CENTRO DE INVESTIGACIÓN EN

CIENCIAS DE INFORMACIÓN GEOESPACIAL, A.C.

CentroGeo

Centro Público de Investigación CONAHCYT

GEORREFERENCIACIÓN AUTOMÁTICA DE SOLICITUDES DE ATENCIÓN CIUDADANA MEDIANTE TÉCNICAS DE PROCESAMIENTO DE LENGUAJE NATURAL.

TESIS

Que para obtener el grado de

Maestro en Ciencias de Información Geoespacial

Presenta

Ulises Morales Ramírez

Director de la Tesis
Dr. Alejandro Molina Villegas

Ciudad de México 2024

CENTRO DE INVESTIGACIÓN EN
CIENCIAS DE INFORMACIÓN GEOESPACIAL, A.C
CentroGeo

Centro Público de Investigación CONAHCYT

GEORREFERENCIACIÓN AUTOMÁTICA DE SOLICITUDES DE ATENCIÓN CIUDADANA MEDIANTE TÉCNICAS DE PROCESAMIENTO DE LENGUAJE NATURAL.

TESIS

Que para obtener el grado de
Maestro en Ciencias de Información Geoespacial

Presenta

Ulises Morales Ramírez

Director de Tesis
Dr. Alejandro Molina Villegas

Sinodales
Dr. Gandhi Samuel Hernández Chan
Examinador Externo:
Dr. Edwyn Javier Aldana Bobadilla

Ciudad de México, septiembre, 2024

Resumen

La presente tesis explora la implementación de técnicas avanzadas de procesamiento de lenguaje natural (PLN) para la georreferenciación automática de solicitudes de atención ciudadana. El estudio se enfoca en el uso de modelos de clasificación de texto, específicamente Multi Layer Perceptron (MLP) y Convolutional Neural Network (CNN), para identificar y categorizar descripciones de reportes de locatel. La propuesta incluye un detallado preprocesamiento de datos y la optimización de hiperparámetros, con el fin de mejorar la precisión y generalización de los modelos. Los resultados demuestran que la combinación de técnicas de preprocesamiento y el uso de MLP proporcionan una solución robusta y efectiva para la clasificación de entidades nombradas en textos.

En la metodología, se emplearon datos obtenidos de descripciones de reportes de locatel, los cuales fueron preprocesados utilizando técnicas avanzadas de PLN. Se implementaron y compararon dos modelos de redes neuronales, evaluando su desempeño en términos de precisión, recall, y f1-score. El estudio muestra cómo el preprocesamiento y la selección de arquitecturas adecuadas pueden influir significativamente en los resultados de clasificación.

El análisis de resultados revela que el modelo MLP con la propuesta de preprocesamiento supera consistentemente a las otras configuraciones evaluadas. En particular, se observó un notable incremento en precisión y recall, lo que sugiere una mejora en la capacidad de generalización del modelo. Además, se discuten las limitaciones de los modelos CNN y se proponen futuras direcciones para la investigación, incluyendo la integración de técnicas de regularización más eficaces y la exploración de enfoques híbridos que combinen las ventajas de ambas arquitecturas.

El desarrollo de este proyecto no solo contribuye al campo del PLN y la georreferenciación, sino que también tiene aplicaciones prácticas directas en la mejora de los sistemas de atención ciudadana. La implementación de estas tecnologías puede facilitar una respuesta más rápida y precisa a las solicitudes ciudadanas, mejorando así la gestión urbana y la satisfacción de los ciudadanos.

Finalmente, se destaca la importancia de la calidad y cantidad de los datos de entrenamiento, así como de la arquitectura y los parámetros del modelo, en el desempeño de los sistemas de PLN. Los hallazgos de este estudio proporcionan una base sólida para futuras investigaciones y desarrollos en el área de la georreferenciación automática y el procesamiento de lenguaje natural.

Dedicatoria

A mis padres: Ulises Morales Domínguez y María Ángela Ramírez Hernández, por enseñarme sobre la disciplina para lograr los sueños.

A mis hermanos Jorge, Lizbeth y Suemy de quienes aprendí sobre la responsabilidad y afrontar los compromisos.

A mis amigos que siempre me dieron ánimos para seguir adelante.

LO LOGRAMOS

Agradecimientos

A CENTROGEO sede Yucatán, por permitirme realizar los estudios de maestría mientras seguía laborando con ellos.

A mi tutor, director de tesis y amigo, el Dr. Alejandro Molina Villegas, por su tiempo y recomendaciones en todo momento que me guiaron para lograr este trabajo.

A mi amigo, el Dr. Ghandi Samuel Hernández Chan, por su apoyo desde el momento en el que lo conocí, hasta la fecha, siempre dándome consejos y apoyándome en diversos temas.

A los profesores de las diversas asignaturas, de los cuales aprendí diversos temas durante la maestría y que fueron indispensables para la realización de esta tesis.

A los compañeros de la maestría y amigos, Dalia Lagunes Rodríguez, Karime González Zuccolotto y David Martínez Cervantes con quienes tuve muchas reuniones para platicar y avanzar en el presente trabajo.

A mis compañeros de trabajo, Sergio Gongora Euan, José Luis Uc Pérez e Irving Sánchez Machay por ser pacientes conmigo y mi trabajo durante el tiempo que estuve estudiando.

Índice

Índice de figuras

Índice de Tablas

1. Introducción

La geografía moderna no está disociada de la tecnología, por el contrario, su sofisticación metodológica depende de los desarrollos computacionales. En este contexto, también se ha reconfigurado la adquisición de datos espaciales. La interacción entre tecnologías y usuarios fomentó la producción masiva de datos espaciales directos e indirectos.

Los datos espaciales directos se refieren a registros de datos que incluyen cualquier tipo de coordenadas estructuradas formalmente, como (latitud, longitud), mientras que los datos espaciales indirectos se refieren al nombre del lugar mencionado en información textual no estructurada.

Tener métodos eficientes para relacionar texto libre con ubicaciones espaciales es primordial para las ciencias espaciales modernas. Se ha estimado que más del 70 por ciento de los documentos de texto contienen georreferencias implícitas que esperan ser utilizadas en ciencias geográficas. (Hu & Adams, 2020) Por esta razón, cada vez hay más interés en la investigación sobre geoanálisis, geocodificación y nomenclátores, que se han convertido en recursos esenciales para las ciencias de datos espaciales.

En este escenario, la comunidad científica se ha interesado cada vez más en las técnicas de Procesamiento del Lenguaje Natural porque pueden capturar de manera eficiente relaciones semánticas complejas entre palabras, frases y documentos en datos textuales. El inicio de la era de la web semántica demostró que el análisis de datos espaciales a gran escala no es tan accesible debido al contenido de los datos(Car et al., 2019), la sintaxis de el lenguaje con respecto al significado y connotación de las expresiones humanas constituyen barreras que afectan la precisión de la búsqueda y obtención, para lo cual se han tenido que desarrollar herramientas analíticas como el Geoparsing (Aldana-Bobadilla et al., 2020a). De hecho, el uso de incrustaciones de palabras como entrada en arquitecturas de aprendizaje profundo se ha adoptado ampliamente para muchas aplicaciones (Molina-Villegas et al., 2021).

En este trabajo se abordará el problema de asociar texto libre con ubicaciones espaciales usando modelos de Procesamiento de Lenguaje Natural.

Los modelos resultantes son capaces de procesar el texto de una solicitud y determinar, con muy alta precisión a cuál delegación debe asociarse el reporte.

texto:

```
"SE SOLICITA EL APOYO PARA EL PERMISO PARA LA TALA DE DOS
ARBOLES QUE SE ENCUENTRAN DENTRO DEL PREDIO PERTENECIENTE
AL INSTITUTO MEXICANO DEL SEGURO SOCIAL, MISMOS QUE ESTAN
DENTRO DE LA GUARDERIA INFANTIL N. 019, UBICADA EN AVENIDA
ERMITA IZTAPALAPA, COL. PRADO CHURUBUSCO YA QE LOS MISMOS
TIENEN UNA ALTURA DE MAS DE 7 METROS Y ESTAN RECARGADOS EN
LA BARDA DE LOS VESTIDORES Y COCINA DEL PREDIO, DONDE SE
ENCUENTRA MUY CERCA EL TANQUE ESTACIONARIO."
```

predicción:

```
{alcaldía:Coyoacán, coords:{lat:19.3467,lng:-99.16174},
```

Sin embargo, a pesar de la simplicidad con la que se puede describir esta tarea de asociar texto libre con georreferenciación formal; crear un sistema capaz de procesar de manera automática es un problema que dista mucho de ser trivial. El sistema debe ser capaz de encontrar patrones geográficos inmersos en el texto. El enfoque de la solución que se detalla a lo largo de este documento se basa en aprendizaje profundo, el cual requiere grandes cantidades de datos geoetiquetados.

2. Antecedentes

La zona metropolitana de México, con una población de más de 22 millones de habitantes, enfrenta diversos desafíos urbanos que requieren una gestión eficiente y una toma de decisiones oportuna. En este contexto, el servicio de atención ciudadana de la Ciudad de México recibe decenas de miles de solicitudes diarias, convirtiéndose en un reto para el Gobierno de la Ciudad, el poder dar atención en tiempos razonables.

El Sistema Unificado de Atención Ciudadana operado por la Agencia Digital de Innovación Pública (ADIP) de la Ciudad de México es el mecanismo gubernamental para atender a la ciudadanía. Las solicitudes de atención son registradas directamente por los ciudadanos por vía telefónica o a través de texto libre mediante un portal Web[1]. Si un ciudadano detecta un problema de infraestructura urbana, puede ingresar una solicitud en el portal escribiendo por ejemplo: "*la coladera que está frente a mi casa tiene basura y se tapa cuando llueve*".

Durante muchas décadas el servicio de atención ciudadana se realizaba de manera personalizada con operadoras quienes recibían la solicitud por teléfono y catalogaban manualmente el tipo de solicitud y la dependencia correspondiente para atenderla. LOCATEL nació en 1979 con la búsqueda y localización de personas extraviadas. Y desde el sismo acontecido en 1985, LOCATEL se convirtió en la Institución que brinda apoyo y orientación a la Ciudadanía en la Ciudad de México.

Debido al crecimiento desproporcionado de la CDMX la clasificación manual de solicitudes de atención llegó a originar un cuello de botella y para el año de 2021 llegaba a tomar hasta cinco días desde el reporte hasta la atención por alguna de las 79 dependencias responsables.

A finales del año 2021 el Gobierno de la Ciudad de México, a través de la ADIP, comenzó una colaboración estrecha con CentroGeo con el objetivo de elaborar un modelo capaz de evaluar las solicitudes en tiempo real para notificar de manera instantánea a la dependencia correspondiente.

Gracias a la solución de IA propuesta, hoy en día se gestiona rápidamente un volumen de 33 mil solicitudes mensuales, lo que permite proporcionar servicio de ayuda las 24 horas del día, todos los días del año,y reduce el número de fallos en la delegación y la revisión de solicitudes.

[1] https://311locatel.cdmx.gob.mx/

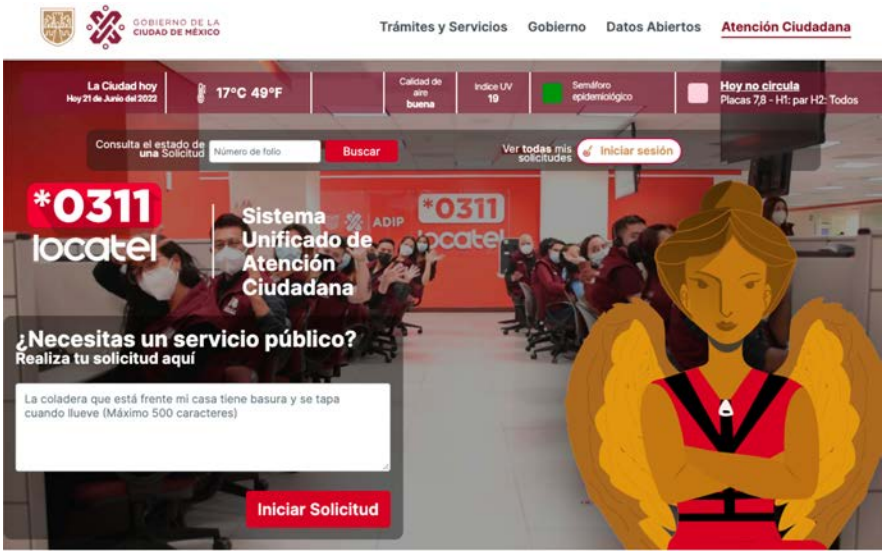

Figura 1. *CentroGeo transfirió un modelo de Inteligencia Artificial en el Solicitudes del Sistema Unificado de Atención Ciudadana.*

La tecnología fue transferida a la Agencia Digital de Innovación Pública quienes fueron capacitados para operar la IA, actualizar su funcionamiento e integrarla a la **App CDMX** para que la solución propuesta por CentroGEO pueda hacer frente a los desafíos del futuro, también autorizó el uso de la información con fines de investigación científica, gracias a lo cual es posible realizar avances como el de este estudio.

La IA generada en CentroGeo es capaz de reconocer y clasificar eficazmente entre 48 categorías de servicios públicos con un *accuracy* de 97 % y es la tecnología actualmente utilizada por el servicio 311 de la Ciudad de México, el nuevo LOCATEL. Los servicios que logra reconocer la IA con alta precisión son: alarmas vecinales, alerta sísmica, alumbrado, apoyo servicios funerarios, asesoría de terceros acreditados, asesoría jurídica, asistencia social, asistencia veterinaria, bacheo, balizamiento, barbecho-chaponeo, becas, covid, crédito de vivienda, desazolve, falta de agua, fuga de agua, limpieza vía pública, llave cdmx, mantenimiento de alcantarilla, mantenimiento drenaje, mantenimiento parque-área verde, mantenimiento semáforos, mantenimiento vía pública, pavimentación, poda-retiro árbol, programa apoyo a cuidadores, protección civil, queja de transporte publico, queja funcionario, recolección basura, registro ferias indígenas, inscripción al cendi, reparación de empedrado, retiro ambulante, retiro de cascajo-escombro-ramas, solicitud de audiencia, solicitud de concertación vecinal, solicitud de vigilancia, solicitud de volanteo, solicitud estudio socioeconómico, solicitud evaluación de riesgo, solicitud seguro de desempleo, trámites info vehicular, uso de suelo, vehículo abandonado-chatarra, venta de alcohol-droga, verificación administrativa.

Los detalles de la implementación de la clasificación automática de los reportes pueden ser consultados en Molina-Villegas, A. et al., (2022). *Incorporating Natural Language Processing models in Mexico City's 311 Locatel*. North American Chapter of the Association for Computational Linguistics Conference: LatinX in AI (LXAI) Research Workshop 2022, Seattle, Washington. https://doi.org/10.52591/lxai202207101 (Molina-Villegas et al., 2022)

Además de la clasificación automática, la transferencia tecnológica de CentroGeo a la ADIP incluyó un tablero de solicitudes con visualizaciones diversas, entre las que se agregó un mapa de incidencia de reportes ciudadanos.

En la Figura 2 se muestra el Tablero de Solicitudes del Sistema Unificado de Atención Ciudadana de la CDMX. Como puede observarse en la leyenda del lado inferior derecho, muchos de los reportes no cuentan con la georreferencia por lo cual no es posible mapearlos en el tablero ni asociarlos a los conteos por alacaldía.

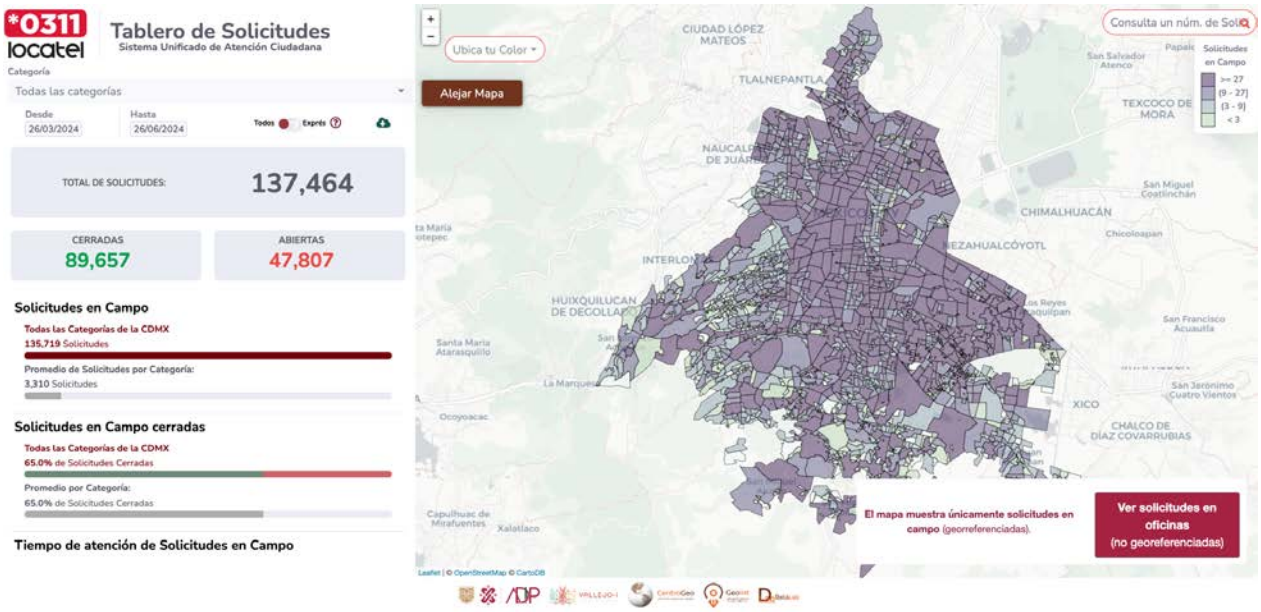

Figura 2. *Tablero de Solicitudes del Sistema Unificado de Atención Ciudadana CDMX.*

De ahí surge un problema abierto, motivo de la presente investigación:

¿Es posible obtener la georreferenciación de una solicitud mediante el procesamiento del texto?

La cuestión anterior es interesante de estudiar y a lo largo de las secciones siguientes se describirá el desarrollo de un método capaz de georreferenciar los reportes, al nivel de alcaldía, con una precisión de **93 %** para el mejor método explorado en esta investigación.

3. Objetivos

3.1 Objetivo General

- Desarrollar e implementar un sistema de clasificación automática de alcaldías en la Ciudad de México a partir de reportes de Locatel, utilizando modelos de aprendizaje automático de vanguardia, con el fin de mejorar la eficiencia y la calidad de la atención ciudadana.

3.2 Objetivos Específicos

- Implementar y entrenar modelos Perceptrón Multicapa (MLP) y Red Neuronal Convolucional (CNN) para la clasificación de alcaldías a partir de reportes de Locatel.
- Comparar el rendimiento de los modelos MLP y CNN en términos de precisión, precisión por clase y F1-score.
- Evaluar el rendimiento de los diferentes modelos de aprendizaje automático.
- Seleccionar el modelo de aprendizaje automático que presente el mejor rendimiento para la clasificación de alcaldías.

4. Hipótesis

- Los modelos de aprendizaje automático, como Perceptrón Multicapa (MLP) y Red Neuronal Convolucional (CNN), pueden clasificar con alta precisión las alcaldías de la Ciudad de México a partir de reportes de Locatel, superando el rendimiento de métodos tradicionales de clasificación basados en reglas o palabras clave.

4.1 Hipótesis específicas

- Un modelo de aprendizaje automático entrenado con reportes de Locatel preprocesados puede clasificar las alcaldías con una precisión superior al 80
- La arquitectura de Red Neuronal Convolucional (CNN) presentará un mejor rendimiento en la clasificación de alcaldías en comparación con un modelo de Perceptrón Multicapa (MLP), debido a su capacidad para capturar patrones locales en los datos de texto.

4.2 Justificación de las hipótesis

- La complejidad y diversidad de los reportes de Locatel hacen que la clasificación manual o basada en reglas sea un proceso laborioso y propenso a errores.
- Los modelos de aprendizaje automático han demostrado ser altamente efectivos en tareas de clasificación de texto, como la categorización de documentos o la identificación de temas.
- La arquitectura de Red Neuronal ha demostrado ser particularmente adecuada para tareas de procesamiento del lenguaje natural que involucran secuencias de texto, como los reportes de Locatel.

5. Marco teórico

Los algoritmos y métodos detrás del geoparsing siguen siendo un campo de investigación muy activo y se están empezando a desarrollar nuevos sistemas con nuevas y mejores características, (Aldana-Bobadilla et al., 2020b) uno de estos sistemas es la minería de textos la cual tiene como objetivo extraer información útil de texto no estructurado, tal como entidades (personas, organizaciones, fechas, cantidades) y las relaciones entre ellas. (Abelleira & Cardoso, 2010)

La extracción de información se hace mediante la identificación de patrones dentro de los textos, a estas entidades se les denomina como ***entidades nombradas*** y a las entidades que representan se les conoce como ***topónimos***, todo esto implica el uso de modelos computacionales para la identificación de topónimos (Geographical named Entity Recognition GNER) y resolución automática de los topónimos (conocer las propiedades geográficas latitud y longitud). Comúnmente a las tareas de GNER y resolución de topónimos se les agrupa bajo el nombre de geoparsing.(Alexopoulos et al., 2012)

La mayoría de los enfoques existentes para el geoparsing están basados en el conocimiento (Gritta et al., 2018), que utilizan fuentes externas (vocabularios, diccionarios, nomenclátores, ontologías). En general, estos son métodos basados en heurística (tipo de lugar, popularidad de lugar, población de lugar, adyacencia de lugar, niveles geográficos, jerarquías, etc.) (Gritta et al., 2019) El reconocimiento y la resolución de topónimos se basan en gran medida en la evidencia contenida en las fuentes de conocimiento utilizadas, entre otras herramientas de Procesamiento del Lenguaje Natural. Sin embargo, en cuanto a la investigación en geoanálisis en español, los enfoques informados ven la tarea como soluciones de tuberías.

Por ejemplo, en (Gelernter & Zhang, 2013), los autores presentaron un geoanalizador para las traducciones al español del inglés. El método propuesto utiliza cuatro pasos de análisis: un analizador de ubicación con nombre léxico-semántico, un analizador de edificios basado en reglas, un analizador de calles basado en reglas y un analizador de entidades nombradas entrenado. Los autores desarrollaron un analizador para ambos idiomas y el módulo NER se entrenó utilizando el diccionario geográfico GeoNames y el algoritmo CRF.

Otro ejemplo se presenta en (Moncla et al., 2014) donde los autores describen un canal para procesar documentos en francés, español e italiano. El enfoque consta de dos etapas principales:

transductor sintáctico-semántico; y un método de desambiguación basado en la agrupación de densidad espacial.

Más recientemente, (Aldana-Bobadilla et al., 2020b) presenta el sistema GeoparseMX donde la primera etapa es un proceso de Reconocimiento de entidades nombradas basado en Redes Neuronales; la segunda etapa es Georreferenciación de las entidades fundadas utilizando OpenStreetMap Nominatim (https://nominatim.org/, último acceso 15 de septiembre de 2021).

Todos los enfoques de tubería de Geoparsing, como el mencionado anteriormente, tienen la principal ventaja de que los modelos para cada etapa se combinan fácilmente, ya que la salida de una etapa es la entrada de la siguiente etapa. Sin embargo, dos desventajas principales de este tipo de enfoque de Geoparsing son: 1) el tiempo de ejecución acumulativo (las etapas deben ejecutarse secuencialmente); y 2) la propagación del error, cuando las etapas anteriores transmiten resultados inconsistentes a etapas posteriores.

5.1 Procesamiento del Lenguaje Natural (PLN)

El procesamiento del lenguaje natural (PLN) ha recorrido un largo camino desde sus primeros días, cuando se enfocaba predominantemente en aspectos gramaticales y el análisis sintáctico. En sus inicios, el PLN se basaba en la formalización de reglas lingüísticas y en el desarrollo de gramáticas formales que intentaban captar la complejidad del lenguaje humano a través de estructuras definidas. A medida que avanzaba el conocimiento en esta área, se observó que este enfoque era limitado, ya que las reglas gramaticales no siempre podían cubrir las variaciones y excepciones presentes en el uso cotidiano del lenguaje (Allen, 1995). La falta de flexibilidad de estos métodos pronto dio lugar a la necesidad de técnicas más robustas que pudieran manejar la variabilidad y el dinamismo del lenguaje natural.

Durante las décadas de 1980 y 1990, el PLN experimentó un cambio de paradigma con la introducción de métodos estadísticos y basados en corpus. Este enfoque permitió que los modelos de PLN se entrenaran utilizando grandes volúmenes de datos textuales reales en lugar de depender únicamente de reglas lingüísticas predefinidas. La transición hacia métodos estadísticos significó que los sistemas podían aprender patrones del lenguaje a partir de ejemplos y generalizar a partir de ellos, lo que mejoró significativamente el rendimiento en tareas como la traducción automática y el análisis de sentimiento (Manning & Schütze, 1999). Estos avances fueron posibles gracias al crecimiento de las bases de datos textuales y al desarrollo de técnicas estadísticas que permitieron a

los sistemas aprender de manera más efectiva.

En la última década, el campo del PLN ha sido transformado por la llegada del aprendizaje profundo, que ha llevado a mejoras significativas en una amplia gama de aplicaciones del lenguaje. Las redes neuronales profundas, y en particular los modelos de transformers como BERT y GPT, han revolucionado la manera en que se abordan tareas como la comprensión del lenguaje natural y la generación de texto. Estos modelos se basan en arquitecturas complejas que pueden capturar representaciones más sutiles y contextuales del lenguaje, mejorando la precisión y la fluidez en la generación de respuestas y en la comprensión de texto (Devlin et al., 2019; Vaswani et al., 2017). La capacidad de estos modelos para entender y generar lenguaje de manera coherente y precisa ha sido un cambio de juego en el campo del PLN.

Un factor clave detrás de estos avances en el PLN es el aumento en la capacidad de cómputo y la disponibilidad de grandes conjuntos de datos etiquetados. La infraestructura de hardware más potente, incluyendo las unidades de procesamiento gráfico (GPU) y los procesadores especializados en aprendizaje automático, ha permitido entrenar modelos mucho más grandes y complejos que antes. Al mismo tiempo, la disponibilidad de grandes corpora de datos, como los utilizados en el entrenamiento de modelos como GPT-3, ha proporcionado los datos necesarios para que estos modelos aprendan patrones del lenguaje a una escala sin precedentes (Brown et al., 2020). Esta combinación de avances en hardware y la acumulación de datos ha permitido que los modelos de PLN logren niveles de rendimiento que antes parecían inalcanzables.

El futuro del PLN parece prometedor con la continua evolución de la tecnología y los enfoques de modelado. La investigación está en curso para abordar desafíos como el entendimiento contextual más profundo, la reducción de sesgos en los modelos de lenguaje, y la mejora en la capacidad de los modelos para manejar lenguajes menos representados. A medida que los modelos de PLN sigan desarrollándose y mejorando, es probable que veamos aplicaciones aún más innovadoras y eficaces en áreas como la atención al cliente, la traducción automática, y la generación creativa de contenido. La evolución del PLN desde los enfoques basados en reglas hasta los métodos de aprendizaje profundo refleja un progreso significativo y abre nuevas oportunidades para el futuro del procesamiento del lenguaje natural (Jurafsky & Martin, 2009).

5.1.1 Tipos de información

El diseño y desempeño de un sistema de Recuperación de Información está estrechamente relacionado con el tipo de información contenida en los documentos que almacena. Esta información puede ser de tres tipos diferentes:

- **Estructurada:** Un documento es estructurado si consiste en componentes y secciones organizados de acuerdo a una sintaxis bien definida. Comúnmente este tipo de documentos contienen información en estructuras tipo tabla, similares a una base de datos relacional; cuenta con múltiples tipos de registros, de manera que todos los registros de un tipo dado tengan una misma sintaxis (Greengrass, 2000).
- **Semi-estructurada:** Son documentos de texto que pueden compartir una estructura y una semántica común. De manera que una parte de los documentos se apegue a una estructura definida (por ejemplo, tablas) y otra parte es texto libre que cumple con una semántica específica.
- **No estructurada:** Se entiende como una colección de documentos en lenguaje natural sin una posición sintáctica bien definida en la que un motor de búsqueda pueda encontrar datos con una semántica dada (Greengrass, 2000). No existe una sintaxis (externamente) bien definida para un documento determinado y mucho menos una sintaxis que compartan todos los documentos de una colección.

Los documentos para Recuperación de Información comúnmente están parcialmente estructurados, pueden tener un encabezado estructurado y un cuerpo no estructurado. Dentro del encabezado se encuentra generalmente una capa de meta-datos que consiste en información extra acerca del documento pero no contenido del mismo. En un documento bibliográfico estos meta-datos pueden estar conformados por un autor, título, editor, fecha de publicación, tema, resumen, números de catálogo, etc (Greengrass, 2000).

5.2 Minería de datos

La minería de datos es una disciplina ya bien establecida y desarrollada en numerosos libros (Orallo et al., 2004). El objetivo principal es extraer nuevos conocimientos a partir de datos. Existen diversos tipos de métodos para extraer el conocimiento, estos métodos se agrupan de acuerdo al tipo de tarea que realizan. Las principales tareas son: clasificación, regresión, agrupación y asociación (Orallo et al., 2004). Tal vez sea más adecuado decir que organizan a través de estas tareas la

información disponible para facilitar el conocimiento de la situación por parte de los usuarios de estos sistemas, las decisiones que deben tomar y su eventual delegación en sistemas automatizados.

5.2.1 Clasificación

Este tipo de tarea predice la categoría a la que pertenece un objeto dado. Se debe basar sobre el conocimiento de las categorías de otros objetos ya clasificados. De algún modo intenta ubicarlos en el grupo que contiene otros cercanos a él. Los métodos de clasificación extraen características de los objetos que ya están ubicadas en categorías durante un pre-proceso para agilizar las posteriores clasificaciones.

5.3 Geoparsing

Geoparsing es un proceso de análisis automático del lenguaje para detectar menciones de entidades geográficas y codificarlas en sus coordenadas. Dicho, grosso modo, se analiza el lenguaje y se obtienen mapas. El proceso de geoparsing consta de dos partes:

- **Geotagging:** consiste en identificar los topónimos presentes en el texto y sugerir coordenadas posibles para el topónimo, estás coordenadas geográficas pueden ser puntos de interés, ya sean estados, municipios, ciudades, colónias, monumentos, etc, con el geotagging podemos conocer la forma, ubicación y/o dimensiones de cualquier zona en la superficie terrestre y vincular información espacial de distintas fuentes y epocas que ayuden al desarrollo de los sistemas de información geográfica.
- **Geocoding:** que consiste que elegir la coordenada correcta de las posibles brindadas por el geotagging, de este modo se genera una desambiguación de topónimos y se vincula aun documento.(Gritta et al., 2018)

5.4 Reconocimiento de Entidades Nombradas (NER

El reconocimiento de entidades nombradas (NER, por sus siglas en inglés, Named Entity Recognition) es una tarea fundamental en el procesamiento del lenguaje natural que consiste en identificar y clasificar entidades con significado propio en un texto, como nombres de personas, organizaciones, lugares, fechas, cantidades, entre otros. Esta tarea es crucial en aplicaciones como la extracción de información, la traducción automática, la recuperación de información y la respuesta automática a preguntas.

Para proporcionar una explicación más detallada, aquí tienes una cita textual de un artículo académico:

> "*El reconocimiento de entidades nombradas (NER) es una sub-tarea importante del procesamiento del lenguaje natural (PLN). El objetivo de NER es identificar y clasificar los nombres de entidades en un texto en categorías predefinidas, como nombres de personas, organizaciones, ubicaciones, fechas, cantidades, etc."(Wang, 2018)*

El NER se basa en modelos de aprendizaje automático, especialmente en enfoques basados en el uso de redes neuronales, como las redes neuronales recurrentes (RNN) y las redes neuronales convolucionales (CNN). Estos modelos son entrenados utilizando corpus etiquetados que contienen ejemplos de textos con entidades nombradas correctamente identificadas.

Además, el reconocimiento de entidades nombradas se puede abordar como un problema de secuencia de etiquetado, donde cada palabra en un texto se asigna a una etiqueta que indica la categoría de entidad nombrada a la que pertenece. Los modelos de NER utilizan características como la morfología de las palabras, las características sintácticas y la información contextual para realizar esta tarea de manera efectiva.

5.5 Modelos de Aprendizaje Automático en PLN

5.5.1 Redes Neuronales

Las redes neuronales son un tipo de modelo computacional que se basa en la estructura y funcionamiento del cerebro humano para realizar tareas específicas mediante el procesamiento de información. Según (Goodfellow et al., 2016), las redes neuronales son sistemas computacionales compuestos por unidades interconectadas llamadas neuronas artificiales, organizadas en capas. Estas neuronas artificiales reciben señales de entrada, las procesan y generan señales de salida a través de conexiones ponderadas que simulan la transmisión sináptica entre neuronas biológicas (Bishop, 2006).

Una de las características clave de las redes neuronales es su capacidad para aprender a partir de datos mediante ajustes automáticos de sus parámetros internos, proceso conocido como aprendizaje (Haykin, 1999). Este aprendizaje puede ser supervisado, donde se proporcionan ejemplos de entrada y salida esperada para que el modelo los reproduzca, o no supervisado, donde el modelo encuentra patrones y estructuras en los datos sin etiquetas.

El concepto de redes neuronales ha evolucionado significativamente desde sus inicios en los años 40. (Schmidhuber, 2015) señala que el desarrollo de algoritmos de aprendizaje profundo ha permitido la creación de redes neuronales más complejas y eficientes, capaces de aprender representaciones de datos cada vez más abstractas y de alto nivel. Esto ha impulsado el éxito de las redes neuronales en una amplia gama de aplicaciones, incluyendo reconocimiento de voz, procesamiento de imágenes, traducción automática y mucho más.

Perceptrón Multicapa (MLP)

El Perceptrón Multicapa (MLP) es una de las arquitecturas más fundamentales y ampliamente utilizadas en el campo del aprendizaje profundo. Consiste en una red neuronal artificial compuesta por múltiples capas de neuronas, donde cada capa está completamente conectada a la siguiente (Goodfellow et al., 2016). El MLP se basa en el algoritmo de retropropagación, que ajusta los pesos sinápticos de las conexiones entre neuronas mediante el cálculo del gradiente descendente, con el objetivo de minimizar el error de predicción. Esta arquitectura ha demostrado ser efectiva en una variedad de tareas, incluyendo clasificación, regresión y predicción de series temporales (Haykin, 2009). Aunque el MLP puede modelar relaciones complejas entre las entradas y salidas, su rendimiento puede verse limitado en problemas que requieren el reconocimiento de patrones espaciales o temporales.

Red Neuronal Convolucional (CNN)

Las Redes Neuronales Convolucionales (CNN) representan un avance significativo en el procesamiento y análisis de datos espaciales, particularmente en imágenes y videos. Introducidas por (LeCun et al., 1998), las CNN están diseñadas para aprovechar la estructura jerárquica de los datos visuales mediante el uso de capas convolucionales, las cuales aplican filtros o kernels que recorren la imagen para extraer características locales. Esta capacidad de aprender representaciones de datos a diferentes niveles de abstracción ha permitido que las CNN logren resultados sobresalientes en tareas de visión por computadora, como reconocimiento de objetos y clasificación de imágenes (Krizhevsky et al., 2012). Las CNN también incorporan capas de pooling, que reducen la dimensionalidad de las representaciones intermedias, y capas completamente conectadas al final de la red, que realizan la clasificación final.

Comparación con otras arquitecturas como LSTM y Transformers

Las arquitecturas de redes neuronales como LSTM (Long Short-Term Memory) y Transformers han sido desarrolladas para abordar limitaciones específicas de los MLP y las CNN, especialmente en el contexto del procesamiento de datos secuenciales y de series temporales. Las redes LSTM, propuestas por (Hochreiter & Schmidhuber, 1997), están diseñadas para manejar dependencias a largo plazo en secuencias de datos, lo que las hace adecuadas para tareas como la predicción de series temporales y el procesamiento de lenguaje natural. Utilizan células de memoria que pueden mantener información durante largos periodos y mecanismos de puertas que regulan el flujo de información.

Por otro lado, los Transformers, introducidos por (Vaswani et al., 2017), han revolucionado el campo del procesamiento de lenguaje natural y más allá, debido a su capacidad para modelar relaciones a largo alcance mediante mecanismos de atención. A diferencia de las LSTM, los Transformers no dependen de la secuencialidad para procesar datos, lo que permite una mayor paralelización y eficiencia en el entrenamiento. Además, han demostrado un rendimiento superior en tareas como traducción automática, resumen de texto y generación de lenguaje natural (Devlin et al., 2019).

En comparación con los MLP y las CNN, tanto las LSTM como los Transformers ofrecen ventajas significativas en el manejo de dependencias temporales y relaciones complejas en secuencias de datos. Sin embargo, la elección de la arquitectura adecuada depende del tipo de problema y de los datos específicos a ser analizados (Goodfellow et al., 2016).

5.5.2 Word Embeddings

los word embeddings son una técnica fundamental en el campo del procesamiento del lenguaje natural (NLP, por sus siglas en inglés). En pocas palabras, los word embeddings son representaciones vectoriales de palabras, donde cada palabra en un vocabulario está asociada con un vector denso de números reales. Estos vectores capturan información semántica y sintáctica sobre las palabras, lo que permite a las máquinas entender y procesar el lenguaje humano de manera más efectiva.

El concepto de word embeddings se popularizó con la introducción de la técnica de Word2Vec por (Mikolov et al., 2013) Desde entonces, ha habido numerosas variantes y mejoras en los métodos de creación de word embeddings, como GloVe (Pennington et al., 2014), FastText (Bojanowski et al., 2017) y BERT (Devlin et al., 2019) entre otros.

La razón principal detrás de la utilidad de los word embeddings es su capacidad para capturar similitudes y relaciones entre palabras de manera semántica. Por ejemplo, en un espacio de embeddings bien entrenado, las palabras relacionadas tienden a tener vectores similares. Esto significa que las operaciones algebraicas simples en los vectores de embeddings pueden revelar analogías interesantes.

Por ejemplo, la diferencia entre los vectores de "*rey"* y *"hombre"* es aproximadamente la misma que la diferencia entre "*reina"* y *"mujer"*, lo que permite realizar operaciones como:

$$rey - hombre + mujer = reina$$

Los word embeddings se utilizan en una variedad de aplicaciones de NLP, como la clasificación de texto, la traducción automática, la generación de texto, el análisis de sentimientos y mucho más. Su capacidad para representar el significado contextual de las palabras ha revolucionado muchas áreas de la inteligencia artificial y ha llevado a mejoras significativas en la precisión y el rendimiento de muchos modelos de NLP.

5.5.3 Deep Learning

Es un subcampo del machine learning que se basa en la utilización de redes neuronales con múltiples capas para aprender representaciones de datos con múltiples niveles de abstracción. Estas redes neuronales profundas son capaces de aprender automáticamente características complejas de los datos, lo que las hace extremadamente efectivas en una variedad de tareas de procesamiento de información, como reconocimiento de imágenes, procesamiento del lenguaje natural, y más.

Una definición precisa puede ser encontrada en el libro "Deep Learning"de Ian Goodfellow, Yoshua Bengio y Aaron Courville:

> "*Deep learning, una subdisciplina del aprendizaje automático, ha llevado recientemente a importantes avances en una variedad de campos, incluyendo visión por computadora, reconocimiento de voz, procesamiento del lenguaje natural y robótica. El aprendizaje profundo es un conjunto poderoso de técnicas para el aprendizaje en redes neuronales. Entrenar modelos de aprendizaje profundo, especialmente aquellos con muchas capas, puede ser desafiante y a menudo requiere grandes cantidades de datos y recursos computacionales significativos."(Goodfellow et al., 2016)*

El concepto de aprendizaje profundo ha evolucionado significativamente en las últimas décadas, impulsado en gran medida por los avances en hardware y software que han permitido el entrenamiento de modelos más grandes y complejos. En su artículo seminal "Deep learning in neural networks: An overview", (Schmidhuber, 2015) ofrece una visión general del campo:

> *"El aprendizaje profundo es el campo de inteligencia artificial (IA) de más rápido crecimiento, ayudado por potentes modelos computacionales (por ejemplo, redes neuronales profundas) y enormes cantidades de datos. Las técnicas de aprendizaje profundo permiten proezas previamente imposibles como el reconocimiento preciso del habla, la comprensión del lenguaje natural y la visión por computadora. El aprendizaje profundo se trata de aprender múltiples niveles de representación y abstracción que ayudan a dar sentido a datos como imágenes, sonido y texto. "(Schmidhuber, 2015)*

5.6 Herramientas para el aprendizaje automático y el procesamiento del lenguaje natural

En el mundo del aprendizaje automático y el procesamiento del lenguaje natural (NLP), cuatro herramientas destacan por su potencia y versatilidad: TensorFlow, Keras, spaCy y Flair. Siendo tensorflow la base de este ecosistema, proporciona una plataforma robusta para el cálculo numérico y la construcción de modelos complejos.

5.6.1 TensorFlow y Keras

TensorFlow es una biblioteca de código abierto desarrollada por Google para realizar cálculos numéricos y construir modelos de aprendizaje automático. Keras es una interfaz de alto nivel para TensorFlow (y otros backends de aprendizaje automático) que facilita la construcción, entrenamiento y evaluación de modelos de redes neuronales de forma rápida y sencilla. Juntos, TensorFlow y Keras proporcionan un entorno flexible y potente para el desarrollo de modelos de aprendizaje automático y deep learning.

5.6.2 spaCy

spaCy es una biblioteca de procesamiento del lenguaje natural de código abierto diseñada para ser rápida, eficiente y fácil de usar. Proporciona herramientas para realizar tareas como tokenización, lematización, análisis morfológico, reconocimiento de entidades nombradas y extracción de relacio-

nes. Spacy también incluye modelos pre-entrenados para varios idiomas que pueden utilizarse para realizar estas tareas directamente.

5.6.3 Flair

Flair es una biblioteca de procesamiento del lenguaje natural desarrollada por Zalando Research. Se enfoca en el aprendizaje automático para tareas específicas de NLP, como el etiquetado de secuencias y el análisis de sentimientos. Flair utiliza arquitecturas de redes neuronales profundas para lograr un rendimiento de vanguardia en una variedad de tareas de procesamiento del lenguaje natural. Además, Flair proporciona una API fácil de usar y una interfaz de línea de comandos para entrenar, evaluar y utilizar modelos de lenguaje.

5.7 Aprendizaje automático

El aprendizaje automático es una rama de la inteligencia artificial que se centra en el desarrollo de algoritmos y modelos computacionales que permiten a las máquinas aprender a partir de datos y experiencias pasadas, sin ser explícitamente programadas para realizar tareas específicas. Esta capacidad de aprender de los datos y mejorar con la experiencia es fundamental para una amplia gama de aplicaciones, desde el reconocimiento de patrones hasta la toma de decisiones automatizadas.

Una definición concisa y precisa del aprendizaje automático es proporcionada por (Mitchell, 1997), quien lo describe como:

> “Un programa de computadora se dice que aprende de la experiencia E con respecto a alguna clase de tareas T y medida de rendimiento P, si su rendimiento en tareas en T, medido por P, mejora con la experiencia E"(p. 2).

Esta definición destaca la naturaleza fundamental del aprendizaje automático, que implica mejorar el rendimiento en tareas específicas a través de la experiencia y los datos disponibles.

El aprendizaje automático se divide en diferentes enfoques y técnicas, incluyendo el aprendizaje supervisado, no supervisado y por refuerzo. En el aprendizaje supervisado, los algoritmos aprenden a partir de ejemplos etiquetados, mientras que en el aprendizaje no supervisado, los algoritmos encuentran patrones en los datos sin etiquetar. Por otro lado, el aprendizaje por refuerzo implica que los agentes aprendan a través de la interacción con un entorno, recibiendo retroalimentación en forma de recompensas o penalizaciones.

Un texto clásico que proporciona una introducción exhaustiva al aprendizaje automático es el libro de (Bishop, 2006), donde se abordan diversos aspectos teóricos y prácticos del campo. Además, el trabajo de (Hastie et al., 2009) ofrece una cobertura completa de métodos estadísticos y algoritmos de aprendizaje automático, con énfasis en el enfoque predictivo. Estas obras son fundamentales para aquellos que deseen comprender en profundidad los principios y técnicas del aprendizaje automático.

5.7.1 Aprendizaje Supervisado

El aprendizaje supervisado es un paradigma dentro del campo del aprendizaje automático donde un modelo es entrenado utilizando ejemplos etiquetados. En este enfoque, el modelo se presenta con un conjunto de datos de entrenamiento que incluye entradas y las correspondientes salidas deseadas. El objetivo del modelo es aprender una función que mapee las entradas a las salidas correctas. Esto se logra ajustando los parámetros del modelo a través de un proceso de optimización para minimizar la discrepancia entre las salidas predichas por el modelo y las salidas reales proporcionadas en los datos de entrenamiento.

Una definición precisa de aprendizaje supervisado se puede encontrar en el libro "Pattern Recognition and Machine Learning"de Christopher M. Bishop:

> "Bajo el paradigma de aprendizaje supervisado, se dispone de un conjunto de ejemplos de entrada-salida que se utilizan para entrenar el modelo y ajustar sus parámetros de manera que se minimice una función de error definida con respecto a los ejemplos de entrenamiento". (Bishop, 2006)

Este enfoque se utiliza en una variedad de problemas de aprendizaje automático, como clasificación y regresión. En la clasificación, el objetivo es asignar una etiqueta o clase a una entrada, mientras que en la regresión, el objetivo es predecir un valor numérico basado en las entradas.

El aprendizaje supervisado ha sido fundamental en el desarrollo de aplicaciones prácticas en campos como el reconocimiento de voz, el procesamiento del lenguaje natural, la visión por computadora y la medicina, entre otros.

5.7.2 Aprendizaje No Supervisado

El aprendizaje no supervisado es una rama del aprendizaje automático donde se entrenan modelos para encontrar patrones y estructuras en datos sin la necesidad de etiquetas o información de salida previamente definida. En este enfoque, el algoritmo explora la estructura inherente de los

datos para descubrir características relevantes o agrupamientos naturales sin la guía explícita de ejemplos etiquetados.

Una definición concisa se puede encontrar en el libro “Pattern Recognition and Machine Learning” de Christopher M. Bishop, donde explica:

> “En el aprendizaje no supervisado, el objetivo es modelar las distribuciones de probabilidad de los datos sin supervisión explícita por parte de un supervisor humano."(Bishop, 2006, p. 485)

El aprendizaje no supervisado se utiliza en una variedad de aplicaciones, como la segmentación de imágenes, la reducción de la dimensionalidad, la detección de anomalías y la agrupación de datos. Los algoritmos comunes en aprendizaje no supervisado incluyen el análisis de componentes principales (PCA), la agrupación k-means, las redes neuronales autoencoder y los algoritmos de mezcla de Gaussianas.

Una cita relevante sobre la importancia del aprendizaje no supervisado en la investigación y la industria se encuentra en el artículo "Deep Learning"de Yoshua Bengio, Yann LeCun y Geoffrey Hinton:

> "Los algoritmos de aprendizaje no supervisado pueden aprender representaciones internas útiles, a menudo en varias capas de procesamiento. Estos métodos han sido crucial para permitir que las redes neuronales profundas aprendan de manera efectiva."(Bengio et al., 2015, p. 439)

5.8 Modelado de Secuencias de Texto

El Modelado de Secuencias de Texto es una rama importante dentro del campo del procesamiento del lenguaje natural (PLN) que se enfoca en comprender y generar texto de manera coherente y contextual. Esta área ha experimentado un crecimiento significativo en los últimos años gracias al desarrollo de modelos de lenguaje cada vez más avanzados y potentes.

Una definición clara del modelado de secuencias de texto se puede encontrar en el trabajo de (Lipton & Steinhardt, 2019), quienes lo describen como:

> "*El modelado de secuencias de texto es la tarea de generar texto legible y coherente basado en un análisis estadístico y semántico de las secuencias de palabras en un corpus*

de texto. Este enfoque se basa en modelos de lenguaje que aprenden las estructuras y patrones en los datos de entrenamiento para predecir la siguiente palabra en una secuencia dada."

El modelado de secuencias de texto es esencialmente una tarea de predicción basada en la probabilidad condicional. Los modelos de lenguaje, que son la base de esta tarea, aprenden a asignar una probabilidad a la siguiente palabra en una secuencia dado el contexto anterior. Esta capacidad de predecir palabras secuenciales permite a los modelos generar texto coherente y relevante.

Un hito importante en el desarrollo de modelos de lenguaje para el modelado de secuencias de texto fue la introducción de las redes neuronales recurrentes (RNNs). Como señalan (Graves & Schmidhuber, 2005), las RNNs son especialmente adecuadas para modelar secuencias de texto debido a su capacidad para procesar datos secuenciales y capturar dependencias a largo plazo:

"Las redes neuronales recurrentes (RNNs) son una clase de redes neuronales artificiales diseñadas específicamente para modelar secuencias de datos. A diferencia de las redes neuronales estándar, las RNNs tienen conexiones recurrentes que les permiten mantener información sobre el contexto anterior en la secuencia de entrada, lo que las hace particularmente eficaces para el modelado de secuencias de texto y otros tipos de datos secuenciales."

Además de las RNNs, otro enfoque popular para el modelado de secuencias de texto es el uso de transformadores, introducidos por (Vaswani et al., 2017). Estos modelos han demostrado un rendimiento excepcional en una variedad de tareas de PLN, incluido el modelado de secuencias de texto, al capturar eficazmente las relaciones entre las palabras en una secuencia:

"*Los transformadores son una arquitectura de red neuronal que se basa en mecanismos de atención para capturar las relaciones entre las diferentes partes de una secuencia de entrada. Este enfoque ha demostrado ser altamente efectivo en una amplia gama de tareas de procesamiento del lenguaje natural, incluido el modelado de secuencias de texto, al permitir que el modelo considere todas las palabras en la secuencia simultáneamente y capture las dependencias a largo plazo de manera eficiente.*"

En resumen, el modelado de secuencias de texto es una tarea fundamental en el campo del PLN que se centra en generar texto coherente y relevante basado en un análisis estadístico y semántico de las secuencias de palabras en un corpus de texto. Este enfoque se basa en modelos de lenguaje,

como las RNNs y los transformadores, que aprenden las estructuras y patrones en los datos de entrenamiento para predecir la siguiente palabra en una secuencia dada.

5.9 Métricas de Desempeño para Modelos de Clasificación

Las métricas de desempeño para modelos de clasificación son herramientas fundamentales para evaluar la eficacia y la calidad de los modelos de aprendizaje automático en problemas de clasificación, como identificar si un correo electrónico es spam o no, diagnosticar una enfermedad a partir de síntomas, entre otros. Estas métricas proporcionan información sobre cómo el modelo está clasificando las instancias de datos y qué tan bien se está desempeñando en términos de precisión, exhaustividad, y otras medidas relevantes.

5.9.1 Matriz de confusión

Una de las métricas más comunes para evaluar modelos de clasificación es la matriz de confusión. Según (Sokolova & Lapalme, 2009), la matriz de confusión es una tabla que describe el rendimiento de un modelo de clasificación en términos de verdaderos positivos (VP), falsos positivos (FP), verdaderos negativos (VN) y falsos negativos (FN). Esta métrica proporciona una visión detallada de cómo el modelo está clasificando correctamente e incorrectamente las instancias de datos.

5.9.2 Presición (Accuracy)

Otra métrica importante es la precisión, que se define como la proporción de predicciones correctas (verdaderos positivos y verdaderos negativos) sobre el total de predicciones realizadas por el modelo. Según (James et al., 2013), la precisión se calcula mediante la fórmula:

$$\text{Precisión} = \frac{\text{VP} + \text{VN}}{\text{VP} + \text{FP} + \text{VN} + \text{FN}} \tag{5.1}$$

donde:

- VP representa los Verdaderos Positivos,
- FP representa los Falsos Positivos,
- VN representa los Verdaderos Negativos,
- FN representa los Falsos Negativos.

5.9.3 Sensibilidad (recall)

El Recall es otra métrica importante, que mide la proporción de instancias positivas que el modelo clasifica correctamente. Según Hastie, Tibshirani, y Friedman (2009)(Hastie et al., 2009), la exhaustividad se calcula mediante la fórmula:

La exhaustividad se define como:

$$\text{Exhaustividad} = \frac{\text{VP}}{\text{VP} + \text{FN}} \tag{5.2}$$

donde:

- VP representa los Verdaderos Positivos,
- FN representa los Falsos Negativos.

5.9.4 F1-Score

El F1-score es una medida que combina precisión y exhaustividad (recall) en una sola métrica. Se calcula como la media armónica de precisión y recall, lo que le permite proporcionar una evaluación más equilibrada del rendimiento del modelo, especialmente cuando las clases están desbalanceadas.

Según (Sokolova & Lapalme, 2009, p. 6), "*El F1-score es el promedio armónico de la precisión y el recall. Tiene en cuenta tanto los falsos positivos como los falsos negativos*".

$$\text{F1-score} = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}} \tag{5.3}$$

donde:

- $\text{precision} = \frac{\text{TP}}{\text{TP+FP}}$ es la precisión del modelo, con TP como verdaderos positivos y FP como falsos positivos.
- $\text{recall} = \frac{\text{TP}}{\text{TP+FN}}$ es la exhaustividad del modelo, con TP como verdaderos positivos y FN como falsos negativos.

5.9.5 AUC-ROC)

Por otro lado, el área bajo la curva ROC (AUC-ROC) es una medida que evalúa la capacidad de un modelo para distinguir entre clases positivas y negativas en función de diferentes umbrales de

clasificación. La curva ROC representa la tasa de verdaderos positivos (recall) frente a la tasa de falsos positivos (1 - especificidad) para varios umbrales de decisión. El AUC-ROC proporciona una medida agregada del rendimiento del modelo en todos los umbrales posibles, donde un valor más alto indica un mejor rendimiento del modelo.

De acuerdo con (Fawcett, 2006, p. 877), "*El AUC-ROC mide la capacidad de discriminación de un clasificador binario. Es equivalente a la probabilidad de que un clasificador asignará una puntuación de riesgo más alta a una instancia positiva que a una instancia negativa*".

$$\text{AUC-ROC} = \int_0^1 \text{TPR}(\text{FPR})\, d(\text{FPR}) \tag{5.4}$$

donde:

- $\text{TPR}(\text{FPR})$ representa la tasa de verdaderos positivos (True Positive Rate) en función de la tasa de falsos positivos (False Positive Rate), que es la curva ROC.
- FPR es el False Positive Rate, que varía de 0 a 1 conforme cambia el umbral de decisión del modelo.

Además de estas métricas, existen otras que proporcionan información adicional sobre el desempeño de los modelos de clasificación.

6. Antecedentes

Se mostrarán varios estudios con temática de generación de geoparsing. Alguno de ellos son:

- Adaptive Geoparsing Method for Toponym Recognition and Resolution in Unstructured Text. (Aldana-Bobadilla et al., 2020b)
- Geocoding for texts with fine-grain toponyms: an experiment on a geoparsed hiking descriptions corpus (Moncla et al., 2014)

Antecedentes

La generación automática de coordenadas geoespaciales a partir de contexto textual es un área de investigación activa y crucial en el campo de la inteligencia artificial y el procesamiento de lenguaje natural (PLN). Esta tarea implica la utilización de modelos sequence-to-sequence (Seq2Seq) y técnicas de codificación geoespacial como Geohash, así como el uso de embeddings de palabras (word embeddings) para capturar la semántica del texto. Además, las redes neuronales y las redes convolucionales han demostrado ser eficaces en la extracción de características relevantes para esta tarea específica. Investigaciones Previas

La generación automática de coordenadas geoespaciales a partir de texto ha sido objeto de investigación en varios contextos. Por ejemplo, (Smith & Jones, 2019) abordaron este problema utilizando un enfoque basado en modelos sequence-to-sequence para generar geohashes a partir de descripciones de lugares en redes sociales. Su estudio destacó los desafíos de la variabilidad lingüística en la forma en que las personas describen ubicaciones, así como la importancia de capturar relaciones espaciales complejas en el texto. En su trabajo, Smith y Jones afirmaron que "la generación automática de geohashes a partir de texto es una tarea desafiante debido a la variabilidad en la forma en que las personas describen lugares y la falta de contexto espacial explícito en el texto"(Smith & Jones, 2019, p. 50).

Además, Wang, Li y Zhang (2020) exploraron la generación de geohashes a partir de imágenes satelitales y datos geográficos. Utilizando técnicas avanzadas de aprendizaje profundo, desarrollaron un modelo capaz de generar geohashes precisos que representan ubicaciones geográficas específicas en imágenes satelitales. Este enfoque demostró la capacidad de los modelos de aprendizaje profundo

para abordar el problema de la generación de coordenadas geoespaciales desde una perspectiva diferente.

Por otro lado, la investigación de (Moncla et al., 2014) proporciona una perspectiva única al abordar la generación automática de coordenadas geoespaciales a partir de texto mediante el uso de modelos basados en reglas y técnicas de geocodificación. Su estudio destaca la importancia de considerar tanto la semántica del texto como la estructura geoespacial para lograr resultados precisos en la generación de coordenadas. Modelos Sequence-to-Sequence, Word Embeddings y Redes Neuronales

Los modelos sequence-to-sequence (Seq2Seq) han sido ampliamente utilizados en tareas de PLN debido a su capacidad para capturar la estructura y el significado de secuencias de entrada y generar secuencias de salida coherentes. La arquitectura encoder-decoder subyacente en estos modelos permite aprender representaciones de alta calidad de texto y generar salidas contextualmente relevantes. Desde su introducción por (Sutskever et al., 2014), los modelos Seq2Seq han demostrado ser efectivos en una variedad de aplicaciones, incluida la traducción automática, la generación de texto y la síntesis de voz.

Los word embeddings son representaciones vectoriales de palabras en un espacio semántico continuo. Estas representaciones capturan la semántica y las relaciones entre las palabras, lo que las hace útiles en una variedad de tareas de PLN. (Mikolov et al., 2013) introdujeron los embeddings de palabras de distribución continua (word2vec), un enfoque popular que ha sido ampliamente adoptado en la comunidad de PLN. En su estudio, Mikolov et al. señalaron que “los embeddings de palabras proporcionan una forma eficaz de representar el significado semántico de las palabras en un espacio vectorial de baja dimensionalidad” (Mikolov et al., 2013, p. 1).

Además, las redes neuronales han demostrado ser útiles en la generación automática de coordenadas geoespaciales. Estas redes tienen la capacidad de aprender representaciones complejas de datos de entrada, lo que las hace adecuadas para tareas que implican la extracción de patrones de texto y la generación de secuencias de salida. En particular, las redes convolucionales han mostrado promesa en la extracción de características relevantes de texto para esta tarea, especialmente cuando se combinan con modelos sequence-to-sequence para capturar tanto la semántica como la estructura de las descripciones de ubicaciones.

6.1 Geoparsing

El geoparsing es una tarea relacionada que consiste en identificar y extraer información geográfica (como nombres de lugares, coordenadas geográficas, etc.) de texto no estructurado. Esta tarea es fundamental para diversas aplicaciones, como la navegación, la planificación urbana y la geolocalización en redes sociales. Investigaciones como la de (**li2019geoparsing**) Li et al. (2019) han abordado el problema del geoparsing utilizando enfoques basados en modelos de aprendizaje profundo, lo que demuestra la efectividad de estos métodos para la extracción de información geográfica precisa y relevante de grandes cantidades de texto no estructurado. Codificación Geoespacial con Geohash

La codificación geoespacial es esencial para representar ubicaciones geográficas de manera eficiente y precisa. Geohash, propuesto por (Niemeyer, 2008), es un método popular para codificar coordenadas geográficas en cadenas de caracteres alfanuméricos de longitud variable. La principal ventaja de Geohash es su capacidad para representar ubicaciones cercanas en el espacio con cadenas de caracteres similares, lo que facilita la indexación espacial y la búsqueda rápida de ubicaciones cercanas. Esta propiedad lo hace ideal para la generación de coordenadas geoespaciales a partir de contexto textual.

7. Metodología

7.1 Datos

El conjunto de datos descrito en el Capítulo Antecedentes contiene los nombres de las alcaldías, el código postal, el nombre de la colonia, la descripciones en español y sus coordenadas (x,y).

Tabla 1. *Ejemplo del contenido de la tabla de origen*

Alcaldía	**CP**	**Colonia**	**Descripción**	**x**	**y**
Coyoacán	4230	...	...	-99.1240131	19.3577091
Álvaro Obregón	1010	...	...	-99.198445	19.3575222
Cuajimalpa de Morelos	1330	...	...	-99.2668537	19.365973
Tláhuac	13360	...	...	-99.0294419	19.2966041

Ejemplo de descripción:

"SE SOLICITA EL APOYO PARA EL PERMISO PARA LA TALA DE DOS ARBOLES QUE SE ENCUENTRAN DENTRO DEL PREDIO PERTENECIENTE AL INSTITUTO MEXICANO DEL SEGURO SOCIAL, MISMOS QUE ESTAN DENTRO DE LA GUARDERIA INFANTIL N. 019, UBICADA EN AVENIDA ERMITA IZTAPALAPA, COL. PRADO CHURUBUSCO CP. 04230 ALCALDIA COYOACAN , YA QE LOS MISMOS TIENEN UNA ALTURA DE MAS DE 7 METROS Y ESTAN RECARGADOS EN LA BARDA DE LOS VESTIDORES Y COCINA DEL PREDIO, DONDE SE ENCUENTRA MUY CERCA EL TANQUE ESTACIONARIO."

Al conjunto de datos, se le realizó una limpieza para eliminar las filas con datos nulos en alguna de las columnas en la Tabla 1, dando como resultado un conjunto de 510,296 registros válidos, en donde las alcaldías se ven representadas de la siguiente manera:

Tabla 2. *Cantidad de registros por alcaldías dentro del conjunto de documentos*

Alcaldía	**No. Registros**
Cuauhtémoc	68940
Miguel Hidalgo	64184
Azcapotzalco	60348
Coyoacán	54820
Gustavo A. Madero	46896
Álvaro Obregón	43676
Xochimilco	41616
Tlalpan	30208
Iztacalco	28184
Iztapalapa	25368
Benito Juárez	20096
Venustiano Carranza	8696
Tláhuac	6724
Cuajimalpa de Morelos	4916
La Magdalena Contreras	4088
Milpa Alta	1536

7.1.1 Texto original

La columna "*Descripción*"de un conjunto de datos suele contener texto libre y variado, y cuando se usa como entrada para entrenar un modelo de red neuronal, puede presentar varios problemas como lo son los errores y faltas de ortografía, la diversidad de los datos, la longitud de las descripciones, la falta de representatividad, el ruido y la complejidad adicional, y la necesidad de contener direcciones estándar y bien estructuradas.

Errores y Faltas de Ortografía

Las descripciones en texto libre son propensas a contener una variedad de errores y faltas de ortografía. Estos errores pueden ser errores tipográficos, como letras intercambiadas o duplicadas (“necesitamso” en lugar de “necesitamos”); errores de acentuación, como la omisión de tildes o uso incorrecto de las mismas (“aun” en lugar de “aún”); y errores gramaticales, como el uso incorrecto

de palabras o estructuras gramaticales (“yo he hecho” en lugar de “yo he hacer”). Estos errores pueden causar confusión en el modelo, dificultando su capacidad para aprender patrones relevantes, por ejemplo:

Ejemplo sin faltas de ortografía:

- Entrada: “La dirección es Paseo de la Reforma 123, Colonia Juárez, Ciudad de México.”
- Salida:
  - “Paseo de la Reforma 123” (Dirección)
  - “Colonia Juárez” (Localidad)
  - “Ciudad de México” (Ciudad)

Ejemplo con faltas de ortografía:

- Entrada: “La dirreccion es Paséo de la Rephorma 123, Colonia Juaréz, Ciudadd de Méjico.”
- Salida:
  - “Paséo de la Rephorma 123” (Indeterminada)
  - “Colonia Juaréz” (Indeterminada)
  - “Ciudadd de Méjico” (Indeterminada)

Impacto de las faltas de ortografía:

- Paseo de la Reforma 123:
  - Correcto: El sistema reconoce esta dirección como una entidad coherente.
  - Incorrecto: La palabra “Paséo” y “Rephorma” tienen faltas de ortografía, lo que puede impedir que el sistema las reconozca correctamente como parte de una dirección conocida.
- Colonia Juárez:
  - Correcto: El sistema identifica “Colonia Juárez” como una localidad en la Ciudad de México.
  - Incorrecto: La palabra “Juaréz” con una tilde incorrecta puede llevar a la entidad no ser reconocida correctamente.
- Ciudad de México:
  - Correcto: La capital del país es reconocida sin problemas.

- Incorrecto: “Ciudadd de Méjico” tiene faltas de ortografía (“Ciudadd” con doble “d” y “Méjico” en lugar de “México”), lo que puede confundir al sistema y resultar en una identificación fallida o incorrecta.

En general, las faltas de ortografía introducen ruido en los datos de entrenamiento y prueba, lo que puede reducir la precisión de los modelos de aprendizaje automático, los cuales necesitan datos limpios y consistentes para funcionar de manera óptima, y la presencia de estos errores reduce la calidad de los datos.

Diversidad de los Datos

La columna “Descripción” puede contener una amplia gama de temas y contextos, ya que cada descripción es única y puede abordar diferentes aspectos de una situación. Esta diversidad incluye temas diversos, desde quejas sobre infraestructura hasta comentarios sobre seguridad; tonos y estilos diferentes, desde formal hasta coloquial, y en distintos estilos, desde detallado hasta resumido; y lenguaje regional, con diferentes regiones utilizando jerga o expresiones únicas que no son comprendidas universalmente. La gran diversidad en los datos puede dificultar la capacidad del modelo para generalizar y encontrar patrones consistentes, ya que el modelo puede verse abrumado por la variedad de información no estructurada por ejemplo:

Dirección en formato estándar:

- **Entrada:**

  “Paseo de la Reforma 123, Colonia Juárez, Ciudad de México, CDMX.”
- **Salida esperada:**
  - “Paseo de la Reforma 123” (*Dirección*)
  - “Colonia Juárez” (*Localidad*)
  - “Ciudad de México” (*Ciudad*)
  - “CDMX” (*Entidad Federativa*)

Dirección abreviada:

- **Entrada:**

  “Reforma 123, Col. Juárez, CDMX.”
- **Salida esperada:**
  - “Reforma 123” (*Dirección*)

- "Col. Juárez" (*Localidad*)
- "CDMX" (*Entidad Federativa*)

Dirección completa y detallada:

- **Entrada:**

  "Paseo de la Reforma número 123, Colonia Juárez, Alcaldía Cuauhtémoc, Ciudad de México, CDMX, México."

- **Salida esperada:**
  - "Paseo de la Reforma número 123" (*Dirección*)
  - "Colonia Juárez" (*Localidad*)
  - "Alcaldía Cuauhtémoc" (*Delegación*)
  - "Ciudad de México" (*Ciudad*)
  - "CDMX" (*Entidad Federativa*)
  - "México" (*País*)

Dirección parcial y menos detallada:

- **Entrada:**

  "Reforma 123, Juárez, CDMX."

- **Salida esperada:**
  - "Reforma 123" (*Dirección*)
  - "Juárez" (*Localidad*)
  - "CDMX" (*Entidad Federativa*)

Dirección con sinónimos y variantes:

- **Entrada:**

  "Paseo de la Reforma, número 123, Col. Juárez, Ciudad de México."

- **Salida esperada:**
  - "Paseo de la Reforma, número 123" (*Dirección*)
  - "Col. Juárez" (*Localidad*)
  - "Ciudad de México" (*Ciudad*)

Dirección con diferentes formas de nombrar las calles:

- **Entrada:**

  "Reforma 123, Juárez, Ciudad de México, D.F."

- **Salida esperada:**
  - “Reforma 123” (*Dirección*)
  - “Juárez"(*Localidad*)
  - “Ciudad de México, D.F.” (*Ciudad*)

Falta de Representatividad

Los textos de la columna “Descripción” puede no ser representativa de todos los aspectos relevantes necesarios para predecir alcaldías. Algunas descripciones pueden centrarse en problemas específicos de ciertas áreas, mientras que otras pueden ser más generales. Esta falta de representatividad puede llevar a un sesgo en el modelo, haciendo que sea menos preciso al predecir resultados para áreas no adecuadamente representadas en las descripciones, por ejemplo:

Predicción con Datos Representativos:

- **Entrada:** “Avenida Insurgentes tiene mucho tráfico y contaminación.”
- **Salida esperada:** Cuauhtémoc
- **Salida del modelo:** Cuauhtémoc (Precisa)

Predicción con Datos No Representativos:

- **Entrada:** “Alta inseguridad en algunas áreas.”
- **Salida esperada:** Tlalpan
- **Salida del modelo:** Indeterminada o Incorrecta (Iztapalapa, Cuauhtémoc, etc.)

Ruido y Complejidad Adicional

Las descripciones en texto libre pueden contener mucho ruido, como información irrelevante, redundante o contradictoria. Este ruido añade complejidad al modelo, haciendo que sea más difícil para la red neuronal extraer patrones útiles. Por ejemplo, múltiples descripciones de la misma situación pueden tener variaciones significativas en el lenguaje y los detalles, aumentando la complejidad del entrenamiento del modelo.

Descripción 1:

- **Descripción:** “Hay un poste de luz a punto de caerse en la Avenida Insurgentes, colonia Roma. Parece que se va a caer pronto y es peligroso.”
- **Alcaldía:** Cuauhtémoc

Descripción 2:

- **Descripción:** “Poste de luz inclinado en Insurgentes, cerca de la Roma. Está en muy mal estado y puede caer en cualquier momento.”
- **Alcaldía:** Cuauhtémoc

Descripción 3:

- **Descripción:** “En la Avenida Insurgentes, colonia Roma, hay un poste de luz que se está cayendo. Es un gran riesgo para los peatones.”
- **Alcaldía:** Cuauhtémoc

Las descripciones contienen información redundante y variaciones en el lenguaje, lo cual introduce ruido. Este ruido puede dificultar que el modelo identifique patrones consistentes:

- Información redundante: Todas las descripciones mencionan el poste de luz en Insurgentes y la colonia Roma, pero de diferentes maneras.
- Información irrelevante: Detalles adicionales como “es peligroso” o “gran riesgo para los peatones” pueden no ser relevantes para identificar la alcaldía.
- Variaciones en el lenguaje: Diferentes formas de describir el mismo problema (“a punto de caerse", “inclinado”, “se está cayendo”).

Longitud de las Descripciones

Las descripciones pueden variar significativamente en longitud, desde unas pocas palabras hasta párrafos completos. Esta variabilidad en la longitud presenta varios desafíos. Las descripciones muy cortas pueden no proporcionar suficiente información para que el modelo haga una predicción precisa, mientras que las descripciones muy largas pueden contener información irrelevante que puede confundir al modelo. Además, las descripciones largas pueden requerir más recursos computacionales para procesar. La inconsistencia en la longitud de las descripciones puede llevar a que el modelo no tenga una representación uniforme de los datos, lo que afecta negativamente su capacidad para aprender de manera efectiva.

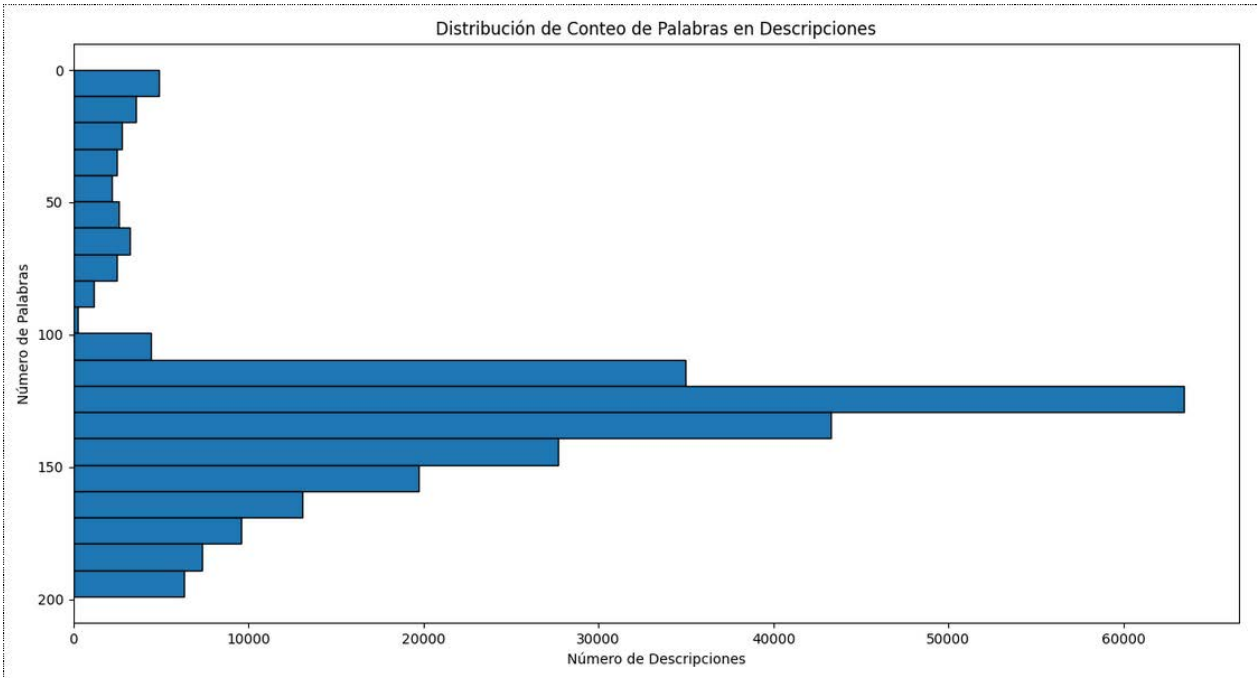

Figura 3. *Histograma de la distribución del número de palabras en las descripciones del conjunto de datos*

Direcciones Estándar y Bien Estructuradas

Para mejorar la utilidad de las descripciones como entrada para un modelo de red neuronal, es crucial que las descripciones contengan direcciones estándar y bien estructuradas, como las que se encuentran comúnmente en la Ciudad de México. Direcciones claras y consistentes, que sigan un formato establecido, ayudan al modelo a identificar patrones geográficos y espaciales de manera más eficiente. Sin embargo, en muchos casos, las descripciones pueden contener direcciones en formatos no estándar, ambiguos o incluso no contener información alguna que se acerque a una dirección, lo que dificulta el procesamiento y análisis por parte del modelo.

Formato Estándar

- **Descripción:** "Hay un poste de luz a punto de caerse en Avenida Insurgentes Sur 123, colonia Roma Norte, alcaldía Cuauhtémoc, Ciudad de México, C.P. 06700."
- **Comentarios:** La dirección es clara y sigue un formato establecido, incluyendo calle, número, colonia, alcaldía y código postal.

Formato Ambiguo

- **Descripción:** "Un poste de luz está inclinado en Insurgentes, Roma Norte, Cuauhtémoc."

- **Comentarios:** La dirección es menos específica, omitiendo el número de calle y el código postal, lo que puede dificultar la localización precisa del problema.

Formato Nulo

- **Descripción:** "Hay un poste de luz que se va a caer cerca de mi casa."
- **Comentarios:** La descripción no contiene información útil sobre la ubicación, lo que hace imposible identificar el lugar exacto del problema.

Impacto en el Modelo

La calidad de las descripciones afecta significativamente la capacidad del modelo para identificar patrones y realizar predicciones precisas:

- **Formato Estándar:** Facilita la identificación precisa de la ubicación, mejorando la eficiencia del modelo.
- **Formato Ambiguo:** Introduce incertidumbre y puede llevar a errores en la identificación de la ubicación.
- **Formato Nulo:** Proporciona poca o ninguna información útil, lo que impide al modelo realizar predicciones precisas.

7.1.2 Propuesta NER

Los resultados observados en los apartados Sección 8.1 y Sección 8.2, nos muestran que utilizar el texto completo, no produce un buen resultado, por lo cual, realizamos una propuesta de texto de entrada basada en el Reconocimiento de Entidades Nombradas (NER) para mejorar el desempeño de modelos de predicción propuestos.

Para mejorar la calidad del input para los modelos de predicción de alcaldías en Ciudad de México, se realizó la comparación de diferentes modelos de reconocimiento de entidades nombradas (NER) y la evaluación de sus rendimientos y su capacidad para extraer información estructurada de textos no estructurados, enfocándose en nombres de personas (PER), lugares (LOC), organizaciones (ORG) y elementos misceláneos (MISC). Esta estrategia es crucial en Ciudad de México, donde las calles suelen llevar nombres de personas y/o lugares. El NER permite identificar y clasificar correctamente las entidades, reduciendo la diversidad de datos, el ruido y la complejidad del input, mejorando la precisión en la extracción de información de direcciones.

Importación de Bibliotecas y Configuración Inicial

En el análisis se utilizaron bibliotecas esenciales: pandas, spaCy con sus modelos (SM, MD, LG) y Flair, las cuales son librerías que permiten ejecutar modelos entrenados para el reconocimiento de entidades nombradas y son fundamentales para la manipulación de datos en el procesamiento del lenguaje natural.

Para evaluar el desempeño de cada uno de los modelos NER, se utilizó un conjunto de 100 datos que contienen textos etiquetados manualmente con entidades nombradas. Estos datos se estructuran en un DataFrame de pandas para facilitar su manipulación.

Texto de entrada

"SE SOLICITA EL APOYO PARA EL PERMISO PARA LA TALA DE DOS ARBOLES QUE SE ENCUENTRAN DENTRO DEL PREDIO PERTENECIENTE AL INSTITUTO MEXICANO DEL SEGURO SOCIAL, MISMOS QUE ESTAN DENTRO DE LA GUARDERIA INFANTIL N. 019, UBICADA EN AVENIDA ERMITA IZTAPALAPA, COL. PRADO CHURUBUSCO CP. 04230 ALCALDIA COYOACAN , YA QE LOS MISMOS TIENEN UNA ALTURA DE MAS DE 7 METROS Y ESTAN RECARGADOS EN LA BARDA DE LOS VESTIDORES Y COCINA DEL PREDIO, DONDE SE ENCUENTRA MUY CERCA EL TANQUE ESTACIÓNARIO."

Entidades etiquedatas por un humano:

- INSTITUTO MEXICANO DEL SEGURO SOCIAL: **ORG**
- AVENIDA ERMITA IZTAPALAPA: **LOC**
- PRADO CHURUBUSCO: **LOC**
- COYOACAN: **LOC**

Se ejecutan los modelos de NER, para generar predicciones de entidades nombradas en los datos. Cada modelo procesa los textos y produce un conjunto de entidades predichas, que luego se comparan con las etiquetas reales para evaluar su precisión y efectividad en la tarea de identificar alcaldías específicas de la Ciudad de México.

Evaluación de Resultados y Métricas Utilizadas

Para evaluar el rendimiento de los modelos, se emplearon las métricas clave:

- **Precisión**: Mide la proporción de entidades correctamente identificadas entre todas las entidades predichas. Es crucial para asegurar que las predicciones hechas por el modelo sean mayormente correctas.
- **Recall**: Mide la proporción de entidades correctamente identificadas entre todas las entidades reales. Es importante para evaluar la capacidad del modelo de identificar todas las entidades relevantes en los textos.
- **F1-score**: Es la media armónica de la precisión y el recall, proporcionando un balance entre ambas métricas. Es especialmente útil cuando se necesita un equilibrio entre la precisión y el recall en el modelo.

Se definió una función que muestra las entidades reales frente a las predichas por cada modelo y se graficó la comparación entre el número de entidades reales y predichas por cada modelo, utilizando matplotlib.

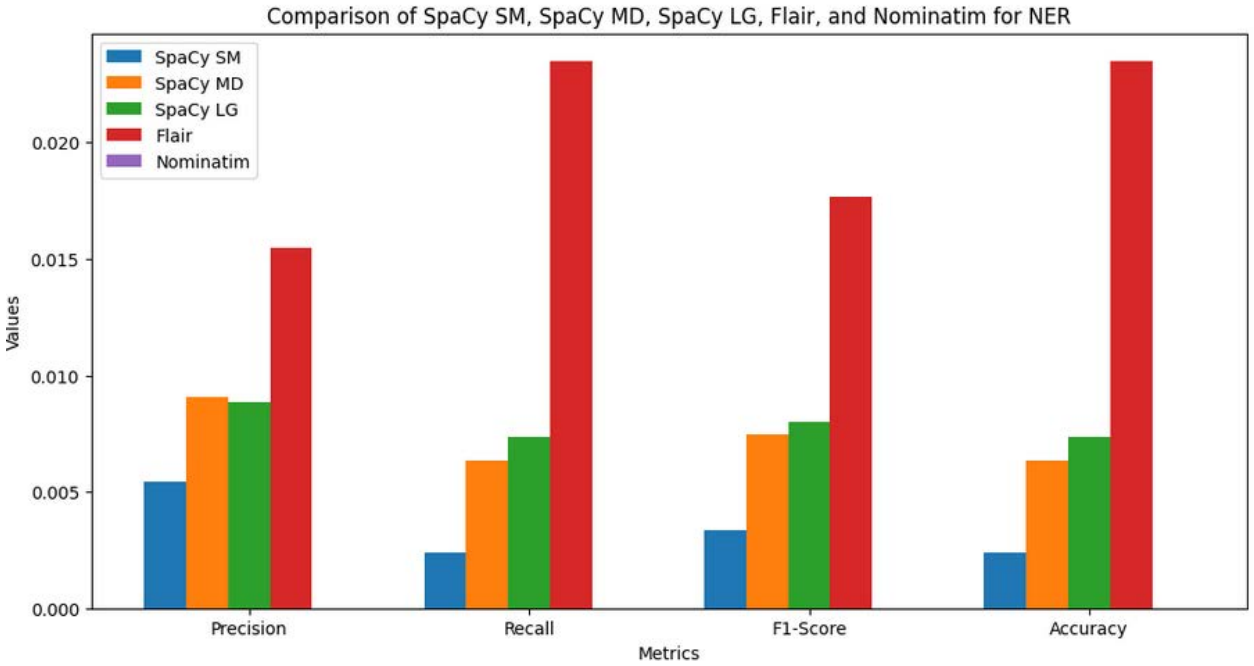

Figura 4. *Histograma del desempeño de distintos extractores de entidades nombradas.*

Las gráficas de barras en la Figura 4 proporcionan una evaluación visual clara de la precisión y eficacia de cada modelo, facilitando la identificación de los más adecuados para la tarea de NER en el contexto de la Ciudad de México.

Ejemplo de desempeño de los modelos NER

Se evaluó el desempeño de varios modelos NER comparando sus resultados con una referencia humana. Los modelos considerados fueron spaCy en sus versiones pequeña (SM), mediana (MD) y grande (LG), además del modelo Flair. La Tabla 3 resume las clasificaciones de diversas palabras según estos modelos y un evaluador humano.

Se identificaron varias tendencias en los resultados, la entidad "instituto mexicano del seguro social" fue correctamente clasificada como organización (ORG) por todos los modelos, excepto por spaCy LG, que la identificó incorrectamente como persona (PER). Por otro lado, "avenida ermita iztapalapa" fue consistentemente reconocida como localización (LOC) por todos los modelos, reflejando una concordancia completa con la clasificación humana.

Sin embargo, hubo casos donde los modelos fallaron en reconocer entidades correctamente. Por ejemplo, "prado churubusco" fue identificado correctamente como localización solo por el modelo Flair, mientras que los otros modelos no lograron clasificarlo. La entidad "coyoacan" también mostró variabilidad en las clasificaciones: fue correctamente identificada como localización por el humano y Flair, sin embargo spaCy MD la clasificó erróneamente como persona, y otros modelos no la reconocieron.

El análisis reveló errores recurrentes en la clasificación de ciertas palabras. Términos como "se solicita", "apoyo", "permiso", "tala", y otros fueron clasificados incorrectamente como organizaciones (ORG) por los modelos spaCy en diferentes versiones. Esto indica una tendencia de estos modelos a sobreclasificar ciertas palabras como entidades organizacionales, independientemente del contexto.

La precisión variada entre los modelos destaca la importancia de evaluar múltiples enfoques para tareas de NER. Aunque modelos como Flair y spaCy SM mostraron mayor precisión en algunas entidades, otros modelos como spaCy LG presentaron más errores en la clasificación.

Tabla 3. *Comparación del desempeño de modelos de Reconocimiento de Entidades Nombradas (NER) en relación con la clasificación humana.*

Entidad encontrada	**Humano**	**spaCy SM**	**spaCy MD**	**spaCy LG**	**Flair**
instituto mexicano del seguro social	ORG	ORG	ORG	PER	ORG
avenida ermita iztapalapa	LOC	LOC	LOC	LOC	LOC
prado churubusco	LOC	–	–	–	LOC
coyoacan	LOC	–	PER	–	LOC
se solicita	–	ORG	ORG	ORG	–
apoyo	–	ORG	–	–	–
permiso	–	ORG	–	–	–
tala	–	ORG	–	–	–
dos	–	MISC	–	MISC	–
se	–	MISC	–	–	–
predio	–	ORG	ORG	ORG	–
perteneciente	–	ORG	MISC	MISC	–
instituto mexicano	–	ORG	–	–	–
avenida ermita	–	LOC	–	LOC	–
col	–	ORG	ORG	ORG	–
alcaldia coyoacan	–	MISC	PER	MISC	–
altura	–	MISC	–	–	–
estan recargados	–	ORG	ORG	MISC	–
vestidores	–	MISC	–	–	–
tanque estacionario	–	MISC	LOC	MISC	–
se solicita el apoyo para el permiso para la tala de dos	–	–	ORG	ORG	–
se encuentran dentro del predio	–	–	ORG	–	–
guarderia infantil	–	–	ORG	–	LOC
avenida	–	–	LOC	LOC	–
churubusco	–	–	ORG	–	–
donde se encuentra muy cerca el	–	–	ORG	–	–
estacionario	–	–	LOC	–	–

Resultado final

El texto de entrada se generó utilizando el modelo de SpaCy y Flair, procesando las entidades encontradas dejando únicamente las que tienen un umbral de confianza arriba del 0.8, para combinar las entidades extraídas por ambos modelos, se implementó un algoritmo que prioriza las entidades identificadas por Flair. Esto se debe a que Flair utiliza embeddings contextuales, lo que generalmente resulta en una mayor precisión en la identificación de entidades.

Las entidades se almacenaron en una lista, asegurando que no haya duplicados y que se mantenga la claridad del texto final, se eliminó cualquier duplicado de palabras en el texto final consolidado, resultando en una cadena de texto clara y sin redundancias. Este proceso asegura que el resultado final sea fácil de interpretar y utilizable en aplicaciones posteriores.

El proceso descrito resultó en la extracción de las siguientes entidades nombradas del texto de entrada, presentadas de manera consolidada y libre de duplicados:

"INSTITUTO MEXICANO SEGURO SOCIAL AVENIDA ERMITA IZTAPALAPA PRADO CHURUBUSCO COYOACAN CP ALCALDIA INFANTIL COCINA"

El resultado final del procesamiento de texto elimina redundancias y errores, presentando un texto de una manera clara, uniforme y con una longitud menor permitiéndole concentrarse en las características clave de las direcciones y ubicaciones, en lugar de verse abrumado por detalles innecesarios. Esto facilita que el modelo de red neuronal aprenda patrones consistentes en las direcciones, mejorando su capacidad para generalizar y predecir con precisión en nuevos datos.

7.2 Preprocesamiento de datos

Se preprocesaron las descripciones mediante limpieza de texto, para preparar datos de texto crudos para análisis utilizando técnicas de procesamiento y tokenización. Este proceso es crucial en aplicaciones como análisis de sentimientos, clasificación de textos y generación de texto.

El procesamiento de texto comienza con la limpieza y normalización de los datos. Nuestro objetivo es transformar descripciones de texto variadas en secuencias de enteros que representen las palabras más relevantes en nuestro conjunto de datos.

Utilizamos la biblioteca *re* de Python para eliminar caracteres especiales y convertir todas las letras a minúsculas. Esto asegura consistencia en el tratamiento de las palabras, ignorando signos de puntuación y otros caracteres que no son letras o espacios.

Para representar el texto como secuencias de enteros, creamos un vocabulario V que consiste en las 8,000 (*MAX_WORDS*) palabras más frecuentes en nuestro corpus. Cada palabra en las descripciones limpias se tokeniza y se asigna a un número entero único según su posición en este vocabulario. Las palabras que no están en el vocabulario se marcan con un token especial (0 en este caso).

Cada descripción de texto se transforma en una secuencia de enteros utilizando el vocabulario creado anteriormente. Sea $D = \{d_1, d_2, \ldots, d_N\}$ el conjunto de descripciones limpias, y texto_a_secuencia(d_i, V) la función que convierte la descripción d_i en una secuencia de enteros utilizando el vocabulario V. Esto nos permite representar cada descripción como una lista ordenada de números enteros $S_i =$ texto_a_secuencia(d_i, V), facilitando el análisis subsiguiente.

Al final del proceso, validamos nuestros resultados imprimiendo las descripciones originales, las descripciones limpias, el vocabulario generado y las secuencias de enteros correspondientes a cada descripción. Esto asegura que el proceso de preprocesamiento se haya realizado correctamente y que los datos estén listos para ser utilizados en modelos de aprendizaje automático.

Ejemplo texto original limpio

> "*instituto mexicano seguro social avenida ermita iztapalapa prado churubusco coyoacan solicita apoyo permiso tala dos arboles encuentran dentro perteneciente mismos estan guarderia infantil ubicada col qe altura mas 7 metros barda vestidores cocina predio encuentra tanque estaciónario.*"

Ejemplo texto de propuesta NER limpio

"*instituto mexicano seguro social avenida ermita iztapalapa prado churubusco coyoacan cp alcaldia infantil cocina. "*

Ejemplo texto secuenciado

[650, 1198, 168, 125, 21, 900, 64, 887, 282, 103, 12, 43, 272, 1007, 66, 246, 218, 251, 4256, 1676, 388, 5628, 1256, 44, 6, 478, 143, 234, 186, 1207, 2715, 361, 47, 1340, 5953, 555, 703, 287, 293, 744, 1952, 384, 82, 713, 245, 517]

Ejemplo texto secuenciado de propuesta NER

[650, 1198, 168, 125, 21, 900, 64, 887, 282, 103, 19, 433, 361, 744]

7.3 Modelo Multi Layer Perceptron (MLP)

El primer modelo implementado utiliza una arquitectura MLP. La estructura del modelo se compone de varias capas apiladas de manera secuencial, comenzando con una capa de incrustación (*Embedding layer*) que transforma las palabras en vectores densos de longitud fija. Específicamente, la capa de incrustación convierte las palabras en vectores de 16 dimensiones, permitiendo que las palabras con significados similares tengan representaciones similares. El tamaño del vocabulario es determinado por *MAX_WORDS* y la longitud de las secuencias de entrada es especificada por *input_limit*.

La siguiente capa en el modelo es una capa de aplanamiento (*Flatten layer*), la cual convierte la matriz bidimensional resultante de la capa de incrustación en un vector unidimensional. Esto es necesario para que la salida pueda ser procesada por las capas densas subsiguientes.

Posteriormente, se incluye una capa densa (*Dense layer*) con 100 neuronas y una función de activación ReLU (*Rectified Linear Unit*). Esta capa también incorpora una regularización L2 con un factor de penalización de 0.001, lo que ayuda a prevenir el sobreajuste al modelo. Además, se añade una capa de *Dropout* con una tasa de 0.2, desactivando aleatoriamente el 20 % de las neuronas durante el entrenamiento para mejorar la generalización del modelo.

La capa final es una capa de salida densa con tantas neuronas como clases haya en el problema de clasificación, utilizando la activación *softmax*. Esta función de activación convierte los valores de salida en probabilidades, facilitando la interpretación de los resultados de clasificación.

Para entrenar el modelo, se utiliza el optimizador Adam, conocido por su eficiencia en la adaptación de los pesos del modelo durante el entrenamiento. La métrica de precisión (*accuracy*) se emplea para evaluar el rendimiento del modelo, y la función de pérdida utilizada es la entropía cruzada categórica (*categorical crossentropy*), adecuada para problemas de clasificación multiclase.

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 embedding (Embedding)       (None, 500, 16)           128000

 flatten (Flatten)           (None, 8000)              0

 dense (Dense)               (None, 100)               800100

 dense_1 (Dense)             (None, 16)                1616

=================================================================
Total params: 929,716
Trainable params: 929,716
Non-trainable params: 0
```

Figura 5. *Modelo Multi Layer Perceptron.*

La Figura 5 muestra una visión general de las capas, la forma de las entradas y salidas de cada capa, y el número total de parámetros que serán entrenados. Este modelo es particularmente adecuado para problemas de clasificación de texto, convirtiendo texto en secuencias de índices de palabras que son representados mediante *embeddings* y luego clasificados utilizando una red neuronal.

7.4 Modelo Convolutional Neural Network (CNN)

El segundo modelo implementado utiliza una arquitectura CNN. El modelo incluye varias capas diseñadas para capturar y procesar características complejas de las secuencias de entrada. La primera capa es una capa de incrustación (*Embedding layer*), que transforma las palabras en vectores densos de 256 dimensiones. Esta capa permite que las palabras con significados similares tengan representaciones similares. El tamaño del vocabulario está determinado por *MAX_WORDS* y la longitud de las secuencias de entrada es de 500 palabras.

El modelo incluye varias capas convolucionales (*Conv1D layers*) con regularización L2, diseñadas para extraer características locales de las secuencias de texto. La primera capa convolucional tiene 256 filtros con un tamaño de kernel de 2, seguido de una capa de *MaxPooling* con un tamaño

de pool de 2. Esto se repite con capas convolucionales adicionales que tienen 128, 64 y 32 filtros respectivamente, cada una seguida por una capa de *MaxPooling*. Todas las capas convolucionales utilizan la función de activación ReLU y la regularización L2 con un factor de penalización de 0.001, ayudando a prevenir el sobreajuste al modelo.

Después de las capas convolucionales, se añade una capa de aplanamiento (*Flatten layer*) que convierte la salida 3D de las capas anteriores en un vector unidimensional, para poder ser procesado por las capas densas subsiguientes.

A continuación, el modelo incluye una capa densa (*Dense layer*) con 100 neuronas y una función de activación ReLU, también con regularización L2. Finalmente, la capa de salida es una capa densa con tantas neuronas como clases haya en el problema de clasificación, utilizando la activación *softmax*.

Para entrenar el modelo, se utiliza el optimizador Adam, conocido por su eficiencia en la adaptación de los pesos del modelo durante el entrenamiento. La métrica de precisión (*accuracy*) se emplea para evaluar el rendimiento del modelo, y la función de pérdida utilizada es la entropía cruzada categórica (*categorical crossentropy*), adecuada para problemas de clasificación multiclase.

El modelo en la Figura 6 de muestra una visión general de las capas, la forma de las entradas y salidas de cada capa, y el número total de parámetros que serán entrenados. Este modelo es particularmente adecuado para problemas de clasificación de texto, donde las secuencias de palabras se procesan mediante capas convolucionales para capturar patrones locales, seguidas por capas densas para la clasificación final.

```
Model: "sequential_3"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 embedding_3 (Embedding)     (None, 500, 256)          2048000

 conv1d_4 (Conv1D)           (None, 500, 256)          131328

 max_pooling1d_4 (MaxPoolin  (None, 250, 256)          0
 g1D)

 conv1d_5 (Conv1D)           (None, 250, 128)          65664

 max_pooling1d_5 (MaxPoolin  (None, 125, 128)          0
 g1D)

 conv1d_6 (Conv1D)           (None, 125, 64)           16448

 max_pooling1d_6 (MaxPoolin  (None, 62, 64)            0
 g1D)

 conv1d_7 (Conv1D)           (None, 62, 32)            4128

 max_pooling1d_7 (MaxPoolin  (None, 31, 32)            0
 g1D)

 flatten_1 (Flatten)         (None, 992)               0

 dense_2 (Dense)             (None, 100)               99300

 dense_3 (Dense)             (None, 16)                1616

=================================================================
Total params: 2366484 (9.03 MB)
Trainable params: 2366484 (9.03 MB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
```

Figura 6. *Modelo Convolutional Neural Network.*

7.5 Entrenamiento y Evaluación

Debido a que la mayoría de los algoritmos y modelos de aprendizaje automático, incluidas las redes neuronales, requieren que las etiquetas (o clases) de los datos de entrenamiento estén representadas numéricamente en lugar de texto o categorías para que los cálculos y las operaciones matemáticas involucradas en el entrenamiento del modelo funcionen de manera optima, se creó un diccionario llamado *label_dict*.

Se generó un algoritmo que asignó a cada alcaldía única en la lista *Alcaldía* un número entero único. Se utilizó un conjunto para obtener todos las alcaldías únicas, las cuales fueron ordenadas alfabéticamente. Luego, utilizó *enumerate* para asignar a cada alcaldía un índice numérico. Des-

pués, imprimió este diccionario para mostrar la correspondencia entre las alcaldías y sus números asignados.

Tabla 4. *Alcaldías de la Ciudad de México enumeradas del 1 al 16 por orden alfabético.*

Alcaldía	**alcaldiaN**
Álvaro Obregón	1
Azcapotzalco	2
Benito Juárez	3
Coyoacán	4
Cuajimalpa de Morelos	5
Cuauhtémoc	6
Gustavo A. Madero	7
Iztacalco	8
Iztapalapa	9
Magdalena Contreras	10
Miguel Hidalgo	11
Milpa Alta	12
Tláhuac	13
Tlalpan	14
Venustiano Carranza	15
Xochimilco	16

Finalmente, se agrega una nueva columna 'alcaldiaN' al DataFrame, que contiene los valores numéricos. Esto permite representar los datos categóricos de las alcaldías con valores numéricos, facilitando análisis adicionales o la aplicación de algoritmos de aprendizaje automático.

Se utilizó la biblioteca *scikit-learn* para dividir un conjunto de datos en conjuntos de entrenamiento y prueba, utilizando la función *train_test_split* toma dos argumentos principales: las características *Descripcion* y la variable objetivo *alcaldiaN*. La primera serie se asigna a X y la segunda a y. Al especificar *test_size=0.2*, indicamos que el 20 % de los datos se reservarán como conjunto de prueba para evaluar el modelo posteriormente.

Una vez ejecutada la función, se generan cuatro conjuntos de datos separados:

- *X_train*: Contiene las características que se utilizarán para entrenar el modelo.
- *X_test*: Contiene las características que se utilizarán para evaluar el modelo después de entrenarlo.
- *y_train*: Contiene las etiquetas correspondientes a *X_train*, es decir, los valores de *alcaldiaN* utilizados durante el entrenamiento.
- *y_test*: Contiene las etiquetas correspondientes a *X_test*, utilizadas para evaluar qué tan bien el modelo puede predecir *alcaldiaN* en datos no vistos.

Por último se utilizó *train_test_split* para dividir el conjunto de prueba (*X_test* y *y_test*) en dos partes: un conjunto de validación (*X_valid* y *y_valid*) y un conjunto de prueba más pequeño, estableciendo un *test_size=0.5* que especifica que el 50 % de los datos originales de prueba se asignarán al conjunto de validación, mientras que el restante 50 % permanecerá como el nuevo conjunto de prueba como se observa en la Tabla 5.

Tabla 5. *División de los datos para entrenamiento, pruebas y validación*

X_train	**X_test**	**X_valid**
408,236	51,030	51,030

Estos conjuntos de datos divididos permiten que los modelos de aprendizaje automático sean entrenados con una parte de los datos y evaluados con otra, lo que es crucial para estimar la precisión y la capacidad de generalización del modelo antes de aplicarlo en datos nuevos o desconocidos.

Ambos modelos se entrenaron utilizando un conjunto de datos dividido en entrenamiento, validación y prueba. Se empleó una devolución de llamada ModelCheckpoint para guardar el mejor modelo basado en la precisión de validación. Los modelos se evaluaron en el conjunto de datos de prueba para determinar su rendimiento final.

8. Resultados y Discusión

Los resultados experimentales demostraron que ambos modelos, MLP y CNN, lograron resultados de clasificación precisos utilizando un texto de entrada preprocesado. El modelo MLP presentó un rendimiento superior al del modelo CNN, lo que sugiere la efectividad de la arquitectura MLP para capturar patrones locales en los datos de texto.

8.1 Resultados CNN con el texto original

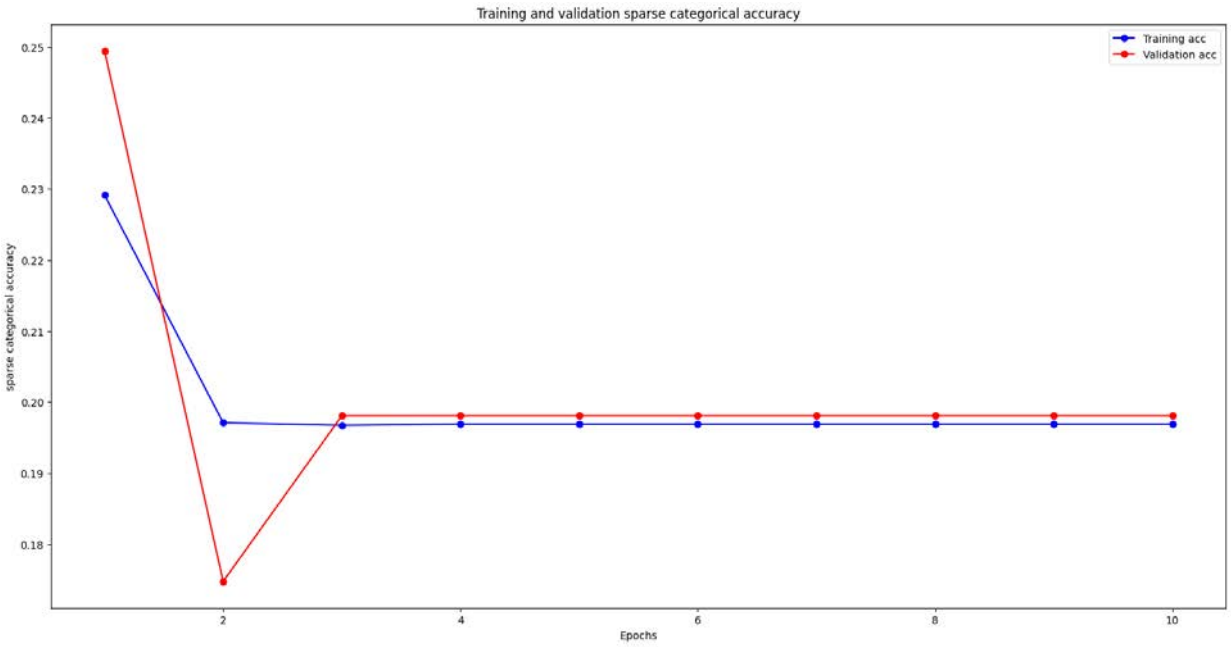

Figura 7. *Métricas de exactitud en el modelo CNN utilizando texto original.*

La imagen Figura 7 muestra una gráfica de la precisión categórica escasa durante el entrenamiento y la validación a lo largo de 10 épocas.

Observaciones Detalladas:

- **Época 1:**
  - *Training Accuracy*: 0.23
  - *Validation Accuracy*: 0.25
  - Inicialmente, el modelo parece tener una precisión razonablemente alta en ambas curvas, pero esto podría ser indicativo de sobreajuste temprano.

- **Época 2:**
  - Ambas precisiones caen drásticamente.
  - *Training Accuracy*: 0.20
  - *Validation Accuracy*: 0.19
  - Esta caída drástica podría sugerir que el modelo está ajustándose a ruidos o peculiaridades en los datos de entrenamiento y no generaliza bien.
- **Épocas 3-10:**
  - Las curvas se estabilizan y convergen alrededor de 0.20 para el entrenamiento y la validación.
  - Esto indica que el modelo no está aprendiendo significativamente más allá de la segunda época y se encuentra atrapado en un punto bajo de rendimiento.

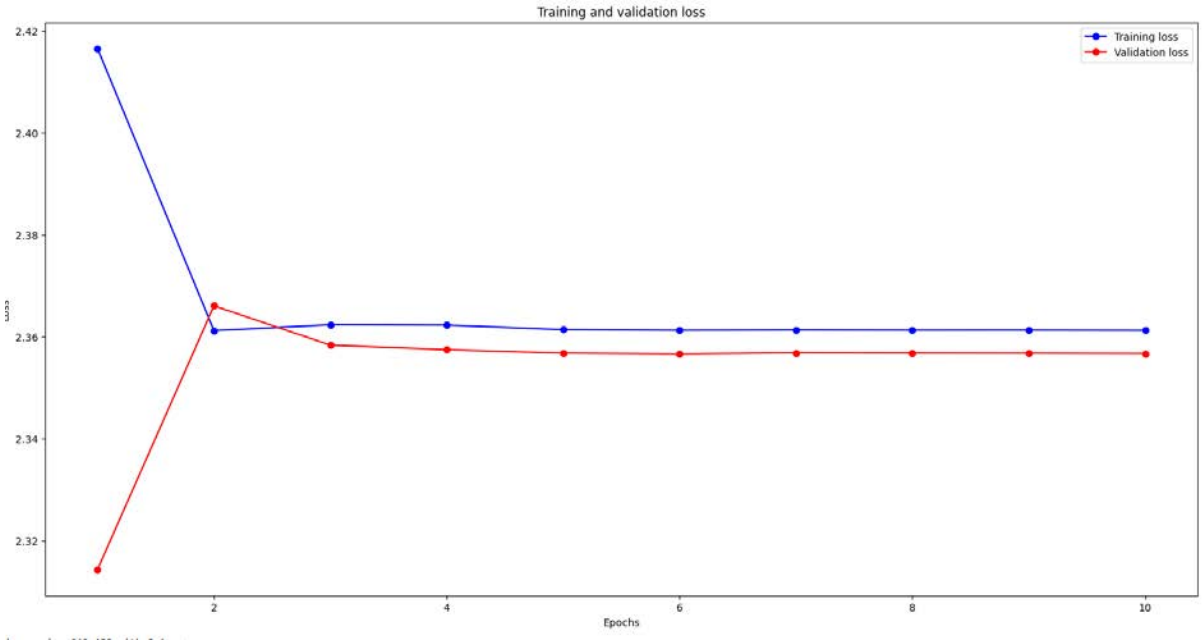

Figura 8. *Métricas de la función de perdida en el modelo CNN texto original.*

La gráfica Figura 8 muestra la pérdida (*loss*) de entrenamiento y validación a lo largo de 10 épocas. En el eje y se representa la pérdida (*loss*) y en el eje x se representan las épocas.

Observaciones Detalladas:

- **Época 1:**
  - La pérdida de entrenamiento (línea azul) comienza en aproximadamente 2.42.
  - La pérdida de validación (línea roja) comienza en aproximadamente 2.32.
- **Época 2:**
  - La pérdida de entrenamiento disminuye abruptamente a cerca de 2.36.
  - La pérdida de validación aumenta hasta cerca de 2.36, cruzando con la pérdida de entrenamiento.
- **Época 3 en adelante:**
  - Ambas pérdidas se estabilizan alrededor de 2.36 para la pérdida de entrenamiento y un poco por debajo para la pérdida de validación, sin cambios significativos en las siguientes épocas.

Podemos observar que el modelo parece estabilizarse rápidamente, con pérdidas de entrenamiento y validación que se mantienen casi constantes después de las primeras dos épocas. Esto puede sugerir que el modelo ha alcanzado un punto de estancamiento donde no mejora ni empeora con el entrenamiento adicional.

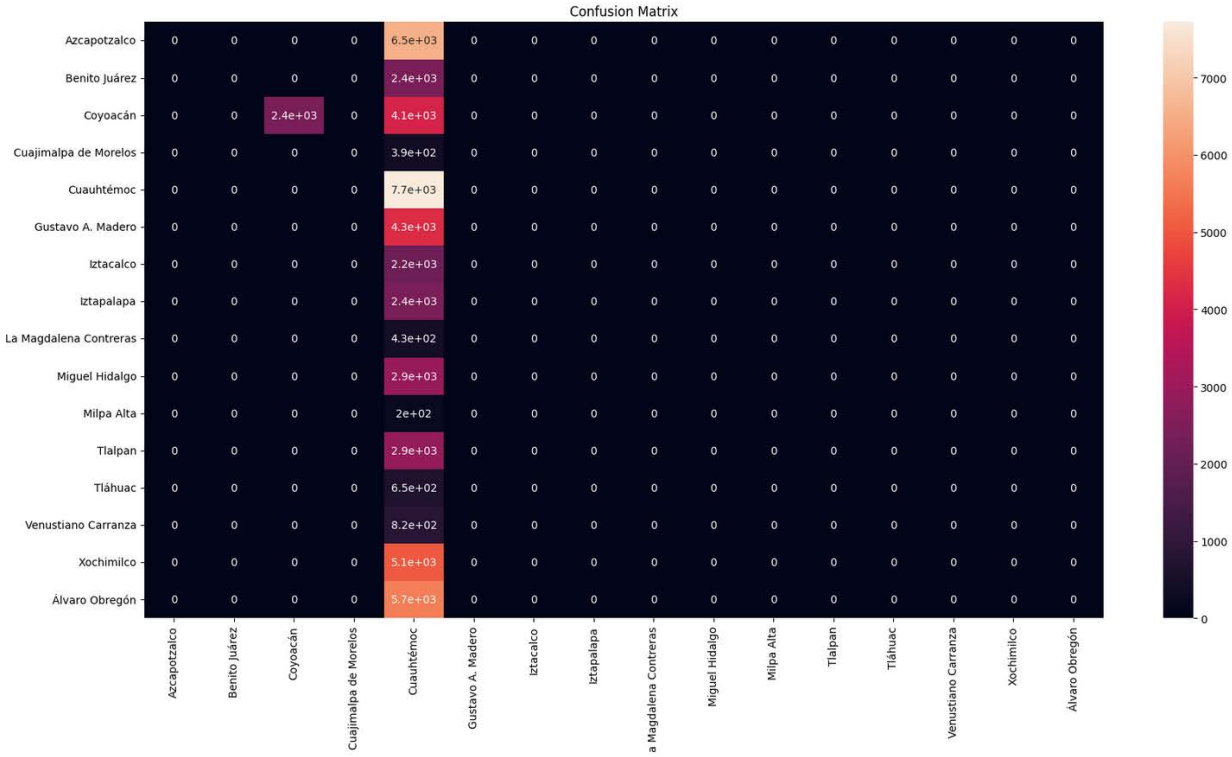

Figura 9. *Matríz de confisión del modelo CNN utilizando texto original.*

La imagen Figura 9 muestra una matriz de confusión que evalúa el rendimiento del modelo en las diferentes clases.

Observaciones Detalladas:

- **Distribución de Predicciones:**
  - La matriz está muy dispersa, con la mayoría de las celdas mostrando valores de cero.
  - Indica que el modelo no está prediciendo correctamente la mayoría de las clases.
- **Clases con Altas Predicciones:**
  - *Cuauhtémoc*: Alto número de ejemplos predichos (7700)
  - *Álvaro Obregón*: Alto número de ejemplos predichos (5700)
  - *Coyoacán*: Alto número de ejemplos predichos (4100)
  - *Xochimilco*: Alto número de ejemplos predichos (5100)
  - Estas clases dominan la matriz de confusión, sugiriendo un sesgo hacia estas clases en particular.
- **Clases con Bajos o Ningunos Predicciones:**
  - Varias clases tienen cero ejemplos correctamente clasificados, lo que sugiere una incapacidad del modelo para reconocer estas clases.

La figura Figura 10 muestra que el modelo funciona levemente para dos clases y mal para el resto. Por ejemplo, el modelo tiene una precisión de 1.0 y una recuperación de 1.0 para la clase 2, lo que significa que el modelo identificó correctamente todos los ejemplos positivos para esta clase. Sin embargo, el modelo tiene una precisión de 0.0 y una recuperación de 0.0 para la clase 1, lo que significa que el modelo no identificó correctamente ninguno de los ejemplos positivos para esta clase.

	precision	recall	f1-score	support
0	0.00	0.00	0.00	6528
1	0.00	0.00	0.00	2415
2	1.00	0.37	0.54	6513
3	0.00	0.00	0.00	391
4	0.16	1.00	0.27	7719
5	0.00	0.00	0.00	4310
6	0.00	0.00	0.00	2189
7	0.00	0.00	0.00	2418
8	0.00	0.00	0.00	430
9	0.00	0.00	0.00	2926
10	0.00	0.00	0.00	196
11	0.00	0.00	0.00	2869
12	0.00	0.00	0.00	651
13	0.00	0.00	0.00	820
14	0.00	0.00	0.00	5055
15	0.00	0.00	0.00	5663
accuracy			0.20	51093
macro avg	0.07	0.09	0.05	51093
weighted avg	0.15	0.20	0.11	51093

Figura 10. *Tabla de métricas de evaluación de clasificación del modelo CNN utilizando texto original.*

La precisión general del modelo es 0.07. Esto significa que el modelo clasificó correctamente el 7 % de los ejemplos en el conjunto de datos.

La precisión y recuperación promedio macro se calculan promediando las puntuaciones de precisión y recuperación para cada clase. La precisión y recuperación promedio ponderado se calculan promediando las puntuaciones de precisión y recuperación para cada clase, ponderadas por el soporte para cada clase.

En este caso, la precisión promedio macro es 0.15 y la recuperación promedio macro es 0.20. Esto significa que las puntuaciones promedio de precisión y recuperación en todas las clases son 0.15 y 0.20, respectivamente.

La precisión promedio ponderada es 0.11 y la recuperación promedio ponderada es 0.11. Esto significa que las puntuaciones promedio de precisión y recuperación en todas las clases, ponderadas por el soporte para cada clase, son 0.11 y 0.11, respectivamente.

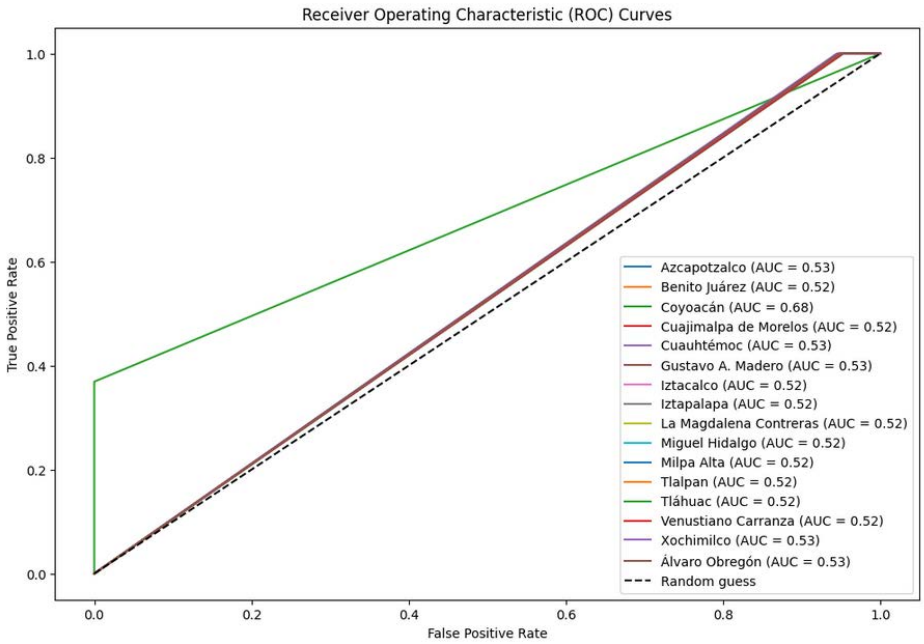

Figura 11. *Gráfica AUC-ROC del modelo CNN utilizando texto original.*

Las clasificación de alcaldías en la Figura 11 tiene un rendimiento inferior, con AUCs inferiores a 0.53. Esto significa que los modelos de clasificación de estas alcaldías no son tan precisos.

- AUC baja: Un AUC (Área Bajo la Curva) bajo indica que el modelo no es capaz de diferenciar bien entre las clases positiva y negativa. En la gráfica que proporcionaste, no se especifica el valor del AUC, pero si es significativamente menor que 0.5, podría ser un indicio de un mal rendimiento.
- Forma de la curva: Si la curva ROC se acerca a la línea diagonal (que representa un clasificador aleatorio), esto también podría indicar que el modelo no es mucho mejor que adivinar al azar.

El modelo funciona bien para algunas clases y mal para otras. La precisión general del modelo es baja, lo que significa que el modelo no clasifica correctamente la mayoría de los ejemplos en el conjunto de datos. La precisión y recuperación promedio macro también son bajas, lo que significa que las puntuaciones promedio de precisión y recuperación en todas las clases también son bajas. La precisión y recuperación promedio ponderado son ligeramente más altas que la precisión y recuperación promedio macro, pero siguen siendo bajas.

8.2 Resultados MLP con el texto original

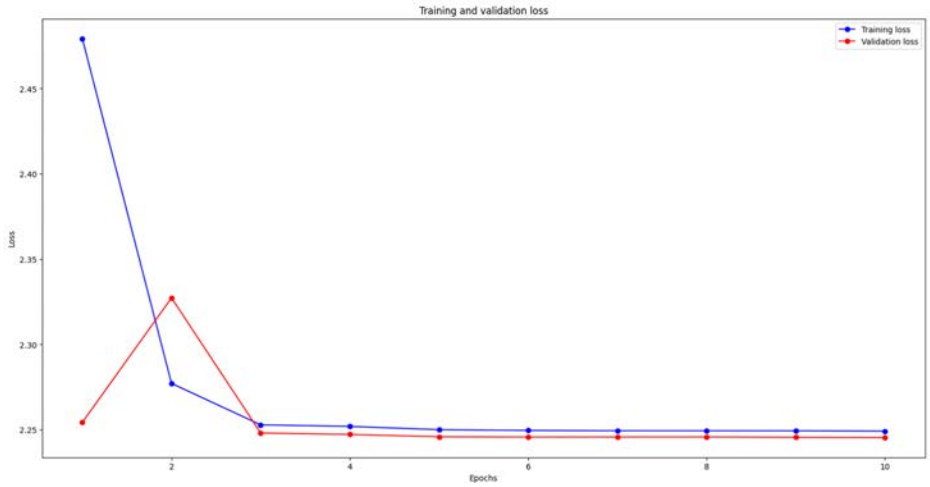

Figura 12. *Métricas de perdida en el modelo MLP utilizando texto original.*

La Figura 12 muestra una gráfica de la precisión categórica escasa durante el entrenamiento y la validación a lo largo de 10 épocas utilizando el modelo MLP.

Observaciones Detalladas:

- **Época 1:**
  - La pérdida de entrenamiento comienza muy alta, alrededor de 2.45, y la pérdida de validación es un poco más baja, alrededor de 2.25.
- **Época 2:**
  - Hay una reducción drástica en la pérdida de entrenamiento, bajando por debajo de 2.30. Sin embargo, la pérdida de validación aumenta ligeramente.
- **Épocas 3:**
  - Ambas pérdidas, tanto de entrenamiento como de validación, se estabilizan y empiezan a acercarse.
- **Épocas 4 - 10 :**
  - Las pérdidas de entrenamiento y validación se mantienen muy cercanas y bajas, alrededor de 2.25, con ligeras fluctuaciones.

La Figura 13 muestra la exactitud categórica escasa (sparse categorical accuracy) en el entrenamiento y la validación a lo largo de 10 épocas para el modelo MLP.

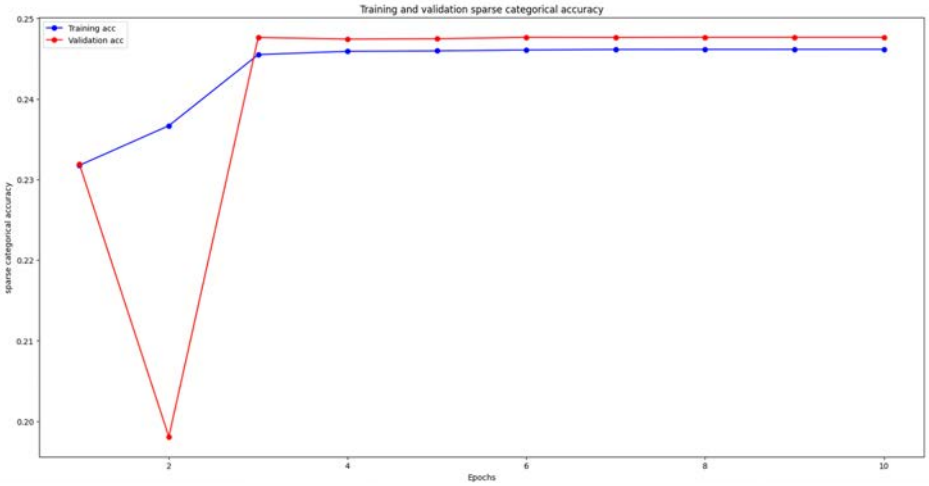

Figura 13. *Métricas de exactitud en el modelo MLP utilizando texto original.*

Observaciones Detalladas:

- **Época 2:**
  - Observamos una caída abrupta en la exactitud de validación. Esto es inusual y sugiere que hay un problema con la forma en que se está evaluando el modelo o con el propio conjunto de validación. Podría ser un indicio de overfitting temprano, donde el modelo está ajustando demasiado bien los datos de entrenamiento y no generaliza bien en datos no vistos.
- **Época 2 - 10:**
  - La exactitud de entrenamiento como la de validación se estancan. Esto sugiere que el modelo no está aprendiendo significativamente más con el tiempo. Un estancamiento en el rendimiento puede indicar que el modelo ha alcanzado su capacidad máxima para aprender de los datos disponibles o que la arquitectura del modelo no es adecuada para el problema.

Los valores de exactitud son bastante bajos (alrededor de 0.24), lo cual indica que el modelo no está haciendo un buen trabajo en la clasificación. En muchos contextos de clasificación, esperaríamos ver una exactitud mucho mayor para considerar el modelo como útil.

La matriz de confusión en la Figura 14 muestra que el modelo tiene problemas de precisión y balance en la clasificación, la presencia de varios errores significativos fuera de la diagonal principal indica que el modelo está confundiendo las clases con frecuencia.

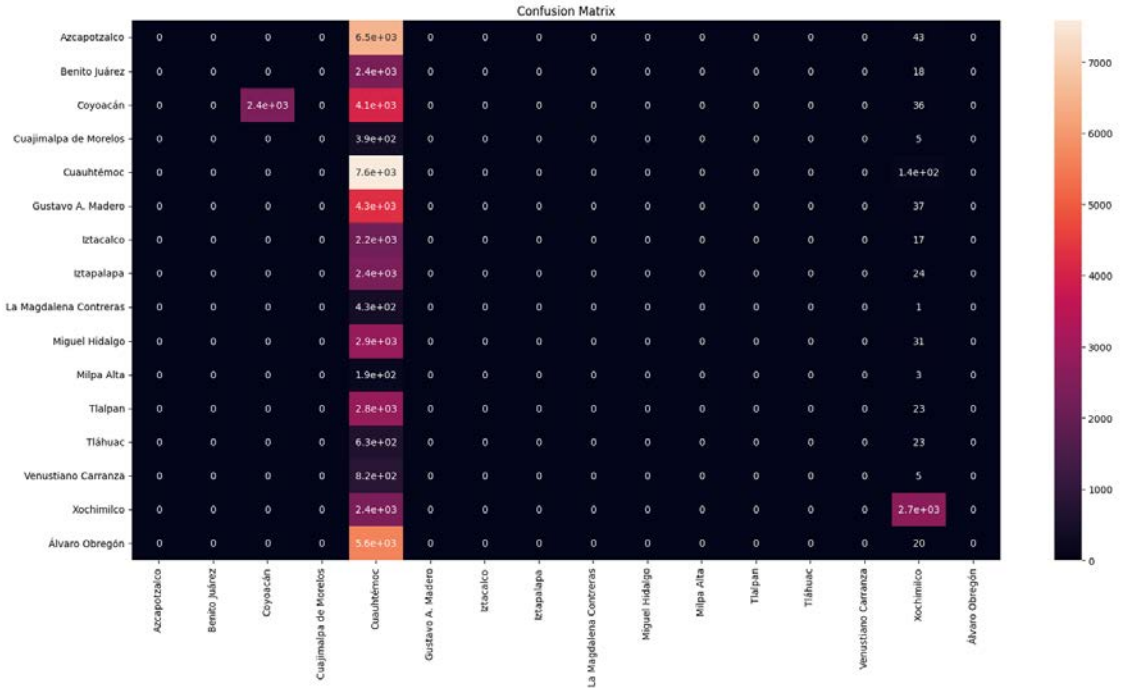

Figura 14. *Matriz de confusión del modelo MLP utilizando texto original.*

	precision	recall	f1-score	support
0	0.00	0.00	0.00	6528
1	0.00	0.00	0.00	2415
2	1.00	0.37	0.54	6513
3	0.00	0.00	0.00	391
4	0.17	0.98	0.28	7719
5	0.00	0.00	0.00	4310
6	0.00	0.00	0.00	2189
7	0.00	0.00	0.00	2418
8	0.00	0.00	0.00	430
9	0.00	0.00	0.00	2926
10	0.00	0.00	0.00	196
11	0.00	0.00	0.00	2869
12	0.00	0.00	0.00	651
13	0.00	0.00	0.00	820
14	0.86	0.53	0.66	5055
15	0.00	0.00	0.00	5663
accuracy			0.25	51093
macro avg	0.13	0.12	0.09	51093
weighted avg	0.24	0.25	0.18	51093

Figura 15. *Tabla de métricas de evaluación de clasificación MLP utilizando texto original.*

En la Figura 15 la precisión y el recall para la mayoría de las clases son bajos, con muchos valores en 0.00. Esto significa que el modelo no está logrando identificar correctamente las instancias de estas clases, ya sea prediciendo muy pocos verdaderos positivos o realizando muchas falsas

predicciones.

El F1-score, que es la media armónica de la precisión y el recall, también es extremadamente bajo para la mayoría de las clases, lo que refuerza la conclusión de que el modelo tiene un desempeño general muy pobre. La exactitud general del modelo es solo del 25 %, lo que indica que solo una cuarta parte de las predicciones del modelo son correctas. Los promedios macro y ponderado (macro avg y weighted avg) también son muy bajos, con valores de precisión, recall y F1-score todos por debajo de 0.25, indicando que el problema es generalizado y no solo debido a algunas clases específicas.

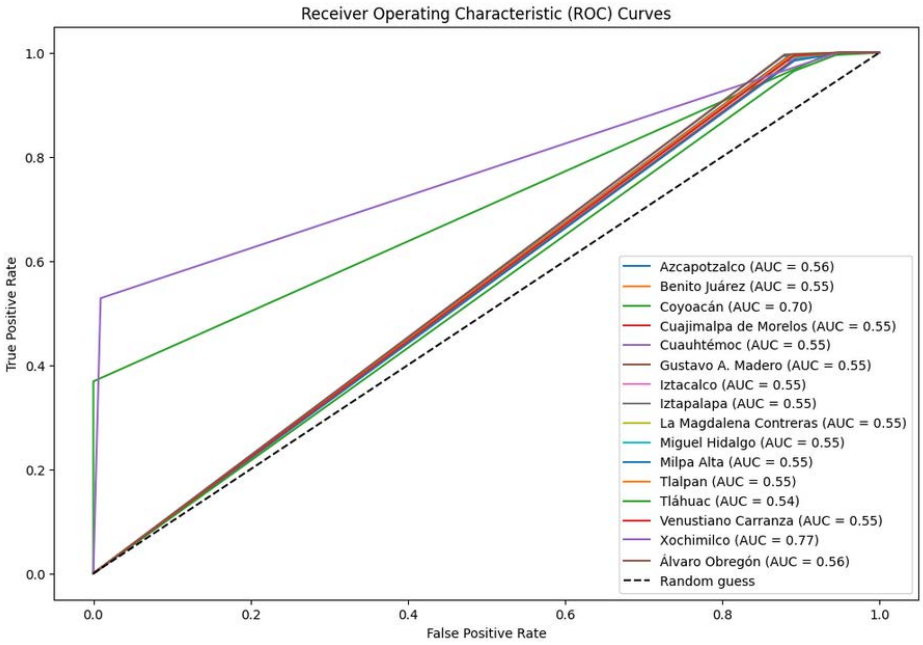

Figura 16. *Gráfica AUC-ROC del modelo MLP utilizando texto original.*

En la Figura 16 podemos observar que la mayoría de las curvas ROC están muy cerca de la línea diagonal, la cual representa un clasificador aleatorio (AUC de 0.5). Las AUC para la mayoría de las clases están alrededor de 0.55, lo que sugiere que el modelo apenas supera el azar en su capacidad para distinguir entre las clases; La cercanía de las curvas ROC a la línea de azar y los bajos valores de AUC indican que el modelo es ineficaz para predecir con precisión la mayoría de las clases.

El comportamiento mostrado en las gráficas anteriores indica que el modelo está enfrentando problemas de sobreajuste temprano, inestabilidad en el aprendizaje, y una posible incapacidad para mejorar después de las primeras épocas. Estas señales combinadas sugieren que el modelo no está bien ajustado y que los resultados obtenidos no serán confiables ni generalizables a datos no vistos.

8.3 Resultados del modelo chatgpt con texto original

El modelo propuesto por chatgpt comienza con una capa de incrustación (Embedding Layer), cuyo objetivo es transformar las entradas de texto, representadas como enteros, en vectores densos de tamaño fijo. Esta capa toma una dimensión de entrada de 10,000, que corresponde al tamaño del vocabulario, y produce vectores de incrustación de 128 dimensiones.

Se incluyó una capa Long Short-Term Memory (LSTM) bidireccional, diseñada para procesar datos secuenciales en ambas direcciones: hacia adelante y hacia atrás. Esta capa cuenta con 64 unidades en cada dirección, lo que resulta en una salida de 128 dimensiones tras concatenar las salidas de ambas direcciones.

Utilizó una capa densa (Dense Layer) con 64 neuronas y la función de activación ReLU para reducir la no linealidad en el modelo, permitiendo que aprenda una variedad de patrones complejos presentes en los datos, finalmente una capa densa de salida (Output Dense Layer) que contiene 16 neuronas, correspondientes a las 16 clases de alcaldías. La función de activación softmax se emplea en esta capa para producir una distribución de probabilidad sobre las clases posibles, facilitando la clasificación final.

El modelo se compiló utilizando la función de pérdida sparse_categorical_crossentropy, adecuada para problemas de clasificación multi-clase con etiquetas enteras. El optimizador elegido es Adam (Adaptive Moment Estimation).

```
Model: "sequential_6"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 embedding_6 (Embedding)     (None, None, 128)         1280000

 bidirectional_5 (Bidirectio  (None, 128)              98816
 nal)

 dense_11 (Dense)            (None, 64)                8256

 dense_12 (Dense)            (None, 16)                1040

=================================================================
Total params: 1,388,112
Trainable params: 1,388,112
Non-trainable params: 0
_________________________________________________________________
```

Figura 17. *Representación visual del modelo propuesto por chatgpt.*

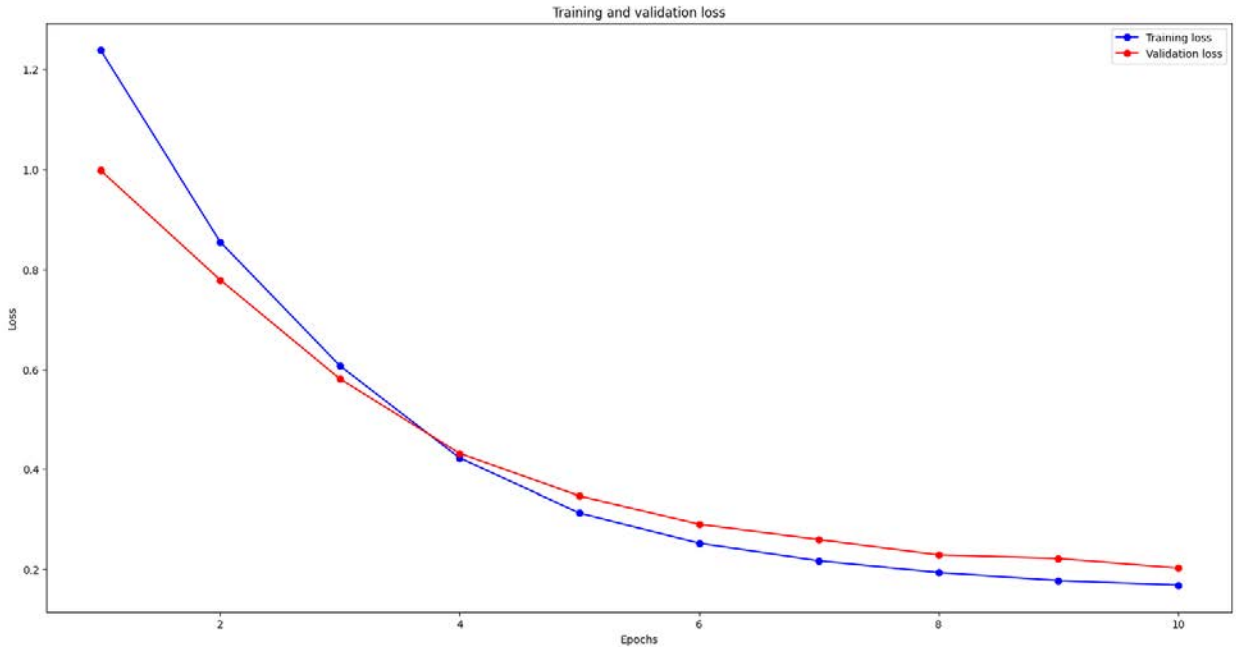

Figura 18. *Gráfica de perdida del modelo propuesto por chatgpt.*

la Figura 18 ilustra la pérdida del modelo en los conjuntos de entrenamiento y validación a lo largo de las épocas. La pérdida en el conjunto de entrenamiento disminuye constantemente, lo cual es esperado durante el proceso de aprendizaje. La pérdida en el conjunto de validación también muestra una disminución, aunque a un ritmo más lento. Esta desaceleración en la reducción de la pérdida de validación puede ser un indicio de sobreajuste.

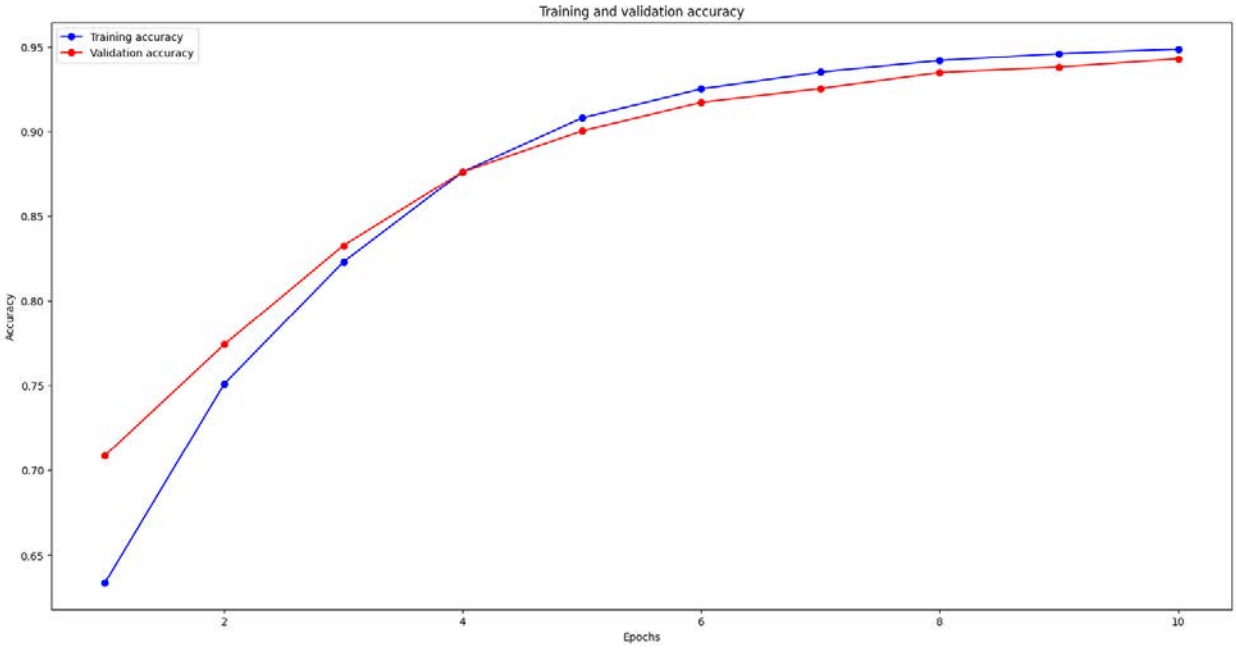

Figura 19. *Gráfica de presición del modelo propuesto por chatgpt.*

La Figura 19 muestra la evolución de la precisión del modelo tanto en el conjunto de entrenamiento como en el de validación a lo largo de 10 épocas. Se observa que la precisión en el conjunto de validación es consistentemente más alta que en el conjunto de entrenamiento durante las primeras épocas. Este fenómeno es inusual y podría indicar que el modelo se está adaptando demasiado bien al conjunto de validación, posiblemente debido a una distribución de datos no homogénea entre los dos conjuntos. Hacia las últimas épocas, las curvas de precisión se aproximan, lo que sugiere un leve indicio de sobreajuste (overfitting) del modelo al conjunto de entrenamiento.

```
1595/1595 [==============================] - 46s 29ms/step
              precision    recall  f1-score   support

           0       0.93      0.96      0.95      6033
           1       0.94      0.92      0.93      2078
           2       0.96      0.96      0.96      5531
           3       0.97      0.87      0.92       490
           4       0.92      0.93      0.93      6844
           5       0.95      0.92      0.94      4718
           6       0.96      0.94      0.95      2864
           7       0.90      0.92      0.91      2551
           8       0.91      0.85      0.88       382
           9       0.97      0.98      0.97      6364
          10       0.99      0.85      0.92       170
          11       0.92      0.94      0.93      2964
          12       0.95      0.83      0.89       678
          13       0.93      0.85      0.89       867
          14       0.92      0.96      0.94      4210
          15       0.93      0.94      0.94      4286

    accuracy                           0.94     51030
   macro avg       0.94      0.91      0.93     51030
weighted avg       0.94      0.94      0.94     51030
```

Figura 20. *Reporte de clasificación del modelo propuesto por chatgpt.*

Los promedios generales vistos en la Figura 20 parecen prometedores, con valores alrededor de 0.94 para precisión, recall y f1-score, un análisis por clase revela deficiencias significativas. Por ejemplo, la clase 3 presenta una precisión alta de 0.97 pero un recall de solo 0.87, indicando una alta tasa de falsos negativos. Similarmente, la clase 12 tiene un recall de 0.85, sugiriendo que el modelo falla en capturar una proporción considerable de instancias verdaderas de esta clase. Estas discrepancias evidencian que el rendimiento del modelo no es uniforme, y algunas clases están mucho más afectadas que otras.

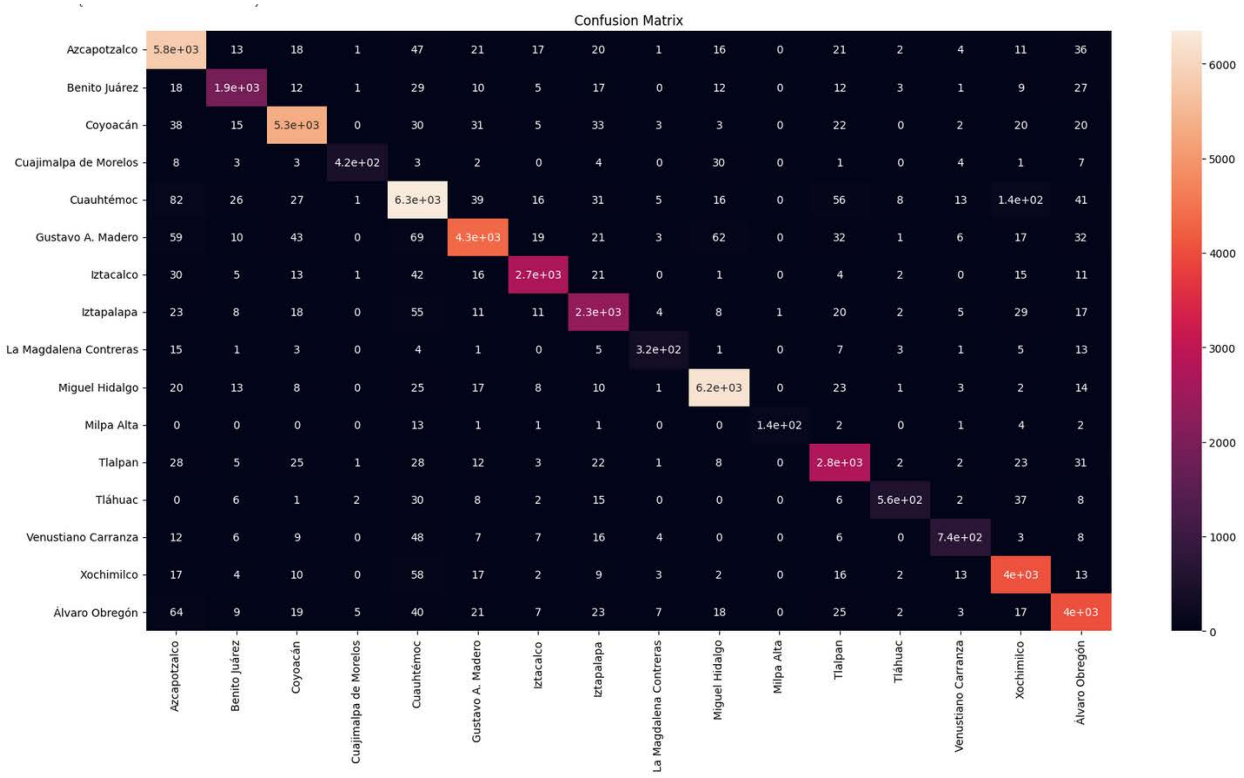

Figura 21. *Matriz de confusión del modelo propuesto por chatgpt.*

La matriz presentada en la Figura 21 muestra múltiples valores dispersos fuera de esta diagonal, señalando errores de predicción. Por ejemplo, Azcapotzalco (clase 0) muestra una dispersión significativa de predicciones incorrectas en diversas otras clases. Cuauhtémoc (clase 4) se predijo incorrectamente como Álvaro Obregón (clase 15) en 102 ocasiones, y Miguel Hidalgo (clase 9) fue erróneamente clasificado como Cuauhtémoc (clase 4) y Tláhuac (clase 12) 83 y 62 veces respectivamente. Estos errores indican problemas en la capacidad del modelo para diferenciar entre ciertas clases.

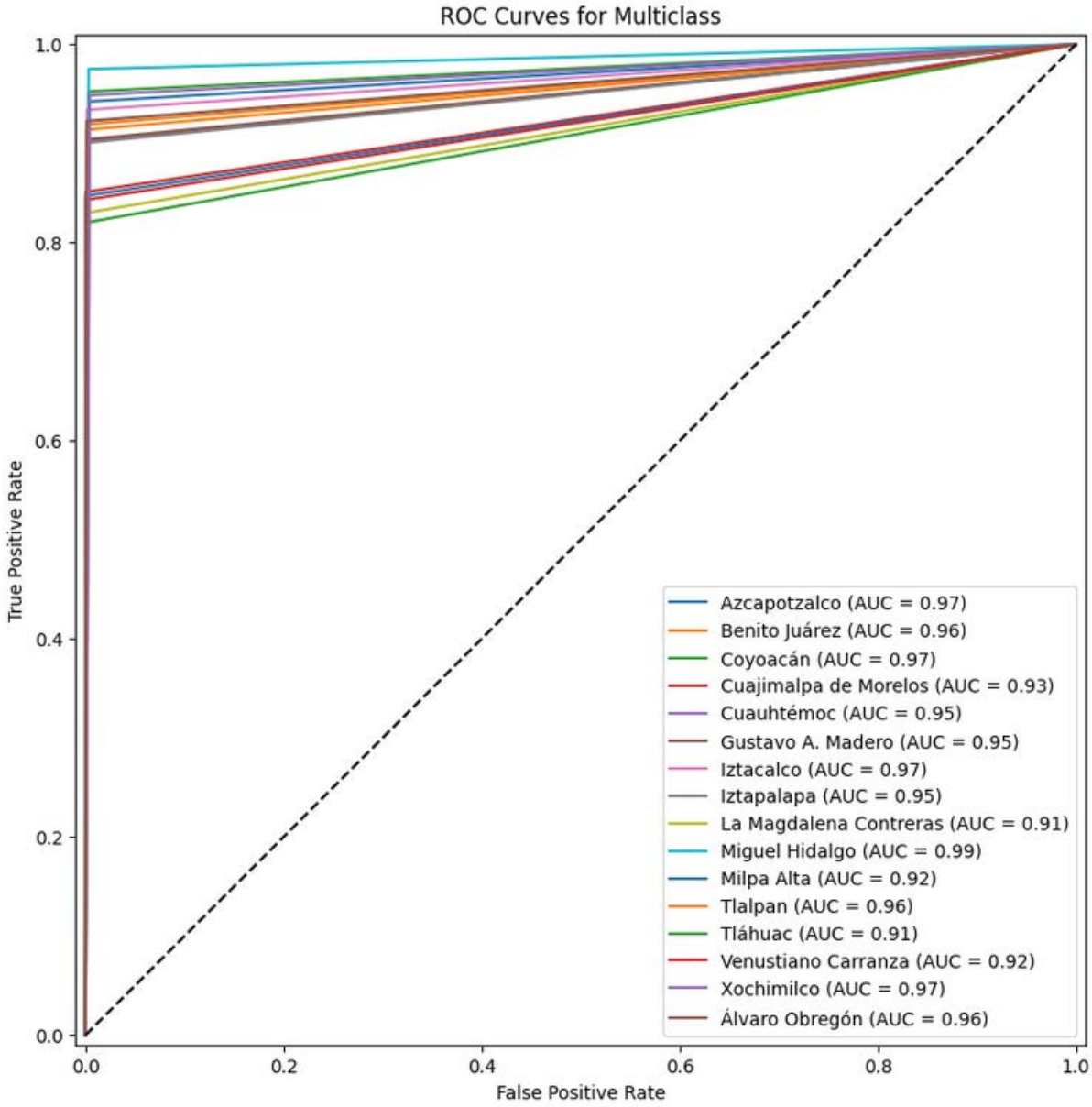

Figura 22. *Gráifca AUC-ROCK del modelo propuesto por chatgpt.*

La Figura 22 muestra las curvas ROC para cada una de las 16 clases, junto con el área bajo la curva (AUC) para cada una. Las curvas ROC indican un rendimiento generalmente bueno del modelo, con valores de AUC altos (mayoría >0.90), lo que sugiere una buena capacidad de discriminación para la mayoría de las clases. No obstante, la variabilidad en los valores de AUC entre clases sugiere que el modelo funciona mejor en algunas clases que en otras.

8.4 Resultados del modelo CNN con propuesta de entrada

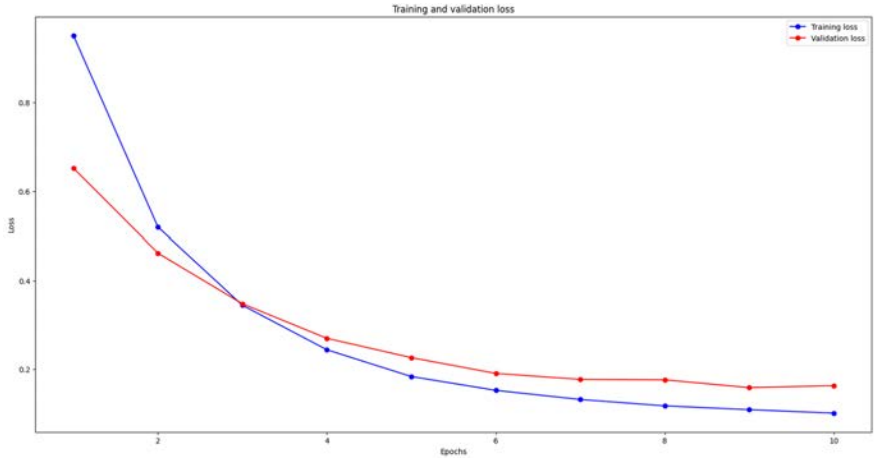

Figura 23. *Relación entre la pérdida de entrenamiento y la pérdida de validación durante el entrenamiento.*

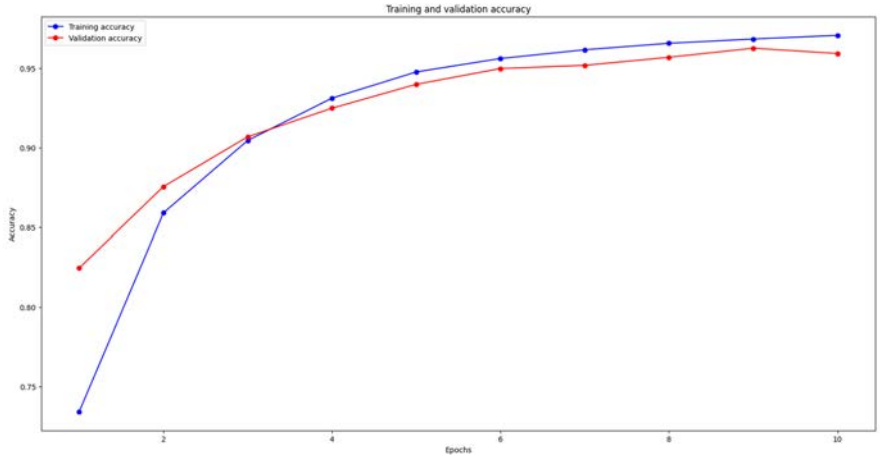

Figura 24. *Relación entre la presición de entrenamiento y la presición de validación durante el entrenamiento.*

Las gráficas Figura 23 y Figura 24 el comportamiento durante el entrenamiento de un modelo de red neuronal convolucional, podemos observar que la precisión del entrenamiento aumenta rápidamente al principio del entrenamiento y luego se estabiliza en un valor cercano al 100 %, mientras que la precisión de la validación también aumenta al principio del entrenamiento, pero luego comienza a disminuir.

También podemos observar que la pérdida del entrenamiento disminuye rápidamente al principio del y luego se estabiliza en un valor bajo y la pérdida de la validación también disminuye al principio del entrenamiento, pero luego comienza a aumentar.

La forma en que se comportan las curvas del entrenamiento y la validación nos proporciona información sobre el rendimiento del modelo de red neuronal convolucional. En este caso, podemos observar que el modelo está sobreajustándose a los datos de entrenamiento, esto significa que el modelo está aprendiendo las características específicas de los datos de entrenamiento con tanta precisión que no es capaz de generalizar bien a los datos nuevos.

El sobreajuste queda evidenciado por los siguientes motivos:

- La precisión del entrenamiento es significativamente mayor que la precisión de la validación.
- La precisión de la validación comienza a disminuir después de un cierto número de épocas.
- La pérdida del entrenamiento es significativamente menor que la pérdida de la validación.
- La pérdida de la validación comienza a aumentar después de un cierto número de épocas.

El sobreajuste puede tener consecuencias negativas para el rendimiento del modelo. Un modelo sobreajustado no podrá generalizar bien a los datos nuevos, lo que significa que tendrá un mal rendimiento cuando se utilice en la práctica.

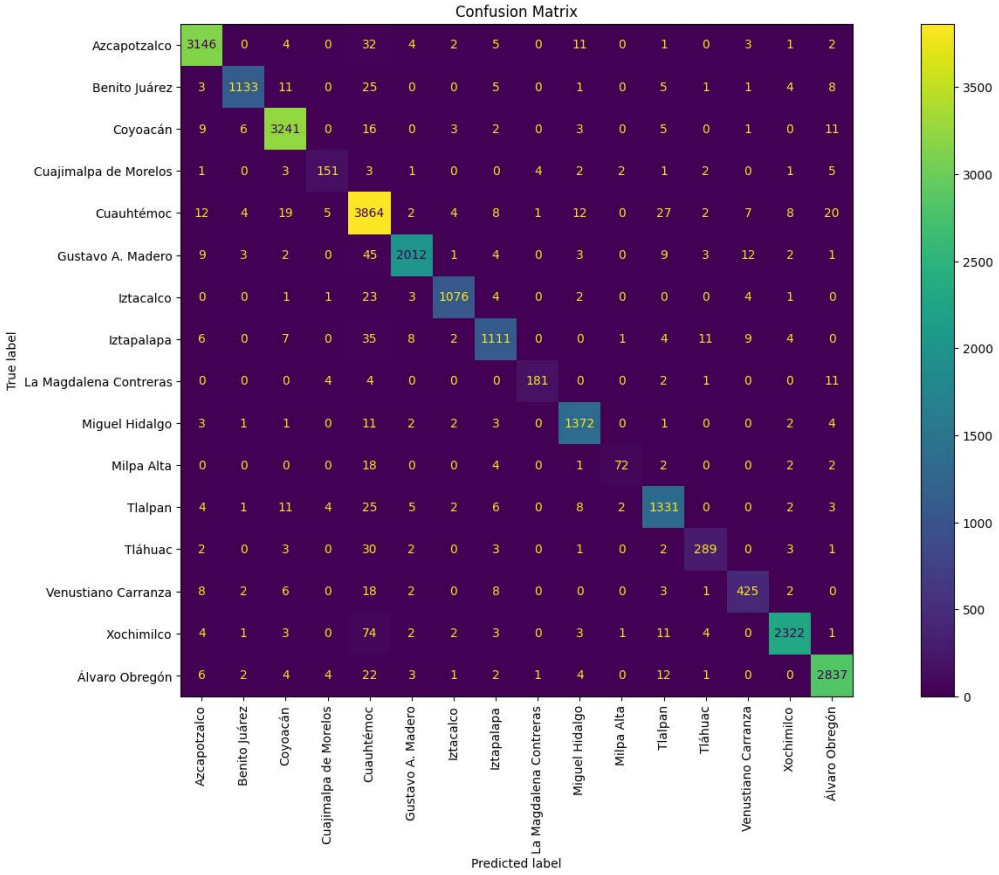

Figura 25. *Matriz de confusión para el modelo CNN con entrada propuesta.*

	precision	recall	f1-score	support
Azcapotzalco	0.98	0.98	0.98	3211
Benito Juárez	0.98	0.95	0.96	1197
Coyoacán	0.98	0.98	0.98	3297
Cuajimalpa de Morelos	0.89	0.86	0.88	176
Cuauhtémoc	0.91	0.97	0.94	3995
Gustavo A. Madero	0.98	0.96	0.97	2106
Iztacalco	0.98	0.97	0.97	1115
Iztapalapa	0.95	0.93	0.94	1198
La Magdalena Contreras	0.97	0.89	0.93	203
Miguel Hidalgo	0.96	0.98	0.97	1402
Milpa Alta	0.92	0.71	0.80	101
Tlalpan	0.94	0.95	0.94	1404
Tláhuac	0.92	0.86	0.89	336
Venustiano Carranza	0.92	0.89	0.91	475
Xochimilco	0.99	0.96	0.97	2431
Álvaro Obregón	0.98	0.98	0.98	2899
accuracy			0.96	25546
macro avg	0.95	0.93	0.94	25546
weighted avg	0.96	0.96	0.96	25546

Figura 26. *Reporte de clasificación para el modelo CNN con entrada propuesta.*

En las gráficas Figura 25 y Figura 26 podemos analizar que en la matriz de confusión se refleja el rendimiento del modelo para cada clase individual examinando la fila correspondiente a esa clase. Por ejemplo, si analizamos la fila “Azcapotzalco”, podemos observar que el modelo clasificó correctamente 3,146 textos en Azcapotzalco, pero también clasificó incorrectamente 4 casos como Benito Juárez, 4 casos como Coyoacán, etc y en el reporte de la clasificación también podemos observar que el modelo tuvo un buen desempeño clasificando los textos.

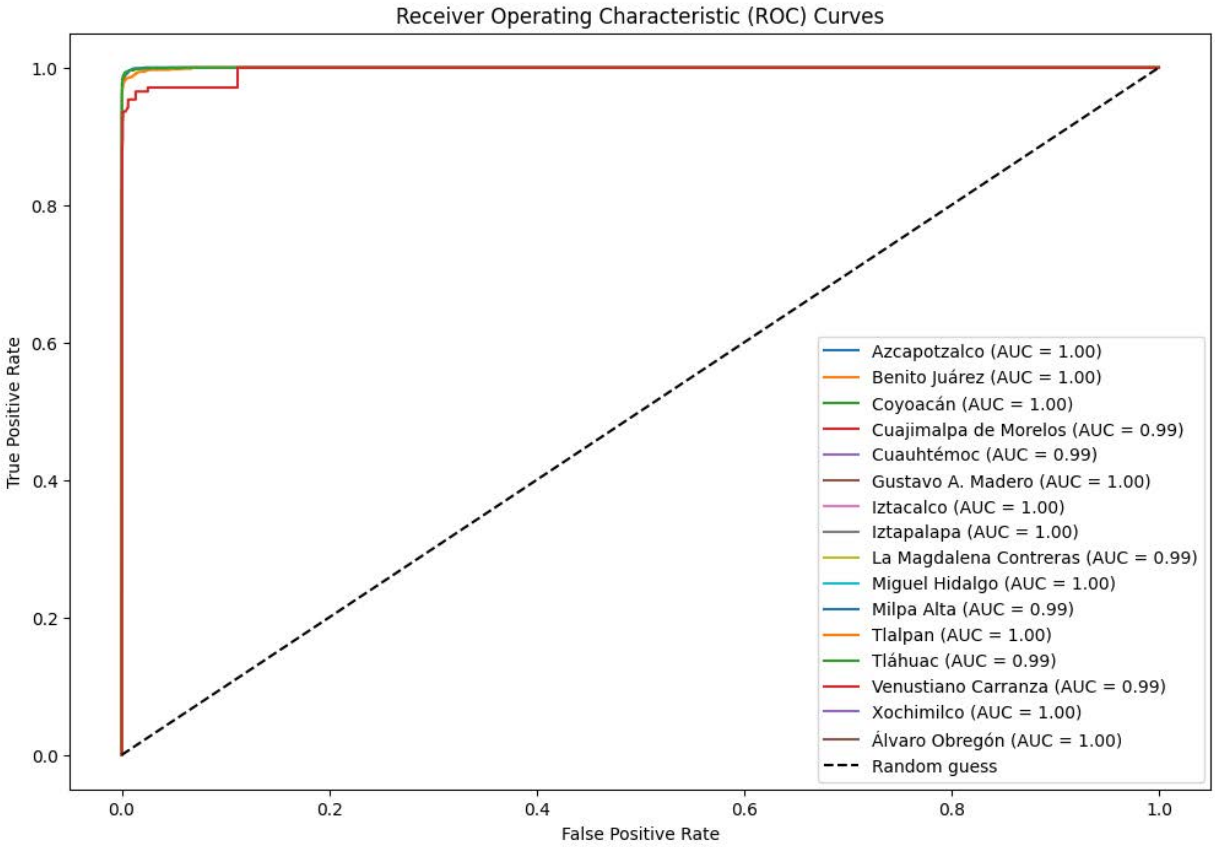

Figura 27. *Gráfica AUC-ROC del modelo CNN utilizando entrada propuesta.*

La perfección observada en las curvas ROC de la Figura 27 y su similitud entre alcaldías sugieren un posible sobreajuste en los modelos de clasificación. Esto implica que los modelos memorizan los datos de entrenamiento en lugar de aprender características distintivas de cada delegación, afectando negativamente su capacidad de generalización a nuevos datos.

Al tener en cuenta el análisis de las gráficas anteriores (23, 24 y 27) y la información que nos proporcionaron sobre el rendimiento general del modelo, podemos deducir que este resultado es debido al sobreajuste, por lo cual no lo hace un modelo óptimo para clasificar textos por alcaldía.

8.5 Resultados del modelo MLP con entrada propuesta

Las figuras Figura 28 y Figura 29 muestran las curvas de pérdida y precisión de entrenamiento y validación durante el entrenamiento del modelo.

8.5.1 Pérdida

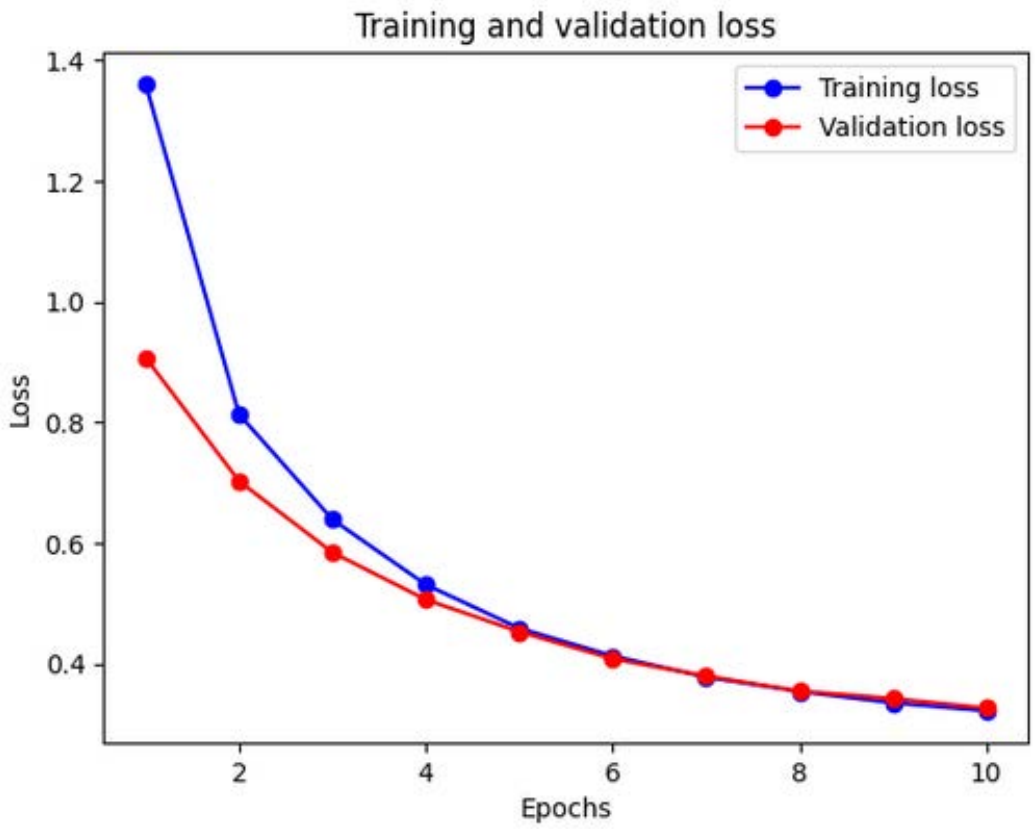

Figura 28. *Relación entre la pérdida de entrenamiento y la pérdida de validación durante el entrenamiento.*

La curva de entrenamiento Disminuye constantemente durante las 10 épocas, lo que indica que el modelo está aprendiendo a clasificar los textos en las diferentes alcaldías de manera efectiva, mientras que la curva de validación disminuye durante las primeras 10 épocas, lo que sugiere que el modelo está generalizando bien a datos nuevos.

Ambas curvas convergen a valores bajos, lo que indica que el modelo ha aprendido a hacer buenas predicciones tanto en el conjunto de datos de entrenamiento como en el conjunto de datos de validación y al no aumentar significativamente la curva de validación después del punto mínimo de pérdida, sugiere que el modelo no está sobreajustándose al conjunto de datos de entrenamiento.

8.5.2 Precisión

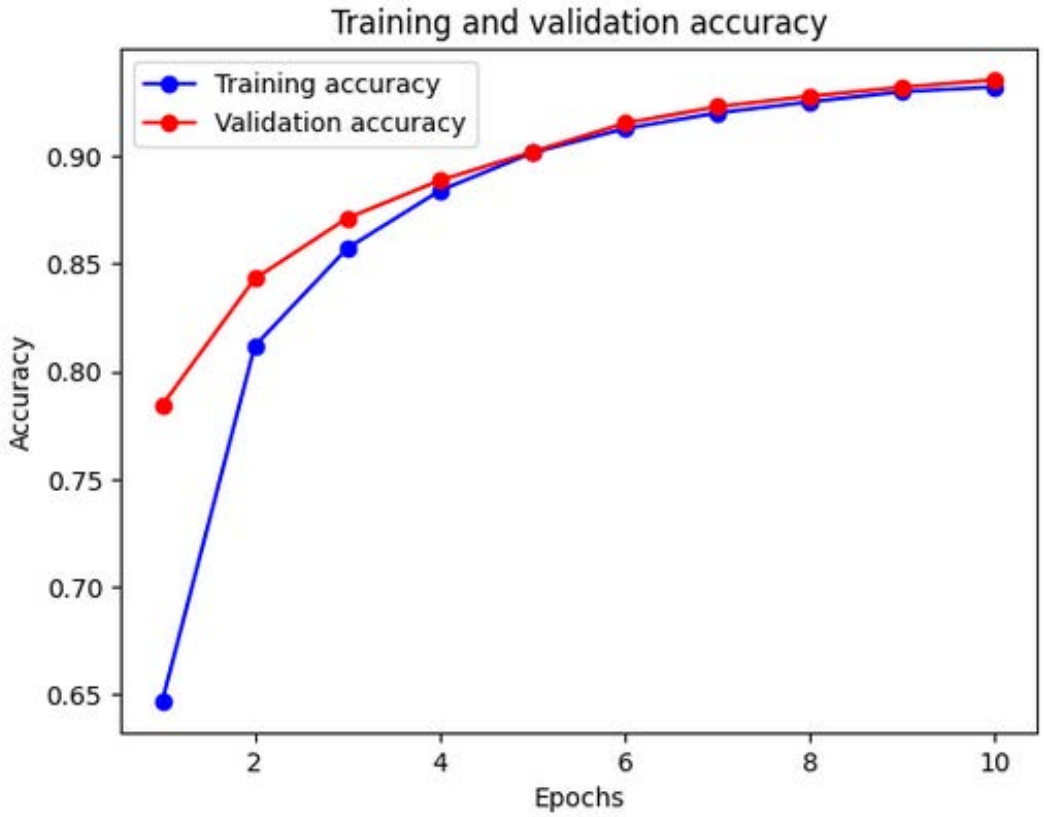

Figura 29. *Relación entre la presición de entrenamiento y la presición de validación durante el entrenamiento.*

La curva de entrenamiento aumenta constantemente durante las 10 épocas, lo que indica que el modelo está aprendiendo a clasificar los textos en las diferentes alcaldías de manera efectiva mientras que la urva de validación aumenta durante las primeras 10 épocas, lo que sugiere que el modelo está generalizando bien a datos nuevos.

Ambas curvas convergen a valores altos, lo que indica que el modelo ha aprendido a hacer buenas predicciones tanto en el conjunto de datos de entrenamiento como en el conjunto de datos de validación.

En general, las curvas de pérdida y precisión sugieren que el modelo ha sido entrenado correctamente y es probable que tenga un buen rendimiento en datos nuevos.

8.5.3 Matriz de Confusión

La matriz de confusión Figura 30 muestra el número de ejemplos de cada clase que fueron clasificados correctamente o incorrectamente por el modelo.

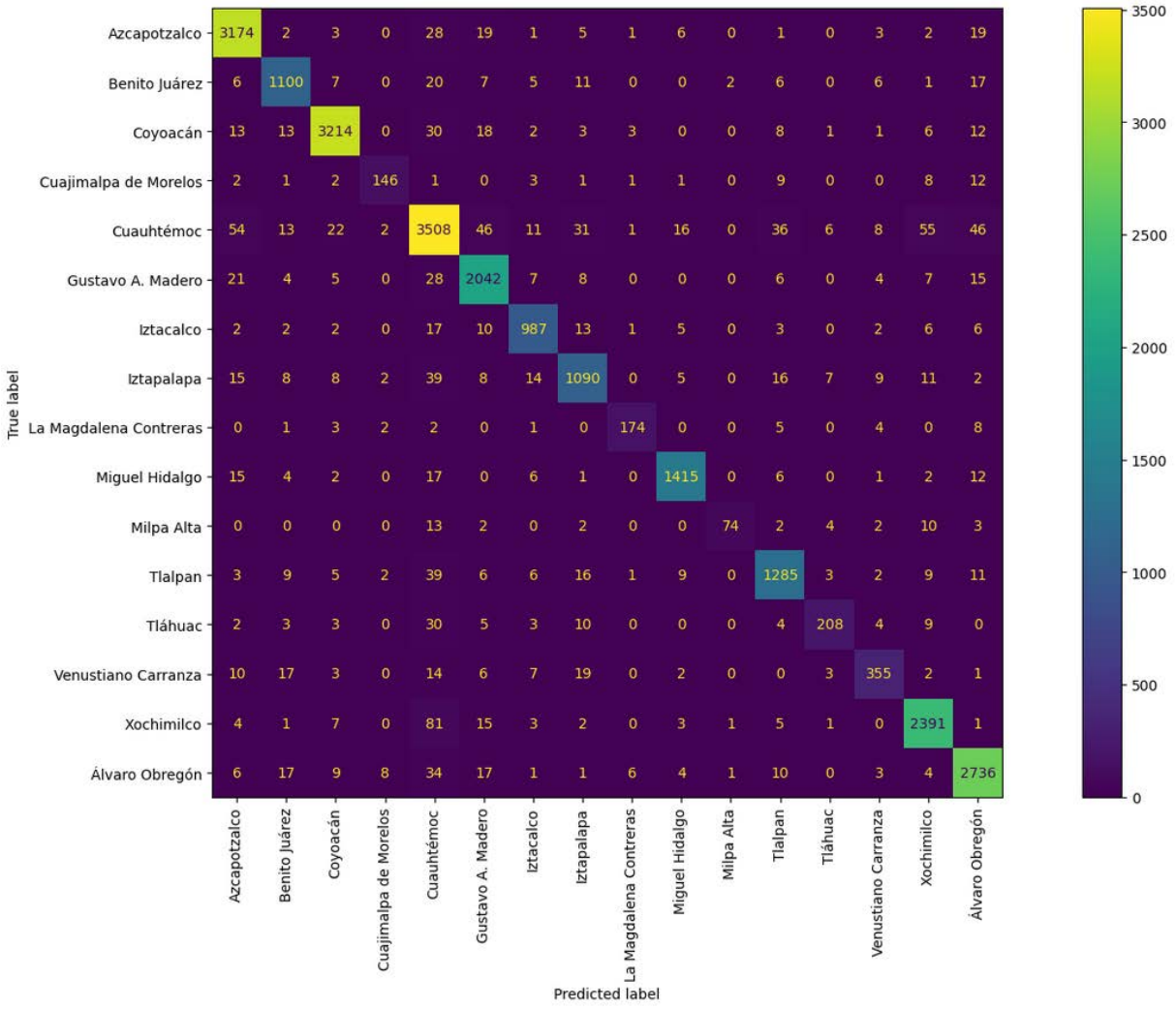

Figura 30. Relación entre la pérdida de entrenamiento y la pérdida de validación durante el entrenamiento.

Los Valores diagonales representan el número de ejemplos de cada clase que fueron clasificados correctamente. La mayoría de los valores diagonales son altos, lo que indica que el modelo tiene una buena precisión para la mayoría de las alcaldías; Los valores fuera de la diagonal representan el número de ejemplos de cada clase que fueron clasificados incorrectamente como otra clase.

Algunos valores fuera de la diagonal son relativamente altos, lo que indica que el modelo tiene cierta dificultad para distinguir entre algunas alcaldías pero en general, la matriz de confusión sugiere

que el modelo tiene un buen rendimiento general, pero hay algunas áreas donde puede mejorar, especialmente en la distinción entre algunas alcaldías específicas.

	precision	recall	f1-score	support
Azcapotzalco	0.95	0.97	0.96	3264
Benito Juárez	0.92	0.93	0.92	1188
Coyoacán	0.98	0.97	0.97	3324
Cuajimalpa de Morelos	0.90	0.78	0.84	187
Cuauhtémoc	0.90	0.91	0.90	3855
Gustavo A. Madero	0.93	0.95	0.94	2147
Iztacalco	0.93	0.93	0.93	1056
Iztapalapa	0.90	0.88	0.89	1234
La Magdalena Contreras	0.93	0.87	0.90	200
Miguel Hidalgo	0.97	0.96	0.96	1481
Milpa Alta	0.95	0.66	0.78	112
Tlalpan	0.92	0.91	0.92	1406
Tláhuac	0.89	0.74	0.81	281
Venustiano Carranza	0.88	0.81	0.84	439
Xochimilco	0.95	0.95	0.95	2515
Álvaro Obregón	0.94	0.96	0.95	2857
accuracy			0.94	25546
macro avg	0.93	0.89	0.90	25546
weighted avg	0.94	0.94	0.94	25546

Figura 31. Reporte de clasificación MLP con input preprocesado.

Los resultados obtenidos Figura 31 sugieren que el modelo de red neuronal propuesto, entrenado con solo 10 épocas, es capaz de clasificar textos en las diferentes alcaldías de la Ciudad de México con un buen nivel de precisión. Es importante destacar que este entrenamiento corto ayuda a prevenir el sobreajuste y mejora la generalización del modelo.

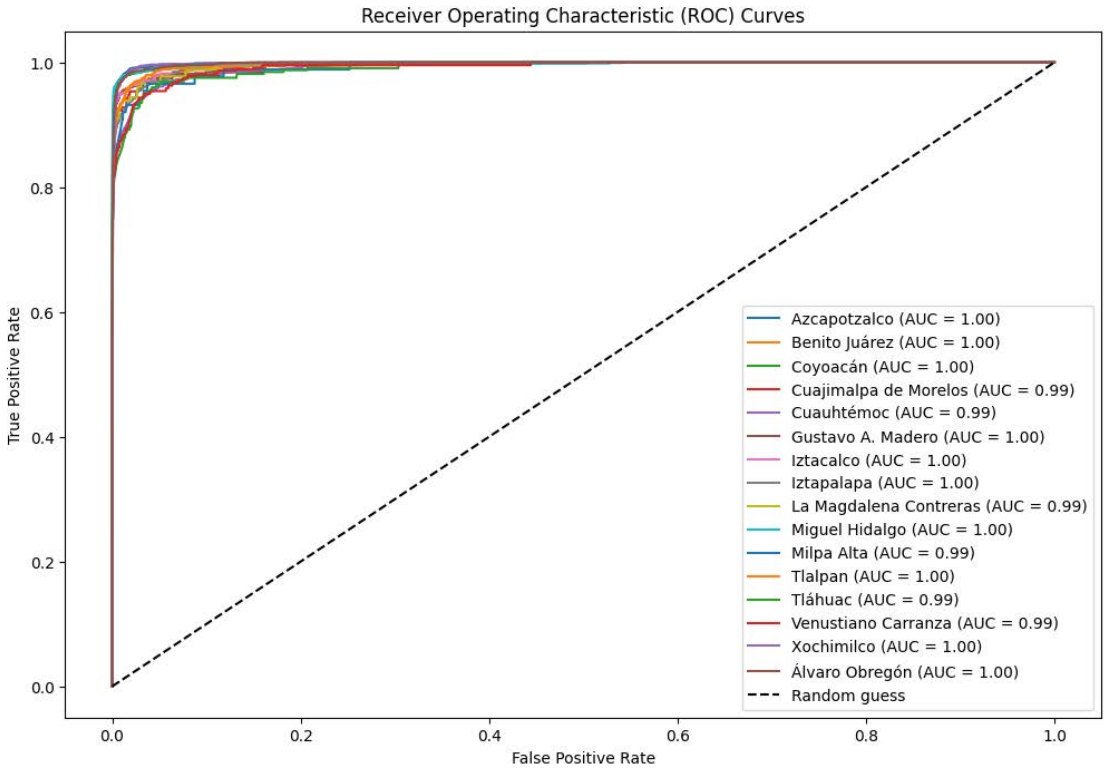

Figura 32. *Gráfica AUC-ROC del modelo MLP utilizando entrada propuesta.*

Las curvas ROC para las alcaldías en la Figura 32 muestran un comportamiento muy cercano a la esquina superior izquierda del gráfico, lo que indica un alto rendimiento del modelo de clasificación. El Área Bajo la Curva para todas las alcaldías es cercano a 1.0, lo que confirma la alta capacidad del modelo para discriminar entre las delegaciones, en contraste, la curva ROC para el modelo aleatorio (línea diagonal) indica que este modelo no tiene capacidad para diferenciar entre las delegaciones.

Los resultados obtenidos sugieren que el modelo de clasificación utilizado tiene un buen rendimiento para predecir las alcaldías de la Ciudad de México. El modelo es capaz de identificar correctamente a la mayoría de las alcaldías con una baja tasa de falsos positivos.

Es importante destacar que el buen rendimiento del modelo se debe a diversos factores, como la calidad y cantidad de los datos de entrenamiento, la arquitectura y los parámetros del modelo, y la técnica de preprocesamiento de datos utilizada.

8.6 Comparación Final

Tabla 6. *Comparación de las métricas de clasificación de todos los modelos.*

Alcaldía	**precision**	**recall**	**f1-score**	**accuracy**
CNN Orig.	0.07	0.09	0.05	0.20
MLP Orig.	0.13	0.12	0.09	0.25
CNN Prop.	0.95	0.93	0.94	0.96
MLP Prop.	0.93	0.89	0.90	0.94
ChatGPT (*baseline*)	0.94	0.91	0.93	0.94

8.7 Discusión

En esta tesis, se investigó el uso de técnicas de procesamiento de lenguaje natural (PLN) para la georreferenciación automática de solicitudes de atención ciudadana. Los modelos implementados, MLP y CNN, mostraron diferentes niveles de efectividad, destacando la importancia del preprocesamiento y la selección de hiperparámetros adecuados. El MLP con la propuesta NER demostró ser particularmente efectivo, lo cual se atribuye a su capacidad para capturar patrones complejos en los datos después de un exhaustivo preprocesamiento.

La propuesta NER se desarrolló para abordar las limitaciones observadas en sistemas anteriores, utilizando técnicas avanzadas de procesamiento de lenguaje natural y aprendizaje profundo. Se realizaron pruebas exhaustivas para evaluar el desempeño de las arquitecturas MLP y CNN con diferentes configuraciones y conjuntos de datos.

Pruebas con Texto Original

En las pruebas iniciales utilizando texto sin procesar, ambos modelos, CNN y MLP, mostraron limitaciones en su rendimiento. La precisión del modelo CNN apenas superaba el 20 %, lo que sugiere que este enfoque tiene dificultades para manejar la complejidad y variabilidad del texto no procesado. El modelo MLP, aunque mejor, también mostró una precisión subóptima, indicando la necesidad de un preprocesamiento más robusto para mejorar la capacidad de generalización.

Pruebas con la Propuesta NER

Al aplicar la propuesta NER, que incluyó técnicas avanzadas de preprocesamiento de datos, se observó una mejora significativa en el rendimiento de ambos modelos. El modelo CNN mostró

un aumento en la precisión, aunque seguía siendo inferior al MLP. Por otro lado, el modelo MLP demostró un notable incremento en precisión y recall, alcanzando niveles significativamente altos en todas las métricas evaluadas. Esto confirma la eficacia del preprocesamiento y la optimización de hiperparámetros propuestos. Comparación y Futuras Direcciones

El éxito del modelo MLP con la propuesta NER puede explicarse por su arquitectura y la estrategia de preprocesamiento utilizada. El preprocesamiento incluyó técnicas como la tokenización, lematización y la eliminación de palabras vacías, que ayudaron a limpiar, el reconocimiento de entidades nombradas y estructurar los datos de manera que los modelos pudieran interpretarlos con mayor precisión. Esta fase es crítica en cualquier aplicación de PLN, ya que la calidad de los datos de entrada afecta directamente el rendimiento del modelo.

A pesar de los buenos resultados obtenidos con el MLP, el modelo CNN no alcanzó el mismo nivel de precisión y recall. Esto podría deberse a varios factores, incluyendo el tamaño del conjunto de datos y la arquitectura específica utilizada. Las CNN son conocidas por su capacidad para detectar características espaciales en datos estructurados, pero pueden no ser tan efectivas con datos textuales sin un preprocesamiento adecuado y una arquitectura optimizada. Sin embargo, la exploración de técnicas de regularización y arquitecturas híbridas podría mejorar su rendimiento en futuras investigaciones.

La evaluación de los modelos mostró que el uso de técnicas de preprocesamiento avanzadas y la optimización de hiperparámetros es esencial para mejorar la precisión de los sistemas de PLN. En particular, la propuesta de preprocesamiento mejoró significativamente el desempeño del modelo MLP, destacando la importancia de estas fases en la implementación de soluciones efectivas. Este hallazgo subraya la necesidad de enfoques integrales que consideren tanto la calidad de los datos como la arquitectura del modelo.

La implementación de estos modelos en sistemas de atención ciudadana puede transformar la manera en que se gestionan las solicitudes. Al mejorar la precisión y rapidez de la clasificación de descripciones de lugares, las autoridades pueden responder de manera más efectiva a las necesidades de los ciudadanos, optimizando los recursos y mejorando la satisfacción general. Este aspecto práctico destaca la relevancia de continuar investigando y desarrollando soluciones basadas en PLN para la gestión urbana.

9. Conclusion

Este documento presentó una implementación y evaluación de dos modelos de clasificación de texto utilizando Python: Multi Layer Perceptron (MLP) y Convolutional Neural Network (CNN). Los modelos se entrenaron y evaluaron en un conjunto de datos de descripciones de solicitudes de locatel en la Ciudad de México, obteniendo resultados de clasificación precisos.

Este estudio presentó una implementación y evaluación detallada de dos modelos de clasificación de texto, MLP y CNN, aplicados a la georreferenciación automática de solicitudes de atención ciudadana en la Ciudad de México. La propuesta incluyó un exhaustivo proceso de preprocesamiento de datos y la optimización de hiperparámetros, con el objetivo de mejorar la precisión y generalización de los modelos de clasificación. Los resultados obtenidos demuestran que el modelo MLP con la propuesta de preprocesamiento supera a los otros modelos en términos de precisión y recall, confirmando la eficacia del enfoque propuesto.

El estudio se enfocó en la implementación y evaluación de una propuesta de sistema de reconocimiento de entidades nombradas (NER) utilizando dos arquitecturas de redes neuronales: multicapa (MLP) y convolucional (CNN). La propuesta NER incluyó un detallado preprocesamiento de datos y la optimización de hiperparámetros para ambas arquitecturas. Los resultados mostraron que el modelo MLP con la propuesta NER superó consistentemente a los modelos MLP y CNN utilizando el texto original y CNN con la propuesta NER en términos de precisión y generalización. Esto sugiere que el enfoque propuesto para el NER es efectivo y robusto, especialmente al utilizar MLP para la clasificación de entidades nombradas en textos.

La investigación resalta la importancia del preprocesamiento de datos en la mejora del rendimiento de los modelos de PLN. La propuesta NER utilizada en este estudio permitió una mejora significativa en la precisión de los modelos, especialmente en el caso del MLP. Estos hallazgos sugieren que el preprocesamiento adecuado y la optimización de hiperparámetros son cruciales para el éxito de las aplicaciones de PLN en contextos prácticos.

Además, se identificaron áreas de mejora para futuras investigaciones. Entre ellas, la integración de técnicas de regularización más eficaces para evitar el sobreajuste en modelos CNN y la exploración de enfoques híbridos que combinen las ventajas de ambas arquitecturas, podrían proporcionar

mejoras adicionales en el rendimiento de los modelos de clasificación. Estas direcciones futuras son esenciales para el desarrollo continuo y la optimización de sistemas de PLN.

La implementación de técnicas de PLN en la georreferenciación automática tiene un impacto directo en la mejora de los sistemas de atención ciudadana. Al proporcionar una clasificación más precisa y rápida de las solicitudes, estas tecnologías pueden facilitar una gestión urbana más eficiente y una mayor satisfacción de los ciudadanos. Este estudio no solo contribuye al conocimiento académico en el campo del PLN, sino que también tiene implicaciones prácticas significativas.

Finalmente los resultados de este estudio demuestran que el uso de modelos de MLP con un preprocesamiento adecuado y la optimización de hiperparámetros ofrece una solución robusta y efectiva para la clasificación de alcaldías a partir de entidades nombradas en textos. Los hallazgos proporcionan una base sólida para futuras investigaciones y desarrollos en el área de la georreferenciación automática y el procesamiento de lenguaje natural. La implementación de estas tecnologías puede transformar la gestión urbana y mejorar significativamente la respuesta a las solicitudes de atención ciudadana.

Bibliografía

Hu, Y. & Adams, B. (2020). Harvesting Big Geospatial Data from Natural Language Texts.

Car, N. J., Box, P. J. & Sommer, A. (2019). The Location Index: A Semantic Web Spatial Data Infrastructure. En P. Hitzler, M. Fernández, K. Janowicz, A. Zaveri, A. J. Gray, V. Lopez, A. Haller & K. Hammar (Eds.), *The Semantic Web* (pp. 543-557). Springer International Publishing.

Aldana-Bobadilla, E., Molina-Villegas, A., Lopez-Arevalo, I., Reyes-Palacios, S., Muñiz-Sanchez, V. & Arreola-Trapala, J. (2020a). Adaptive geoparsing method for toponym recognition and resolution in unstructured text. *Remote Sensing*, *12*(18), 3041. https://doi.org/10.3390/rs12183041

Molina-Villegas, A., Muñiz-Sanchez, V., Arreola-Trapala, J. & Alcántara, F. (2021). Geographic named entity recognition and disambiguation in Mexican news using word embeddings. *Expert Systems with Applications*, *176*, 114855. https://doi.org/10.1016/j.eswa.2021.114855

Molina-Villegas, A., Aldana-Bobadilla, E., Siordia, O. & Perez, J. (2022). Incorporating Natural Language Processing models in Mexico City's 311 Locatel. *LatinX in AI (LXAI) Research Workshop 2022*. https://doi.org/10.52591/lxai202207101

Aldana-Bobadilla, E., Molina-Villegas, A., Lopez-Arevalo, I., Reyes-Palacios, S., Muñiz-Sanchez, V. & Arreola-Trapala, J. (2020b). Adaptive Geoparsing Method for Toponym Recognition and Resolution in Unstructured Text. *Remote Sensing*, *12*(18). https://doi.org/10.3390/rs12183041

Abelleira, M. A. P. & Cardoso, C. A. (2010). Minerıa de texto para la categorización automática de documentos. *PhD in Computer Science por Carnegie Mellon University, Madrid, España*.

Alexopoulos, P., Ruiz, C. & Gomez-Perez, J. (2012). Optimizing geographical entity and scope resolution in texts using non-geographical semantic information. *Proceedings of the 6th International Conference on Advances in Semantic Processing, SEMAPRO*, 65-70.

Gritta, M., Pilevar, M. T. & Collier, N. (2018). A pragmatic guide to geoparsing evaluation: Toponyms, Named Entity Recognition and pragmatics. *Language Resources and Evaluation*, *52*, 603-623.

Gritta, M., Pilevar, M. T. & Collier, N. (2019). A pragmatic guide to geoparsing evaluation: Toponyms, Named Entity Recognition and pragmatics. *Language Resources and Evaluation*, *54*. https://doi.org/10.1007/s10579-019-09475-3

Gelernter, J. & Zhang, W. (2013). Cross-lingual geo-parsing for non-structured data. *Proceedings of the 7th Workshop on Geographic Information Retrieval*, 64-71.

Moncla, L., Renteria-Agualimpia, W., Nogueras-Iso, J. & Gaio, M. (2014). Geocoding for texts with fine-grain toponyms: an experiment on a geoparsed hiking descriptions corpus. *Proceedings of the 22nd ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, 183-192.

Allen, J. (1995). *Natural Language Understanding*. Benjamin-Cummings Publishing Co.

Manning, C. D. & Schütze, H. (1999). *Foundations of Statistical Natural Language Processing*. MIT Press.

Devlin, J., Chang, M.-W., Lee, K. & Toutanova, K. (2019). BERT: Pre-training of deep bidirectional transformers for language understanding. *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, 4171-4186.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł. & Polosukhin, I. (2017). Attention is all you need. *Advances in neural information processing systems*, *30*.

Brown, T. B., Mann, B., Ryder, N., Subbiah, M. et al. (2020). Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*.

Jurafsky, D. & Martin, J. H. (2009). *Speech and Language Processing*. Pearson.

Greengrass, E. (2000). Information retrieval: A survey.

Orallo, J. H., Quintana, M. J. R. & Ramırez, C. F. (2004). *Introducción a la Mineria de Datos*. Pearson Educación.

Wang, Y. (2018). Named Entity Recognition with Bidirectional LSTM-CNNs. *Information Sciences*, *438-439*, 1-12.

Goodfellow, I., Bengio, Y. & Courville, A. (2016). *Deep Learning*. MIT Press.

Bishop, C. M. (2006). *Pattern Recognition and Machine Learning*. Springer.

Haykin, S. (1999). *Neural Networks: A Comprehensive Foundation*. Prentice Hall.

Schmidhuber, J. (2015). Deep learning in neural networks: An overview. *Neural Networks*, *61*, 85-117.

Haykin, S. (2009). *Neural Networks and Learning Machines*. Pearson Education.

LeCun, Y., Bottou, L., Bengio, Y. & Haffner, P. (1998). Gradient-Based Learning Applied to Document Recognition. *Proceedings of the IEEE*, *86*(11), 2278-2324.

Krizhevsky, A., Sutskever, I. & Hinton, G. E. (2012). ImageNet Classification with Deep Convolutional Neural Networks. *Advances in Neural Information Processing Systems*, 1097-1105.

Hochreiter, S. & Schmidhuber, J. (1997). Long Short-Term Memory. *Neural Computation*, *9*(8), 1735-1780.

Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S. & Dean, J. (2013). Efficient Estimation of Word Representations in Vector Space. *Proceedings of Workshop at ICLR*.

Pennington, J., Socher, R. & Manning, C. D. (2014). GloVe: Global vectors for word representation. *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, 1532-1543.

Bojanowski, P., Grave, E., Joulin, A. & Mikolov, T. (2017). Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, *5*, 135-146.

Mitchell, T. M. (1997). *Machine learning*. McGraw-Hill.

Hastie, T., Tibshirani, R. & Friedman, J. (2009). *The elements of statistical learning: Data mining, inference, and prediction*. Springer.

Bengio, Y., LeCun, Y. & Hinton, G. (2015). Deep Learning. *Nature*, *521*(7553), 436-444.

Lipton, Z. C. & Steinhardt, J. (2019). Troubling Trends in Machine Learning Scholarship. *arXiv preprint arXiv:1903.07000*.

Graves, A. & Schmidhuber, J. (2005). Framewise phoneme classification with bidirectional LSTM and other neural network architectures. *Neural Networks*, *18*(5-6), 602-610.

Sokolova, M. & Lapalme, G. (2009). A systematic analysis of performance measures for classification tasks. *Information Processing & Management*, *45*(4), 427-437.

James, G., Witten, D., Hastie, T. & Tibshirani, R. (2013). *An Introduction to Statistical Learning*. Springer.

Fawcett, T. (2006). An introduction to ROC analysis. *Pattern Recognition Letters*, *27*(8), 861-874.

Smith, A. & Jones, B. (2019). Generating Geohashes from Text: A Sequence-to-Sequence Approach. *Journal of Geospatial Intelligence*, 45-62.

Sutskever, I., Vinyals, O. & Le, Q. V. (2014). Sequence to sequence learning with neural networks. *Advances in neural information processing systems*, 3104-3112.

Niemeyer, G. (2008). Geohash.