Bi-LSTM), Redes Generativas Adversariales (GAN) y k-Vecino más cercano (kNN).

En uno de los sistemas donde se implementa Clustering (Kyprianidis , Kotouza, Tsarouchis, Chrysopoulos, & Mitkas, 2020), crean un asistente de un diseñador de moda, donde dicha arquitectura, recomienda los artículos que se pueden combinar según su forma, por ejemplo, una blusa, de tamaño corto, pertenece a una mujer, y puede usarse con un collar. Se basa en los metadatos que contienen las imágenes. Además, está alimentado por dos fuentes de datos diferentes: una interna y una externa. La fuente de datos interna hace referencia a la empresa en la que se emplee el sistema y la externa son las fuentes de internet de tiendas conocidas como ASOS, Shtterstock, Zalando y s.Oliver.

Los algoritmos empleados fueron agrupación de modos K (K-Modes Clustering), Partición en torno a los medoides (PAM), Agrupación jerárquica (HAC), Método de agrupación jerárquica basado en la frecuencia multimétrica (FBHC) y VarSelCluster (VarSel). Con cada algoritmo, se evaluaron las siguientes métricas: “entropía”, cuantifica el valor esperado de la información contenida en los clústeres; la “silueta”, valida la coherencia de los clústeres; la “suma del error cuadrático (WSS)”, define la distancia total de los puntos de datos desde los centroides de sus clústeres y la “identidad” expresa el porcentaje de datos obtenidos en el clúster con una alineación exacta referente a las etiquetas de la característica. FBHC presentó mejores resultados en cuanto la “entropía” (18%) y la “identidad” (38%); mientras que, para la “silueta” y “WSS”, PAM representa el mejor resultado con un 46% y un 23% respectivamente.

Siguiendo con los modelos de Machine Learning tradicionales, (Yong-Goo , Yoon-Jae , Min-Cheol , Seo-Won, & Sung-Jea , 2019), en su modelo para perfeccionar las técnicas de

recomendación de ropa, emplea el algoritmo de K-Means para agrupar los vectores de estilo de las prendas de cada clase y así, lograr separar correctamente el vector de la ropa en estilo y categoría.

También utiliza algoritmos de Deep Learning como una CNN siamesa, a la que se le incluyó una capa de extracción de características de estilo (SFE), y se utilizaron los algoritmos VGG16, GoogleNet y MobileNet para su entrenamiento. La evaluación del sistema realizada en base al Área Bajo la Curva (AUC), dio como resultado un 87%. A pesar de que es un porcentaje relativamente elevado, no varía mucho en comparación a una CNN Siamesa sin incluir la capa SFE, donde el menor resultado se obtiene del algoritmo de GoogleNet con un 83%.

En (Enhancing fashion recommendation with visual compatibility relationship, 2019) se crea un sistema de recomendación basado en CNN, utilizando algoritmos como PopRank, para la recomendación de los productos más populares; Clasificación personalizada bayesiana (BPR-MF), para la optimización de preferencias por pares entre elementos observados y no observados; Clasificación visual personalizada bayesiana (VBPR), un sistema de recomendación basado en imágenes de última generación; Método de clasificación personalizada bayesiana profunda (DVBPR), una extensión del VBPR con representación de imágenes “conscientes de la moda” y CO-BPR, que es el modelo propuesto.

Este sistema de recomendación de moda se basa en dos fuentes de datos: la primera es Amazon, con la que se alcanza un porcentaje del 77% según la evaluación del AUC y un 71% sobre la segunda fuente de datos, llamada Tradesy.com. Además de que los resultados, según el modelo propuesto CO-BPR, no son tan elevados, apenas varían un 0,02% respecto al modelo pre entrenado VBPR para ambos conjuntos de datos.

Para los sistemas de recomendación existen también algoritmos que ya están pre entrenados como es el caso de AlexNet; un algoritmo originado para la clasificación y registro de imágenes. (Shankar, Narumanchi, Ananya, Kompalli , & Chaudhury, 2017) y (Lin, Yang, Liu ,

Hsiao , & Chen , 2015) utilizan este algoritmo para construir su modelo. En (Shankar, Narumanchi, Ananya, Kompalli , & Chaudhury, 2017) se centran en la creación de una red neuronal convolucional profunda CNN VisNet, que permite identificar las similitudes de la ropa para luego buscarlas en una aplicación móvil de la India, llamada “Flipkart”, que es una tienda de artículos de moda. El objetivo es lograr identificar muy bien el elemento con todos sus detalles, para recomendar artículos de vestir, lo más parecido posible a la prenda en cuestión, dentro de la aplicación.

Para medir el rendimiento del módulo, se controló la tasa de conversión CVR, haciendo referencia al porcentaje de usuarios que añaden un producto en su lista, luego de presionar el botón “similar”, con el que se logró un 26%. De igual forma, en (Lin, Yang, Liu , Hsiao , & Chen , 2015) se pretende crear un sistema que sugiera los elementos más parecidos a la prenda de consulta, según el conjunto de datos obtenidos de “Yahoo! Shopping”. Este estudio también se basa en la implementación de CNN, con el algoritmo pre entrenado AlexNet, logrando así, una precisión del 85%.

Por otro lado, un modelo que complementa los sistemas de recomendación de moda es Bi-LSTM . (Han, Wu, Jiang, & Davis, 2017) y (Zhan, y otros, 2019) utilizan este modelo para tener una perspectiva de los conjuntos de ropa, en el que puedan avanzar y retroceder para estudiar cada artículo que conforma el outfit. En ambos trabajos, se realizó un sistema que pueda sugerir una prenda de vestir faltante en un conjunto, o proporcionar un conjunto completo según el elemento que proporcione el usuario.

Según las métricas de evaluación, se obtuvo un AUC del 90% en el primer caso, sobre el conjunto de datos de polyvore, mientras que en el segundo modelo, se logró un mejor resultado con CSN, alcanzando un 96% en el AUC, bajo el mismo conjunto de datos. En (Zhan, y otros, 2019) se compara este resultado con la utilización de un dataset con imágenes de la calle, pero el porcentaje del resultado disminuye a un 86%.

Además de los dataset de moda públicos en la web y la información que se puede obtener en tiendas de ropa online, también se puede utilizar las redes sociales como fuente de datos para entrenar el modelo. (Jaradat, Dokoohaki, Hammar, Wara, & Matskin, 2018) se enfocan en la extracción de los detalles del texto de publicaciones de Instagram, mediante técnicas de Text Mining, como NLTK's TweetTokenizer. Para poder tomar el texto de este tipo de publicaciones se realiza un procedimiento que consta de normalizar el texto eliminando espacios en blanco, emojis, numerales, etc. Después se utilizan incrustaciones de palabras entrenadas para poder encontrar alguna relación del texto con la moda.

A partir de los datos obtenidos del texto, se realiza una clasificación en varios grupos como marcas, artículos, patrones, materiales y estilos. En este sistema, se compara el rendimiento del modelo según dos conjuntos de datos: el primero es un dataset público llamado "Deep Fashion" y el segundo, es un conjunto de datos compuesto por publicaciones de Instagram de una comunidad de modelos de moda (RewardStyle). Los algoritmos empleados fueron dos arquitecturas CNN: VGG16 y ResNet. Los mejores resultados se obtuvieron con ResNet en la predicción de categorías, con un porcentaje del 35% para la precisión en las primeras cinco categorías.

Justificación

En Latinoamérica, Ecuador es uno de los países que está en el top de líderes en emprendimiento según el índice de Actividad Emprendedora Temprana (TEA) y una de las actividades económicas más comunes, es la venta de ropa. Este proceso, a raíz de la pandemia de Covid19, incrementó su acogida en internet, específicamente, en redes sociales (V. Lasio). Sin embargo, entre la gran cantidad de prendas de vestir que se promocionan, suele ser abrumador para el cliente, decidirse por un conjunto, sin antes tener que revisar toda la galería de la tienda.

Para ello, se propone la utilización de algoritmos de Deep Learning, para la creación de un sistema inteligente que, en base a un conjunto de datos establecidos, sea capaz de

recomendar prendas de vestir que se combinen. Adicionalmente, para esta investigación se ha conformado un conjunto de datos obtenidos bajo diferentes etiquetas que estará disponible para el acceso público.

Alcance

El sistema inteligente de recomendación de moda basado en imágenes utilizando Inteligencia Artificial, específicamente Deep Learning (Aprendizaje Profundo), tiene la capacidad de devolver los artículos compatibles dado un artículo de consulta de interés. Dicho sistema puede ser utilizado por las personas naturales y también, por tiendas del comercio electrónico dedicadas a la venta de ropa. Cabe considerar que el módulo basado en inteligencia artificial fue entrenado con una variedad de imágenes que incluyen prendas y conjuntos de ropa de varias marcas como H&M, Zara, entre otras, y también, redes sociales.

Objetivos

Objetivo General

Diseñar e implementar un sistema inteligente de recomendación de moda basado en Deep Learning usando como guía la metodología CRISP-DM.

Objetivos Específicos

- Analizar y seleccionar las técnicas más apropiadas para el diseño de un modelo inteligente que considere un número reducido de parámetros, combine la reducción de memoria y la latencia de ejecución, conservando una alta precisión.
- Analizar y seleccionar un conjunto de datos apropiado para la implementación de un sistema inteligente de recomendación de moda.
- Implementar y desplegar un sistema inteligente de recomendación de moda basado en Deep Learning.

Capítulo II: Marco Teórico

Deep Learning en Sistemas de Recomendación

Los sistemas de recomendación se definen como herramientas de software que tienen como objetivo sugerir productos o servicios a los usuarios, según sus preferencias (Burke, 2002). Con la gran cantidad de información que una persona se encuentra al navegar por la web, los sistemas de recomendación logran que el usuario tenga una experiencia más personalizada y vea contenido que le interese. Estos sistemas pueden ser agrupados en: basados en la popularidad, filtro colaborativo, filtro basado en contenido e híbrido. Los sistemas basados en popularidad no representan una gran personalización ya que solo clasifican el contenido según lo más visto por todos los usuarios, sin importar las preferencias de cada uno.

Los filtros colaborativos se basan en los perfiles de usuarios que compartan características en común, para recomendar ítems a los usuarios que repitan dichos patrones. Por ejemplo, en un grupo de personas con un perfil similar entre todos, se agrega una persona desconocida; si al estudiar sus características, coinciden con las del grupo, se le recomendará contenido que es visto por los integrantes del grupo. La desventaja de este modelo es que se basa en las valoraciones que los usuarios dan a los ítems (Burke, 2002) por lo tanto, si las personas no interactúan con las puntuaciones de los ítems, se tendrán pocos datos para las recomendaciones.

Por otro lado, el filtro basado en contenido se basa en la información de los ítems que se presentan a los usuarios. Es decir, se evalúan las características de los elementos vistos por el usuario y se los relaciona con otros que tengan similitud a ellos. Por ejemplo, si un usuario revisa contenido de películas, pero el género que más ve, son las películas de terror, entonces se harán recomendaciones de películas que sean de este género. Finalmente los modelos híbridos combinan lo mejor de los modelos antes mencionados y realizan recomendaciones basándose en el contenido con el que más interactúa la persona y en las características que comparten dichos ítems.

Adicionalmente a los modelos descritos, se han introducido también técnicas de inteligencia artificial para facilitar la creación de sistemas de recomendación. En (Zhang, Lu, & Jin, 2021), se mencionan los sistemas de recomendación basados en: perceptrón multicapa, donde se utiliza el filtrado colaborativo neural (NCF) (He, y otros, 2017) para estudiar la relación no lineal entre el usuario y el ítem; en autoenconders, utilizado para generar representaciones de características de los artículos en base a su contenido; en CNN, que extraen información de artículos en base a las reseñas de los usuarios y hashtags que utilizan; en RNN, que permiten estudiar la evolución de los intereses del usuario como las características de los artículos; en redes generativas adversariales (GAN), utilizadas para adquirir conocimiento en base a usuario/artículo, grafos de conocimiento, etiquetas e imágenes (Tang, y otros, 2019)

Modelos de Deep Learning

Redes Neuronales Convolucionales (CNN)

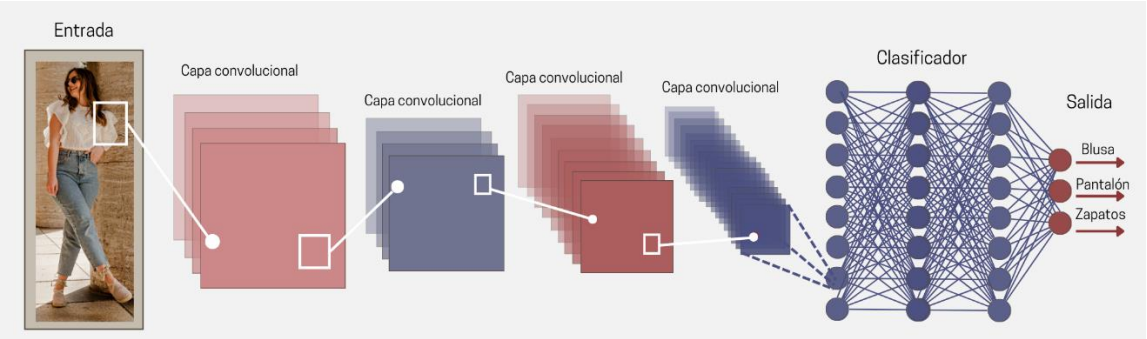
Son redes neuronales especializadas que facilitan la clasificación y reconocimiento de imágenes (Yoshua, Goodfellow, & Aaron , 2016) obteniendo matrices como entrada. Su eficacia radica en que, mediante el reconocimiento de patrones, se determina la clasificación de un objeto. Están compuestas por capas de “convoluciones” y “reducción”. Las capas de convolución son las encargadas de entrenar diferentes filtros, los cuales, van a extraer las características más importantes de la imagen de entrada, identificando así, líneas, colores, texturas, entre otros, hasta llegar a un grado de abstracción más alto, reconociendo si la imagen ingresada pertenece a un elemento en específico.

Por otra parte, las capas de reducción, como su nombre lo indica, permite reducir la cantidad de información que va a pasar por cada filtro, extrayendo los datos más importantes. Estas capas se repiten alternamente cuantas veces sea necesario y al final, el resultado de todo este proceso se convierte en un vector que pasa a una etapa de conexión, donde se utiliza

una red neuronal tradicional y allí se analizan las características extraídas, para determinar la clasificación de la imagen. La arquitectura de esta red se muestra en la figura1

Figura 1

Arquitectura de la red CNN



Nota: La figura presenta la arquitectura de una red neuronal convolucional

Redes Neuronales Recurrentes (RNN)

En (Yoshua, Goodfellow, & Aaron , 2016) se definen como redes neuronales utilizadas para el procesamiento de datos secuenciales. Esto significa que es importante el orden en que suceden los datos para obtener un resultado con sentido, por ejemplo, al decir una oración. Estas redes neuronales trabajan con un estado oculto, donde se guarda la última palabra que se recorrió, pero esto genera un nuevo problema ya que la última palabra de la secuencia es la que tendrá más peso, por lo tanto, si existe un cambio en una oración con un mismo significado, no se podrá predecir correctamente. Como solución, se presentan las Memorias a corto y largo plazo (LSTM).

Ruihui Mu (Mu, 2018) define a las LSTM, como memorias a corto plazo que se utilizan para procesar y predecir acontecimientos importantes, en intervalos variables de tiempo. Estas redes LSTM agregan un procesador al algoritmo que define si la información es útil. La forma de resolver el problema anteriormente planteado, lo hacen mediante un control de puertas (entrada, olvido y salida), donde se verifica la importancia del dato en cuestión según el algoritmo y si no cumple la certificación de este, el dato es abandonado en la puerta del olvido.

Una arquitectura más que trabaja dentro del campo de las RNN, para resolver los problemas de las gradientes en los datos secuenciales, son las Unidades Recurrentes Controladas (GRU) (Pedamallu, 2020). Al igual que las redes LSTM, las GRU también agregan puertas para la validación de información, pero en lugar de la puerta de “olvido'', se usan las puertas de “actualización y restablecimiento". La puerta de actualización es la que define la cantidad de información que se le debe comunicar al siguiente paro, mientras que las de restablecimiento, definen qué información es importante y cuál puede ser omitida.

En los tipos de redes recurrentes mencionados, la predicción de la secuencia se hace solo de manera causal, es decir, que únicamente se mira hacia el pasado, para predecir los datos futuros. Sin embargo, existen situaciones donde es necesario tomar datos del futuro para hacer predicciones correctamente. En el caso del lenguaje natural, con idiomas que tienen un formato SOV (Sujeto, objeto directo, verbo), es necesario revisar lo que se dice al final, para saber el contenido de la oración. La funcionalidad de estos sistemas se puede observar en los traductores, donde se debe tomar cada palabra hasta el final de la frase, para encontrar un sentido en lo que se está traduciendo.

Para ello, las Redes Neuronales Recurrentes Bidireccionales (BRNN) (2016) combinan una RNN que recorre la secuencia desde el inicio y otra RNN que recorre la secuencia desde el final. El concepto básico de las BRNN, es realizar secuencias de entrenamiento hacia atrás y hacia delante, en dos redes recurrentes separadas, pero conectadas a la misma capa de salida (Pierre , Søren, Frasconi, Soda, & Pollastri) De esta manera, en cada punto de secuencia, nuestra BRNN contiene información completa y secuencial de los puntos que están delante y detrás.

Al trabajar con imágenes, el mejor enfoque para desarrollar un modelo, es la arquitectura CNN. Mediante varios filtros, se extraen las características más relevantes de la imagen y se identifican los objetos que hay en ella. Por otro lado, las LSTM son la mejor opción para tratar datos que se caracterizan como una serie temporal (Xavier, 2019) Si bien este

proceso se aplica para imágenes, existen ciertos elementos como por ejemplo, videos, películas, etc., donde se toma importancia la secuencia en la que estas suceden.

Por lo tanto, para realizar una arquitectura más eficiente para las secuencias de imágenes, se combinan ambas estructuras, teniendo como resultado una red LSTM Convolucional (ConvLSTM). Esta red, es una RNN utilizada para la predicción de espacio-tiempo y contiene estructuras convolucionales en las transiciones de estados (Shi, y otros, 2015). La ConvLSTM predice el estado de una celda específica, mediante las entradas y los estados anteriores de sus vecinos.

Metodologías Orientadas a Data Mining

Las metodologías definen los pasos que se deben seguir para llevar a cabo un trabajo y además describe la manera de hacerlo. Cada metodología sigue su propio proceso para el descubrimiento de información, pero existen algunas similitudes entre ellas y entre el modelo genérico que se compone de cinco etapas básicamente. En la primera etapa se hace la selección y fusión de los datos. En la segunda, se tratan los datos para limpiarlos y transformarlos, puesto que, en la tercera etapa, vienen las características, que deben ser de carácter numérico para que sea entendible al modelo, el cual, se encuentra en la cuarta etapa. Finalmente, se tienen los resultados y respuestas al problema inicial en la quinta etapa (Moine, Gordillo, & Haedo, 2011)

Metodología KDD

Sus siglas hacen referencia al “Descubrimiento del conocimiento (Knowledge Discovery in Databases)” y fue el primer modelo aceptado en la comunidad científica en el año 1996. Es el encargado de preparar los datos e interpretar los resultados que dan sentido a los patrones obtenidos. Su objetivo principal se basa en procesar automáticamente volúmenes grandes de datos, para satisfacer las metas del usuario y encontrar conocimiento útil. Esta metodología está compuesta por varias fases iterativas que se resumen en cinco y parten desde la

exploración y entendimiento del problema de negocio. Se proponía que la minería de datos era una de las etapas que complementaban al proceso, pero actualmente, los términos “minería de datos” y “KDD” se utilizan para describir todo el procedimiento para la extracción de datos (Moine & Haedo, 2015)

Metodología SEMMA

Según Fischer (2012), es definida como el “proceso de selección, exploración y modelado de grandes volúmenes de datos para descubrir patrones de negocio desconocidos”. Sus siglas hacen referencia a cinco pilares del proceso: Sample, explore, modify, model y Assess. Esta metodología está enfocada principalmente en los aspectos netamente técnicos, es decir, que excluyen las actividades relacionadas con el análisis y comprensión del problema de negocio. Sus herramientas son llamadas “nodos” y se basan en distintas etapas que componen esta metodología, de manera que, el software ofrece herramientas para cada una de las fases y estas pueden ser usadas también en otras metodologías (Giraldo Mejia & o Jiménez Builes , 2013)

Metodología CRISP-DM

Fue desarrollado en el año 2000 por las empresas SPSS, NCR y Daminer Chrysler y propone seis fases en su metodología que dictan de la siguiente manera: comprensión del negocio, comprensión de datos, preparación de datos, modelado, evaluación y finalmente, la implementación. Cada una de estas fases están compuestas por subtareas generales. Es una de las principales metodologías que siguen los analistas en inteligencia de negocios, al tener un esquema donde las fases se interrelacionan entre sí durante todo el proceso, consiguiendo un modelo iterativo. Por otro lado, una de sus características destacables, es que proporciona dos documentos; el primero describe de manera general cada una de las fases, las tareas y las salidas del proyecto, mientras que el segundo, ofrece información más detalla para la aplicación práctica de este modelo (Moine, Gordillo, & Haedo, 2011)

Para sintetizar la información de cada una de las metodologías descritas, en la Tabla 1 se presenta un cuadro comparativo de sus fases y características principales.

Tabla 1

Comparación de las metodologías orientadas a Data Mining

Metodología	Fases	Características
KDD	Selección Preprocesamiento/limpieza. Transformación/reducción. Minería de datos. Interpretación/evaluación	Procesa automáticamente volúmenes grandes de datos, para satisfacer las metas del usuario y encontrar conocimiento útil.
SEMMA	Muestrear Explorar Modificar Modelar Evaluar	Está enfocada en aspectos técnicos, es decir, que excluyen las actividades relacionadas con el análisis y comprensión del problema de negocio
CRISP-DM	Comprensión del negocio Comprensión de datos Preparación de datos Modelado Evaluación Implementación.	Tiene un esquema donde las fases se interrelacionan entre sí durante todo el proceso, consiguiendo un modelo iterativo

Nota: Esta tabla describe las fases de cada una de las metodologías orientadas a Data Mining y su principal característica. Fuentes: (Moine & Haedo, 2015) (Moine, Gordillo, & Haedo, 2011) (Giraldo Mejia & o Jiménez Builes , 2013)

De las metodologías estudiadas, se puede determinar que, aunque representan varios procesos diferentes, las fases de cada una de ellas, tienen similitudes entre sí. Algunos de

estos modelos están más orientados al proceso de manera general y no proponen tareas específicas, como, por ejemplo, KDD y SEMMA, que dejan a criterio del equipo, la manera de desarrollar cada una de las subtareas de cada fase. Por otro lado, está la metodología CRISP-DM que proporciona a nivel de detalle las tareas de cada fase, puesto que incluye actividades para llevar a cabo la gestión del proyecto. Según un estudio realizado por KDnuggets, la metodología más utilizada es la del modelo CRISP-DM, puesto que toma en cuenta las aplicaciones realizadas en torno al negocio.

Aprendizaje no supervisado

El aprendizaje no supervisado es una forma en la que un modelo aprende datos, es decir, que dichos datos no están etiquetados y por lo tanto, el algoritmo debe intentar entenderlos por sí mismo. Los resultados que se obtienen mediante algoritmos de aprendizaje no supervisado, se basan en similitudes y patrones que se repiten en los datos (Aldo, y otros, 2021). Este tipo de aprendizaje se utiliza para explorar conjuntos de datos desconocidos y es considerado un ejemplo real de inteligencia artificial, ya que funciona bastante similar a la manera en la que los humanos aprendemos. La desventaja principal de los algoritmos de aprendizaje no supervisado, es que el resultado puede ser menos preciso, debido a que no se conoce, previamente, la salida de datos exactos que se va a obtener.

El aprendizaje no supervisado emplea diferentes algoritmos para realizar el ajuste de datos en grupos o clústeres. Los clústeres son la forma más común en la que se implementa aprendizaje no supervisado para obtener una estructura general de los datos, separándolos en subgrupos, según las similitudes que estos posean. Así mismo, existe otro método en el aprendizaje no supervisado, llamado “asociación”. Este término hace referencia a la relación que tienen los datos, sin necesidad que presenten las mismas características (Aldo, y otros, 2021). Por ejemplo, en una tienda online, se asocian los artículos que compran los usuarios y aparece como recomendaciones para otros clientes, que buscaron los mismos productos.

Autocodificadores

En (Zhu, Wang, Bai, Yao, & Bai, 2020) se definen como un tipo específico de redes neuronales diseñadas especialmente para codificar una entrada, de forma comprimida y representativa, para luego decodificarla y reconstruirla lo más parecido posible al dato de entrada. La primera vez que se introdujo este tipo de neuronas, fue en (Rumelhart , Hinton , & Williams , 1985), donde se entrenó a una red neuronal para que reconstruyera su entrada. El objetivo principal de los autocodificadores es el aprendizaje no supervisado de representaciones “informativas” que se pueden utilizar para distintas implicaciones como la agrupación.

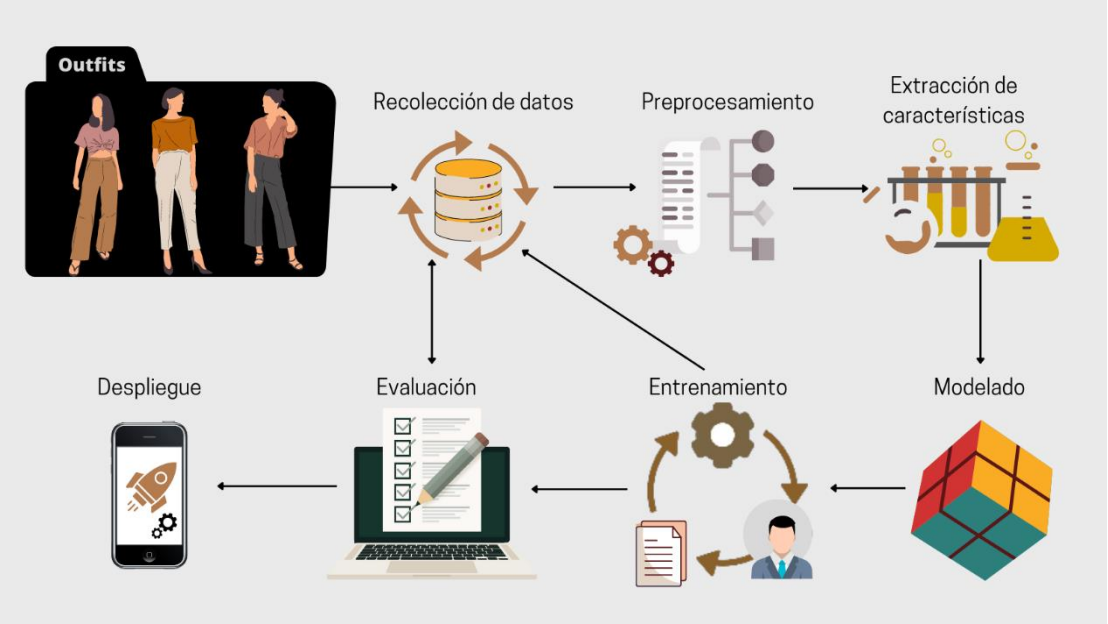
Los autocodificadores suelen presentar una estructura simétrica, de forma que, mediante capas ocultas, la salida de la red, sea bastante similar a la entrada. Los autocodificadores más comunes son: codificadores regularizados, convolucionales, multicapa y autocodificadores regulares o planos. Los codificadores regularizados usan una determinada función de pérdida con la que la red neuronal puede llevar a cabo, funciones más complejas; los codificadores convolucionales, como su nombre lo indica, utilizan capas convolucionales y totalmente conectadas; los codificadores multicapa, son los que emplean redes profundas con varias capas ocultas, mientras que los autocodificadores planos, presentan una sola capa oculta.

Capítulo III: Metodología

En esta sección se especifican los detalles acerca de cada módulo de los componentes como la recolección de datos, preprocesamiento, extracción de características, modelado, entrenamiento, evaluación y despliegue, que se muestran en el framework propuesto.

Figura 2

Framework para la elaboración del modelo de IA



Nota: La figura presenta el framework para la elaboración del modelo de IA y se basa en la metodología CRISP-DM.

Recolección de datos

Para esta investigación se ha creado un dataset con imágenes de outfits obtenidos de diferentes marcas de ropa y redes sociales, con la utilización de web scraping en Python. De esta manera, mediante el algoritmo empleado, se ingresa al buscador de Google, según el enlace que se le haya especificado en el código y se recorre cada imagen, obteniendo la URL de la misma para luego descargarla. Para que el código funcione correctamente, se debe instalar un controlador de Chrome y el navegador de Google.

La librería con la que funciona principalmente el código es “Selenium”, ya que con esta herramienta, se facilita la ejecución de pruebas automatizadas. El algoritmo contiene dos funciones bases que consisten en la obtención de la URL de cada imagen y la descarga de estas. En la primera función, se especifica el enlace de la página de donde se obtendrán las imágenes, necesariamente de Google, se pasa por parámetro el máximo de imágenes que se buscarán y se establece un bucle que recorra dicho array para abrir imagen por imagen.

Al tener acceso a cada imagen, se obtiene la información de la misma y por tanto, su URL. Luego, en la función de descarga, se toman las direcciones de las imágenes devueltas por la función anterior y se establece el formato de descarga, el nombre del archivo y la carpeta en la que se guardarán las imágenes. Mediante la técnica de web scraping, se obtuvo un conjunto 11112 imágenes. Para completar una cantidad considerable de datos, se obtuvieron también 9112 imágenes del dataset “Deep Fashion”, teniendo un conjunto de datos de 20224 imágenes.

Preprocesamiento de datos

Luego de obtener una cantidad considerable de imágenes, se procedió con la limpieza de estas. Como primera parte, se debían eliminar las que no involucraban conjuntos de ropa. Para este proceso se hizo una selección manual de los datos que si nos servían y se apartaron en un directorio diferente, dando como resultado un total de 19635 imágenes. Luego, se eliminaron las imágenes que no cumplían con el tamaño requerido para el conjunto de datos. En este algoritmo se utilizaron las librerías de “Os” y “CV2”. En el código, el primer paso a realizar es la especificación de la ruta de las imágenes, luego, se procede a leer todas las imágenes contenidas y por último, todas se las pasa por una condición, donde se eliminan las imágenes con un tamaño menor a 300 pixeles de ancho, reduciendo así el conjunto a 11112 imágenes.

Posteriormente, se redimensionó el tamaño de las imágenes, ya que todas deben tener una misma medida para que sea entendible al modelo. En esta parte se aplicó la lectura de

todos los archivos contenidos en la carpeta de imágenes. Todas las imágenes pasan por una función donde se especifica, por parámetro, el ancho y el largo a las que serán redimensionadas, siendo así, la unificación de estas a 300x350 píxeles. Por medio de la librería “CV2” aplicamos la función “resize” y enviamos las dimensiones de la nueva imagen. Por último, estas imágenes vuelven a ser guardadas con el mismo nombre y en el mismo directorio.

En el siguiente paso se procedió con la eliminación del ruido contenido en las imágenes. Este proceso se realiza para desechar cualquier desperfecto encontrado en las imágenes. Mediante la librería de “CV2” realizamos el procesamiento y eliminación de ruido en las imágenes y las guardamos en una nueva carpeta. Finalmente, era necesario también quitar el fondo de las imágenes. Para ello, se utilizó la librería “rembg”, la cual, permite utilizar la función “remove”; está se encarga de quitar el fondo de las imágenes, siempre que detecte un objeto que esté definido claramente del fondo. Una vez eliminado el fondo de las imágenes, estas fueron guardadas en otra carpeta con extensión png. Luego de todos estos filtros, el conjunto de datos final se mantuvo con 11112 imágenes.

Ya con la limpieza de datos realizada, se debe segmentar las imágenes, de manera que se separe cada prenda en los conjuntos de ropa. Para este procedimiento, se deben utilizar las anotaciones de las imágenes, donde se establecen los puntos de corte, en x e y, de cada prenda, y dicha información, se almacena en archivos XML. En el código, se debe especificar la ruta de la carpeta de entrada de las imágenes, como la de salida. Si en la ruta de la carpeta de salida, no se encuentra el nombre del directorio, el código crea la carpeta automáticamente. Como siguiente paso, se leen los archivos XML de cada imagen y se detectan las etiquetas que contienen, como el nombre del archivo, el alto, el ancho, el nombre de la clase y sus posiciones en x e y. Ya con esta lista de datos, se procede a realizar el corte y separación de las prendas de vestir y cada conjunto de ropa, se guarda en carpetas diferentes.

Extracción de características

Posteriormente, los datos recopilados deben mapearse en un esquema común en el paso de preprocesamiento. La ingeniería de características hace referencia a la extracción y selección de características adecuadas y fundamentales en el aprendizaje automático, que a su vez, son primordiales en la definición y enriquecimiento de los predictores. Sin embargo, en la ejecución, la definición de características suele representar un desafío que requiere trabajo, tiempo y conocimiento de expertos, así como de ingenio. Adicionalmente, las técnicas tradicionales de ingeniería de características, utilizando datos reales como videos, imágenes, clips de sonidos y datos de sensores, no son la mejor opción para la producción de características específicas.

Por tal motivo, en este estudio se utiliza el aprendizaje no supervisado para extraer y reconocer factores explicativos que se ocultan en los datos. Es decir, mediante varias capas convolucionales recurrentes, podemos obtener información facilitada por las representaciones de aprendizaje de los datos tanto de clasificadores, como otros predictores. Por otro lado, en cuanto a los modelos probabilísticos, generalmente retienen la distribución consecutiva de los componentes explicativos subyacentes del ingreso observado.

Modelado

En nuestro sistema de recomendación de moda, cada conjunto de ropa representa una clase, de manera que la salida del modelo está contemplada entre diversas opciones. El dato de entrada dado por el usuario, será comparado con las imágenes similares de nuestro conjunto de datos y presentará los complementos del artículo de consulta. Partiendo de esta idea, se ha utilizado, principalmente, aprendizaje no supervisado para la extracción de características, en un autocodificador, que transforma las imágenes de entrada, a imágenes de salidas muy similares.

Luego de obtener las características destiladas con el autocodificador compuesto de varias capas ConvLSTM, que permiten manejar datos espaciotemporales y capas

Normalizadoras (Kiros, Hinton, & Lei Ba, 2016), empleadas para reducir el tiempo de entrenamiento, se transformó la salida de este modelo, mediante la función “Flatten()”, convirtiendo los datos a una matriz unidimensional. Posteriormente, utilizamos aprendizaje supervisado para crear la capa de clasificación, conformada por tres capas densas que emplean las funciones de activación “relu” y “softmax”. La salida de dicho clasificador identifica las partes del conjunto de ropa y se obtiene su respectiva etiqueta.

Entrenamiento y evaluación

En esta sección, se separan los datos en entrenamiento (80%) y testeo (20%). Posteriormente, se compila el modelo utilizando la función de coste “Entropía cruzada categórica”, puesto que el modelo presenta más de dos salidas en su clasificación. Se utiliza también, el optimizador “Adam”, por ser el mejor método y más eficiente para la optimización del modelo. Para la evaluación del modelo, se especifica la métrica “accuracy” para verificar la exactitud de predicción del modelo.

Al momento de entrenar el modelo, se utiliza únicamente ese 80% de datos definidos previamente, con un “batch_size” igual a 5, el cual representa la cantidad de datos que ingresarán a la neurona en cada pasada. Se especifica también, que la red neuronal completará 20 ciclos, mediante la variable “epoch” y finalmente, para mostrar la barra de progreso, se establece la variable “verbose” en 1. Para evaluar la exactitud de predicción de este modelo, se utiliza la función “evaluate” y luego se manda a buscar los casos en los que el modelo ha predicho incorrectamente las clases.

Despliegue

Para el despliegue del modelo se utilizó el framework Flask, escrito en Python (Flask Documentation, 2022), para el cual, se tuvo que crear un ambiente de desarrollo en virtual studio code, instalar Flask y luego las librerías que este framework utiliza. La ventaja de

trabajar con Flask, es que la sintaxis que utiliza para la programación, es la misma que Python y por lo tanto, las funciones utilizadas no fueron tan complejas de implementar.

Primeramente, se creó un archivo Python que tiene todas las librerías necesarias para el funcionamiento de la aplicación (véase la figura 3). Entre las librerías, se encuentran tensor Flow para la carga y ejecución del modelo; Flask para la creación de la aplicación web; entre otras necesarias para el procesamiento de los datos.

Figura 3

Librerías requeridas para la implementación de Flask

app.py

```python
#Import necessary libraries
from flask import Flask, render_template, request
import pandas as pd
import numpy as np
import tensorflow as tf
import cv2

from keras.utils import load_img
from keras.utils import img_to_array
from keras.models import load_model

from PIL import Image
from functools import reduce
import os
```

Nota: En esta figura se presentan las librerías necesarias para el ambiente de desarrollo en el framework Flask.

Posteriormente, se establece la función de lectura de la imagen que el usuario subirá a la aplicación (véase la figura 4). En esta función, se redimensionará y se extraerá todos los pixeles de las imágenes. Así también, se realiza una normalización para establecer el rango de números entre 0 y 1.

Figura 4

Función para la lectura de imagen

```python
def read_image(filename):
    # Load the image
    img = load_img(filename, target_size=(224, 224,3))
    # Convert the image to array
    img = img_to_array(img)
    img = img.astype('float32')
    img = img / 255.0
    img=np.concatenate((img,data,data),axis=1)
    # Reshape the image into a sample of 1 channel
    img = img.reshape(1,1, 224, 672, 3)
    # Prepare it as pixel data
    return img
```

Nota: En esta figura se presenta la función establecida para leer la imagen que el usuario sube a la página web.

Luego, se establece la función principal, en la cual, se define la ruta del proyecto: el método Get y Post. En esta función, se cargará el modelo de inteligencia artificial y se envía la imagen al modelo (véase la figura 5). El siguiente paso es recolectar la predicción realizada por el modelo.

Figura 5

Función principal para la ejecución del modelo

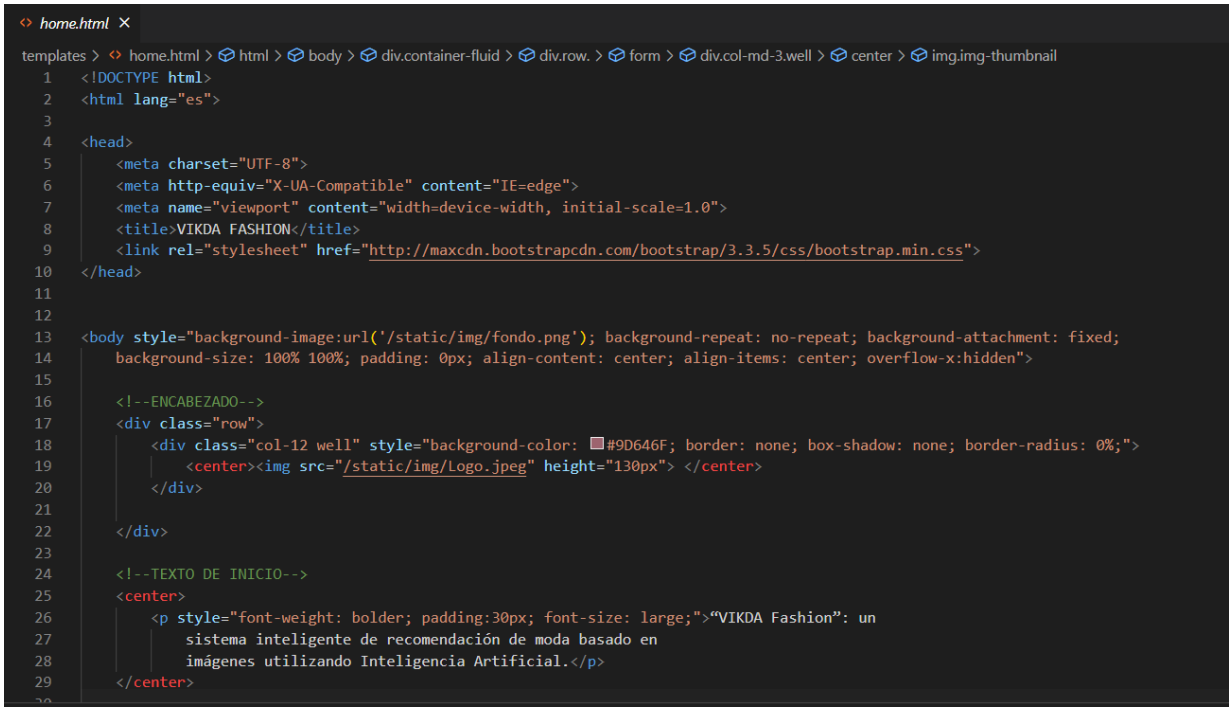
```python
@app.route("/predict", methods = ['GET','POST'])
def predict():
    if request.method == 'POST':
        file = request.files['file']
        if file and allowed_file(file.filename):
                filename = file.filename
                file_path = os.path.join('static/images', 'filename')
                file.save(file_path)
                img = read_image(file_path)
                # Predict the class of an image
                with graph.as_default():
                    model1 = load_model('EncoderClassifier.h5')
                    outfitpred = model1.predict(img)
                    predicted_classes = np.argmax(np.round(outfitpred),axis=1)
                    #predicted_classes = int(np.argmax(outfitpred))
                    print(outfitpred)
                    print(predicted_classes)
                    outfitRecu = os.path.join('/static/ImagesParaLaY', "Outfit"+str(int(predicted_classes))+".jpg")
                    print(outfitRecu)
                return render_template('home.html', user_image = file_path,pred_image=outfitRecu)
    return render_template('home.html')
```

Nota: En esta figura se presenta la función principal para la ejecución del modelo

Finalmente, se crea el archivo “home.html”, el cual será la página principal que mostrará la interfaz gráfica. Este archivo será llamado por la función creada en la figura 6, y se enviaran algunos valores a ser mostrados en esta página.

Figura 6

Código HTML de la página inicial del proyecto



```
home.html
templates > home.html > html > body > div.container-fluid > div.row. > form > div.col-md-3.well > center > img.img-thumbnail
<!DOCTYPE html>
<html lang="es">

<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>VIKDA FASHION</title>
    <link rel="stylesheet" href="http://maxcdn.bootstrapcdn.com/bootstrap/3.3.5/css/bootstrap.min.css">
</head>


<body style="background-image:url('/static/img/fondo.png'); background-repeat: no-repeat; background-attachment: fixed;
    background-size: 100% 100%; padding: 0px; align-content: center; align-items: center; overflow-x:hidden">

    <!--ENCABEZADO-->
    <div class="row">
        <div class="col-12 well" style="background-color: #9D646F; border: none; box-shadow: none; border-radius: 0%;">
            <center><img src="/static/img/Logo.jpeg" height="130px"> </center>
        </div>

    </div>

    <!--TEXTO DE INICIO-->
    <center>
        <p style="font-weight: bolder; padding:30px; font-size: large;">“VIKDA Fashion”: un
            sistema inteligente de recomendación de moda basado en
            imágenes utilizando Inteligencia Artificial.</p>
    </center>
```

Nota: En esta figura se presenta el código HTML de la página principal donde se presenta la funcionalidad del modelo.

En la figura 7, se muestra el despliegue de la interfaz gráfica en la que funciona el modelo. El usuario sube una foto de su galería y al presionar el botón de “buscar complementos”, se muestra el resultado en una imagen, en la parte derecha de la pantalla. Lo que hace el modelo, es buscar un conjunto de ropa que contenga una prenda similar a la que ha especificado el usuario y presentarla. Adicionalmente, se ha agregado un botón de información, donde se da instrucciones al usuario de cómo funciona la página.

Figura 7

Interfaz gráfica de la aplicación web

Nota: En esta figura se presenta la interfaz gráfica de la aplicación web (vista de escritorio).

Capítulo IV: Experimentación

Configuración experimental y preprocesamiento de datos

Los experimentos fueron realizados en dos escenarios diferentes, según la complejidad de procesamiento. Para la recolección y preprocesamiento de datos, por ejemplo, se utilizó una máquina Windows 10 con 8 GB de RAM, mientras que para el entrenamiento del modelo, que requiere de más memoria, se compró instancias en Google Colab, que ofrecen 82 GB de RAM y GPU de 32 GB. Todos los procedimientos y modelos, fueron implementados en Python 3.8.10. Para la recolección de los datos se utilizó la técnica de web scraping, obteniendo así diferentes imágenes de conjuntos de ropa y también se adjuntaron imágenes del dataset “Deep Fashion” a nuestro conjunto de datos.

Posteriormente, se procedió con la limpieza de estos, quitando el fondo y ruido de las imágenes. Para la segmentación de los datos, se utilizó la aplicación “Roboflow” que permite delimitar las zonas donde se encuentran las prendas de vestir y luego se exporta un archivo XML desde la misma aplicación, que contiene las posiciones de cada artículo (blusa, pantalón y zapatos) y dicho archivo, se manda a llamar en el código Python, para realizar un corte o “crop” de las imágenes y colocarlas es archivos diferentes según el conjunto de ropa al que pertenecen. Luego de tener cada conjunto de ropa en una carpeta diferente, dentro del mismo conjunto de datos, se realiza la permutación de los mismos, para facilitar el aprendizaje al modelo.

Extracción de características

El conjunto de datos está conformado por más de 10 mil imágenes y cada imagen, representa una salida diferente, razón por la cual, es complicado manejar este tipo de problemas con aprendizaje supervisado. Por otra parte, el aprendizaje no supervisado se especializa en descubrir las características de los conjuntos de datos desconocidos. Así pues,

con la implementación de un autocodificador, que contiene únicamente su codificador, se destilaron las características de nuestro conjunto de datos.

Por su parte, el codificador recibe como entrada las imágenes a color (RGB) de tamaño 224x224 transformadas en píxeles. Cada fila representa un conjunto de ropa y cada columna representa las diferentes prendas de dicho conjunto (blusa, pantalón y zapatos). Por lo tanto, la entrada está dada por la siguiente sintaxis: (1, 224, 672, 3) donde 1 es el número de pasos de la red; 224 es el tamaño de la imagen; 672=224x3, ya que cada conjunto de ropa está dado por 3 prendas y 3, que significa que la imagen de entrada es a color.

Posteriormente, se define una variable que almacena el valor de la primera capa dada por un TimeDistributed(Conv2D) con 128 convoluciones, un kernel de tamaño (11,11) , 4 pasos de convolución en largo y ancho representado como “strides”, el padding “same”, con el que se rellena de ceros uniformemente de izquierda/derecha y abajo/arriba y por último, el “batch_input_shape” (1, 224, 672, 3), con lo que se define que la red neuronal acepta entradas con tamaño fijo definido, de manera que se evita la creación de vectores con diversas dimensiones.

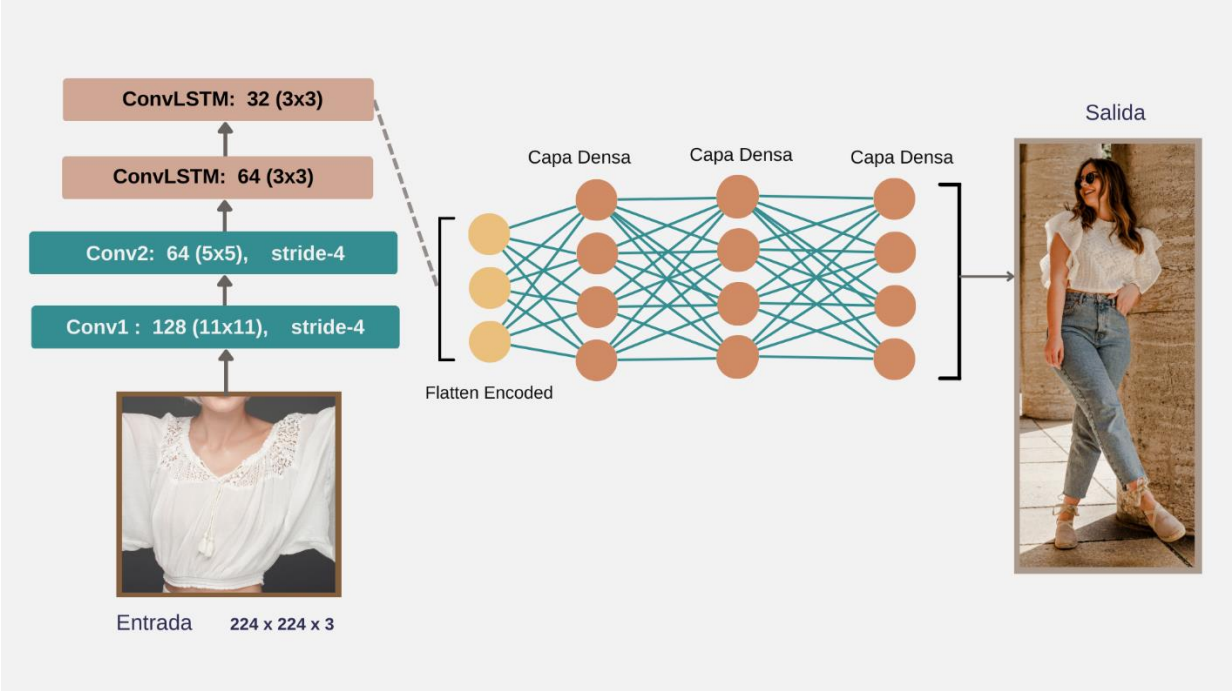
Después de la capa convolucional, se agrega una capa de normalización y se la guarda en la misma variable del paso anterior, para realizar la secuencia de entrada. Luego de normalizar la variable de entrada, se vuelve a crear una capa TimeDistributed(Conv2D), pero con la mitad de las convoluciones, y un kernel de tamaño (5,5) para ir reduciendo la entrada de datos. Esta secuencia se repite dos veces más y luego se envía como salida del codificador.

Entrenamiento y evaluación del modelo

Luego de haber destilado las características del conjunto de datos, se puede emplear aprendizaje supervisado para realizar el clasificador de capas densas. La arquitectura del modelo se puede observar en la figura 8.

Figura 8

Arquitectura del modelo



Nota: La figura presenta la arquitectura del modelo implementado desde la entrada de datos hasta su salida.

Para las capas de clasificación, se toma como entrada la salida del codificador. Posteriormente, se utiliza la función Flatten() para transformar la matriz en un array de una sola dimensión y así pasarlo a las capas densas. La primera capa densa tiene una dimensión de 128 en su salida y la función de activación “relu”. Las funciones de activación definen cómo se dará la salida de las neuronas y por ello es importante elegirlas correctamente (Keras, 2022). En este caso, la función “Relu” es recomendada para el entrenamiento de redes neuronales profundas ya que evita los problemas de gradiente de fuga en las diversas capas.

Luego de la capa densa, se agrega el método de “dropout” para evitar el sobre ajuste, con un 0.2 de fracción de unidades de neuronas que se van a descartar aleatoriamente. Después de esta capa, se vuelve a especificar una capa densa, pero con una dimensión de 80 y la misma función de activación. Así mismo, se agrega el método “dropout” nuevamente y para

la salida, se establece una capa densa pero con activación “softmax”, que usualmente se utiliza como la capa final en clasificadores de redes neuronales profundas y permiten la transformación de la salida, en un vector de probabilidades (Mehta, 2021).

Para la compilación del modelo, se utiliza la función de coste “Categorical Cross Entropy” o “Entropía cruzada categórica”, ya que nuestro modelo presenta más de 2 resultados en la clasificación. El optimizador que se implementa es Adam. Según (Kingma & Ba, 2015), este método es eficiente computacionalmente, fácil de efectuar, no utiliza muchos requisitos de memoria y es óptimo para problemas con gran cantidad de datos y parámetros. Y también se especifica la métrica de evaluación “accuracy”, para comprobar la exactitud del modelo.

Los datos se habían dividido previamente en 20% para testeo y 80% para entrenamiento. De esta manera, el modelo se entrena con ese 80% de datos, un “batch size” de 5, que representa la cantidad de datos que se ingresan a la neurona cada vez. Se utiliza un número bajo para que no se almacenen tantos datos en memoria y el entrenamiento no tarde demasiado. Luego se define un “epoch” de 20, con el que se especifica que las redes neuronales completarán 20 ciclos y un “verbose” igual a 1, que significa que se mostrará una barra de progreso, al entrenar el modelo.

Evaluación experimental

A continuación, se evalúa y analiza el rendimiento de los cinco escenarios planteados. En la tabla 1, se muestran los modelos que se han considerado en este estudio. En primer lugar, está el modelo principal de esta investigación, donde se utiliza un codificador espacio temporal con 4 capas convolucionales recurrentes, además del clasificador con 3 capas densas. A este modelo se lo optimizó incrementando los datos mediante la técnica de “data augmentation”, alcanzando una precisión del 98%.

Tabla 2

Resumen del rendimiento de los modelos

Modelo	Precision	Recall	F1	Accuracy
1.Encoder ConvLSTM with Data Augmentation	98.00	92.00	94.00	92.00
2. Encoder ConvLSTM without Data Augmentation	78.00	80.00	78.00	80.00
3. Encoder CNN	30.00	33.00	31.00	33.00
4. Autoencoder ConvLSTM	52.00	55.00	52.00	55.00
5. Autoencoder CNN	47.00	49.00	48.00	49.00

Nota: En esta tabla se resumen los resultados de cada métrica en los diferentes escenarios evaluados. *Fuente propia.*

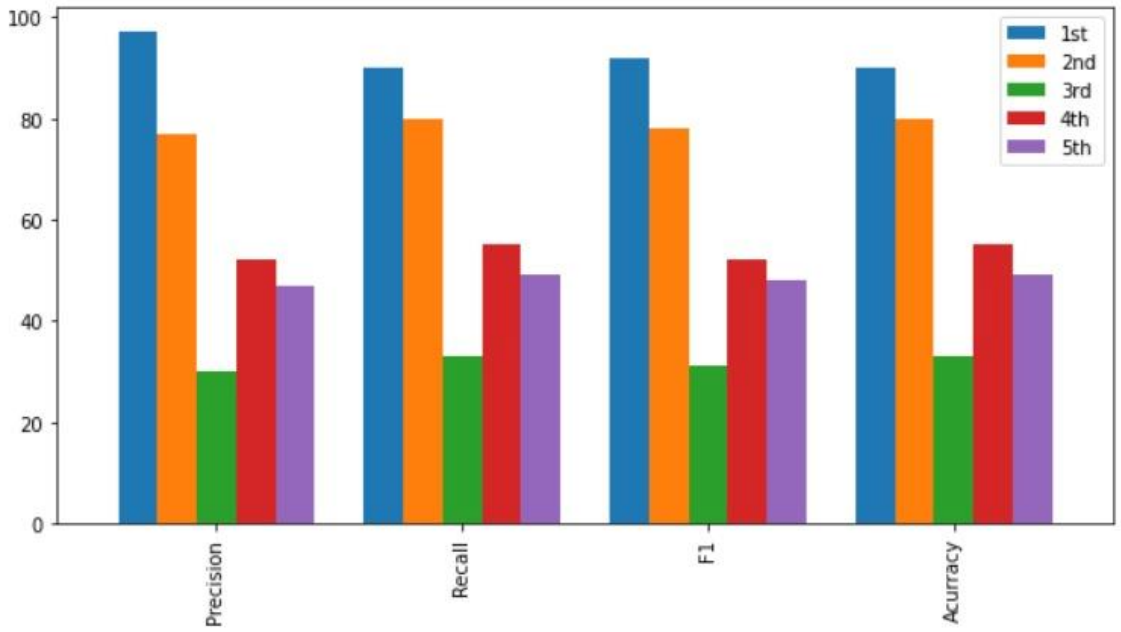
En segundo lugar se encuentra un escenario bastante similar al principal, con la diferencia de que en este, no se aumentaron datos y por lo tanto, su precisión descendió a un 78% (véase la figura 9). A diferencia del primer modelo, en este no se utilizó la técnica de data augmentation, con lo que se demuestra que implementar esta técnica, puede mejorar considerablemente el rendimiento del modelo. En el tercer escenario, no se utilizó el codificador espacio temporal, sino dos capas convolucionales normales con sus respectivas capas de “max pooling”. En este modelo se utilizó igualmente un clasificador de 3 capas densas, pero su precisión quedó en 30%.

Para el cuarto y quinto modelo, se utilizó un autocodificador completo. A diferencia de los tres escenarios descritos en los dos párrafos anteriores, aquí se utiliza tanto el codificador como el decodificador para reconstruir la imagen de entrada, en una imagen de salida muy similar. En el caso del autocodificador espacio temporal (ConvLSTM), se utilizan 4 capas

convolucionales recurrentes para la parte del codificador y así mismo, 4 capas convolucionales recurrentes para el decodificador. Igual que en los modelos anteriores, se utiliza una capa de clasificación para la salida del modelo, compuesta por 3 capas densas. A pesar de utilizar un autocodificador espacio temporal, la precisión del modelo queda en un 52%.

Figura 9

Evaluación de los modelos



Nota: Esta figura muestra un histograma de evaluación de los distintos escenarios propuestos

Para el quinto escenario, se tiene un autocodificador convolucional, donde se emplean 2 capas convolucionales para el codificador y 2 capas convolucionales para el decodificador. Luego de cada capa convolucional en el codificador, se tiene una capa de “max pooling”, mientras que para el decodificador, luego de cada capa convolucional, se tiene una capa de “UpSampling”, para volver la imagen a su tamaño de entrada. Con este modelo se obtuvo una precisión del 47%. Según las evaluaciones de cada modelo, los escenarios donde menos porcentaje se obtuvo, fue donde se utilizaba CNN y el autocodificador completo.

Capítulo V: Conclusiones y Recomendaciones

Conclusiones

En este estudio se desarrolló un sistema inteligente de recomendación de moda basado en Deep Learning, que sugiere un conjunto de ropa, a partir de una prenda de consulta de interés. Los datos que se utilizaron para este proyecto fueron obtenidos mediante la técnica de web scraping de distintas marcas de ropa como H&M, Zalando, SPRIT, entre otras; así también, redes sociales como Facebook, Pinterest y Tumblr. Adicionalmente, se agregaron imágenes del dataset “Deep Fashion.

En el modelo construido se implementó un codificador con capas Convolucionales LSTM para destilar las características y un clasificador compuesto de tres capas densas, además del método de dropout para evitar el sobre ajuste. Adicionalmente, se permutaron los datos para mejorar el aprendizaje de las redes neuronales profundas y con estas condiciones, se logró alcanzar una exactitud del 92% y una precisión del 98%. El éxito de la recomendación se fundamenta en una adecuada selección de datos y técnicas para la implementación del modelo.

Para el desarrollo de todo el modelo, se experimentaron varios entornos con diferentes condiciones. Debido a la cantidad de píxeles que se manejó por cada imagen, era complicado entrenar el modelo en una máquina con pocos recursos. Por esa razón, se utilizó un servidor de Google Colab, que ofrecía la suficiente robustez para procesar y evaluar millones de datos.

El despliegue de este modelo se lo realizó en Flask; un micro framework, que al estar escrito en Python, se facilitaron las funciones del modelo para presentarlas en una interfaz y representó un menor costo en tiempo de desarrollo. Dicha aplicación web es adaptable a dispositivos móviles ya que cuenta con un diseño responsive.

Recomendaciones

Para los problemas de inteligencia artificial, específicamente, aprendizaje profundo, se debe tener un ambiente de desarrollo robusto para realizar todas las pruebas necesarias respecto al modelo, sobre todo si se va a trabajar con imágenes, ya que estas, al transformarlas a píxeles, ocupan bastante memoria y el tiempo de entrenamiento aumenta considerablemente.

Es necesario normalizar todo el conjunto de datos para evitar la dispersión de los valores, de manera que estos sean representados entre 0 y 1. En este paso, se debe considerar también la técnica más óptima para realizar la normalización, ya que no todas las técnicas pueden ser aplicadas a todos los problemas. Todo depende del modelo que se esté implementando y la forma de los datos.

Usualmente, los modelos de inteligencia artificial tienen tendencia al sobre ajuste, por lo tanto, es importante controlar el entrenamiento, según el modelo que se esté utilizando. Por ejemplo, en el caso de las redes neuronales, se utiliza el método de “dropout” para descartar neuronas aleatoriamente y evitar el sobre entrenamiento.

Bibliografía

Kiros, J. R., Hinton, G. E., & Lei Ba, J. (2016). Layer Normalization. *arXiv*.

Aldo, G., Husic, B., Rodriguez, A., Clementi, C., Noé, F., & Laio, A. (2021). Unsupervised Learning Methods for Molecular Simulation Data. *Chemical Reviews*, 121(16), 9722-9758.

Burke, R. (2002). Hybrid recommender systems: Survey and experiments. . *User modeling and user-adapted interaction*, 331-370.

Chakraborty, S., Hoque, M., Jeem, N., Biswas, M., Bardhan, D., & Lobaton, E. (2021). Fashion Recommendation Systems, Models and Methods: A Review. *Informatics*, 49.

Dayan, P. (1999). Unsupervised Learning. *The MIT Encyclopedia of the Cognitive Sciences*.

FISCHER ANGULO, E. (2012). *Modelo para la automatización del proceso de determinación de riesgo y deserción en alumnos universitarios.* SANTIAGO DE CHILE: UNIVERSIDAD DE CHILE.

Flask Documentation. (05 de 08 de 2022). *Welcome to Flask*. Obtenido de Welcome to Flask: https://flask.palletsprojects.com/en/2.2.x/

Giraldo Mejia, J., & o Jiménez Builes , J. (2013). Caracterización del Proceso de Obtención de Conocimiento y Algunas Metodologías para Crear Proyectos de Minería de Datos. *Revista Latinoamericana de Ingeniería de Software*.

Han, X., Wu, Z., Jiang, Y. G., & Davis, L. S. (2017). Learning fashion compatibility with bidirectional lstms. *Proceedings of the 25th ACM international conference on Multimedia*, 1078-1086.

He, X., Liao, L., Zhang, H., Nie, L., Hu, X., & Chua, T. S. (2017). Neural collaborative filtering. *Proceedings of the 26th international conference on world wide web*, 173-182.

IBM. (17 de 08 de 2021). *CRISP-DM Help Overview*. Obtenido de CRISP-DM Help Overview: https://www.ibm.com/docs/en/spss-modeler/saas?topic=dm-crisp-help-overview

Jaradat, S., Dokoohaki, N., Hammar, K., Wara, U., & Matskin, M. (2018). Dynamic CNN models for fashion recommendation in Instagram. *2018 IEEE Intl Conf on Parallel & Distributed Processing with Applications, Ubiquitous Computing & Communications, Big Data & Cloud Computing, Social Computing & Networking, Sustainable Computing & Communications*, 1144-1151.

Keras. (14 de 11 de 2022). *Keras*. Obtenido de Dense layer: https://keras.io/api/layers/core_layers/dense/

Kingma, D. P., & Ba, J. (2015). Adam: A Method for Stochastic Optimization. *3rd International Conference for Learning Representations, San Diego*.

Kyprianidis , A., Kotouza, M. T., Tsarouchis, S. F., Chrysopoulos, A. C., & Mitkas, P. A. (2020). Towards fashion recommendation: an AI system for clothing data retrieval and analysis. *IFIP International Conference on Artificial Intelligence Applications and Innovations*, 433-444.

Lin, K., Yang, H. F., Liu , K., Hsiao , J., & Chen , C. (2015). Rapid clothing retrieval via deep learning of binary codes and hierarchical search. *Proceedings of the 5th ACM on International Conference on Multimedia Retrieval* , 499-502.

Mehta, K. S. (01 de 04 de 2021). *Weights & Biases*. Obtenido de How to Implement the Softmax Function in Python: https://wandb.ai/krishamehta/softmax/reports/How-to-Implement-the-Softmax-Function-in-Python--VmlldzoxOTUwNTc#:~:text=%EF%BB%BFSummary%EF%BB%BF-,What%20Is%20The%20Softmax%20Function%3F,the%20distribution%20add%20to%201.

Moine, J., & Haedo, A. (2015). Una herramienta para la evaluación y comparación de. *XXI Congreso Argentino de Ciencias de la Computación*.

Moine, J., Gordillo, S., & Haedo, A. (2011). Análisis comparativo de metodologías para la gestión de proyectos de minería de datos. *Congreso Argentino de Ciencias de la Computación*.

Mu, R. (2018). A survey of recommender systems based on deep learning. *Ieee Access,*.

Pedamallu, H. (14 de 11 de 2020). *RNN vs GRU vs LSTM*. Obtenido de Analytics Vidhya: https://medium.com/analytics-vidhya/rnn-vs-gru-vs-lstm-863b0b7b1573

Pierre , B., Søren, B., Frasconi, P., Soda, G., & Pollastri, G. (s.f.). Exploiting the past and the future in protein secondary structure prediction. *Bioinformatics*, 937-946. Obtenido de Bioinformatics.

Rumelhart , D., Hinton , G., & Williams , R. (1985). Learning internal representations by error propagation. *California Univ San Diego La Jolla Inst for Cognitive Science.*

Shankar, D., Narumanchi, S., Ananya, H. A., Kompalli , P., & Chaudhury, K. (2017). Deep learning based large scale visual recommendation and search for e-commerce. *Cornell University*.

Shi, X., Chen, Z., Wang, H., Yeung, D. Y., Wong, W. K., & Woo, W. C. (2015). Convolutional LSTM network: A machine learning approach for precipitation nowcasting. *Advances in neural information processing systems*.

Tang, J., Du, X., He, X., Yuan, F., Tian, Q., & Chua, T. S. (2019). Adversarial training towards robust multimedia recommender system. *IEEE Transactions on Knowledge and Data Engineering*, 855-867.

Turnbull, C. (26 de Julio de 2022). *SaleCycle*. Obtenido de Fashion Ecommerce: 11 Essential Online Clothes Shopping Statistics: https://www.salecycle.com/blog/featured/online-fashion-retail-11-essential-statistics/

UNCTAD. (2020). COVID-19 has changed online shopping forever, survey shows. *UNCTAD*. Obtenido de UNCTAD, “COVID-19 has changed online shopping forever, survey shows — UNCTAD,” 10 2020.

V. Lasio, A. A. (s.f.). Global Entrepreneurship Monitor Ecuador 2019/2020. *ESPAE*.

Xavier, A. (25 de 03 de 2019). *An introduction to ConvLSTM*. Obtenido de Neuronio: https://medium.com/neuronio/an-introduction-to-convlstm-55c9025563a7

Yin, R., Li, K., Lu, J., & Zhang, G. (2019). Enhancing fashion recommendation with visual compatibility relationship. *The world wide web conference* , 3434-3440.

Yong-Goo , S., Yoon-Jae , Y., Min-Cheol , S., Seo-Won, J., & Sung-Jea , K. (2019). Deep Fashion Recommendation System with Style Feature Decomposition. *IEEE 9th International Conference on Consumer Electronics (ICCE-Berlin)*, 301-305.

Yoshua, B., Goodfellow, I., & Aaron , C. (2016). *Deep Learning.* London: MIT Press.

Zhan, H., Shi, B., Chen, J., Zheng, Q., Duan, L. Y., & Kot, A. C. (2019). Fashion recommendation on street images. *2019 IEEE International Conference on Image Processing (ICIP)*, 280-284.

Zhang, Q., Lu, J., & Jin, Y. (2021). Artificial intelligence in recommender systems. *Complex & Intelligent Systems*, 439-457.

Zhu, Z., Wang, X., Bai, S., Yao, C., & Bai, X. (2020). Deep Learning Representation using Autoencoder for 3D Shape Retrieval. *arXiv preprint arXiv:2003.05991*.