INSTITUTO POLITÉCNICO NACIONAL

ESCUELA SUPERIOR DE CÓMPUTO

Trabajo Terminal.

Generador de versos musicales en el idioma inglés por medio de procesamiento de lenguaje natural y redes neuronales

TT2020-B002.

Presentan:
Espinosa de los Monteros Lechuga Jaime Daniel
Nava Romo Edgar Adrián
Salgado Gómez Alfredo Emilio

Directores:
Olga Kolesnikova
Ariel López Rojas

Índice.

Índice de figuras.

Índice de cuadros.

Parte I

Trabajo Terminal I

Capítulo 1

Introducción

La industria musical obtiene ganancias a través de la creación y divulgación de la música de manera física y digital (Bourreau and Gensollen 2006 [5]), dejando que aficionados y emprendedores de la música no tengan oportunidad de avanzar en su carrera por falta de creatividad, tiempo y/o recursos, haciendo que la creación de nuevas letras para sus canciones sea un gran obstáculo, nuestra propuesta implica la utilización de nuevas tecnologías que permitan la generación de letras para integrar con sus canciones. Esta es una de las tareas más populares y desafiantes en el área de procesamiento del lenguaje natural. Hay una gran cantidad de trabajos (Generating Text with Recurrent Neural Networks [6], Convolutional Neural Networks for Sentence Classification[7]) que proponen generar texto utilizando redes neuronales recurrentes y/o convolucionales. Sin embargo, la mayoría de los trabajos actuales solo se enfocan en generar una o varias oraciones, ni siquiera un párrafo largo y mucho menos una canción completa.

Las letras de canciones, como un tipo de texto, tienen algunas características propias, estas se constituyen de rimas, versos, coros y en algunos casos, patrones de repetición. Coro se refiere a la parte de una canción que se repite sin modificaciones dentro de la misma después de un verso, mientras que en el verso suelen cambiar una o varias líneas que lo componen. Estas características particulares hacen que generar letras musicales sea mucho más difícil que textos normales.

La mayoría de las investigaciones actuales sobre generación de letras vienen con muchas condiciones, como dar una pieza de melodía (Automatic Generation of Melodic Accompaniments for Lyrics [8]), o solo generar un tipo específico de letra (Conditional Rap Lyrics Generation with Denoising Autoencoders [9]). Sin embargo, la generación de letras por medio de in-

teligencia artificial dado un estilo y tema en particular, ha sido muy poco trabajado y debido a esto planeamos centrarnos en este nuevo problema. Estamos interesados en ver si el modelo propuesto puede aprender diferentes características de un género musical y generar letras que sean acorde a este. Actualmente, en el mercado se encuentran cuatro aplicaciones web que tienen una funcionalidad similar a la propuesta en este Trabajo Terminal:

- These lyrics do not exist.
- Bored humans - lyrics_generator.
- DeepBeat.
- Premium Lyrics.

En el cuadro 1 que se presenta a continuación, se muestran las características de aplicaciones web similares y comparándolas con nuestra propuesta:

Software	Características	Precio
These Lyrics do not Exist	Aplicación web que genera letras completamente originales de varios temas, hace uso de IA para generar coros y versos originales; se puede escoger el tema principal de la letra, género musical e incluso el estado de ánimo al que iría dirigido.	Gratuito (Contiene Anuncios)
Boredhumans lyrics_generator	Aplicación web en el que la IA fue entrenada con una base de datos con miles de letras para generar letras de canciones totalmente nuevas. La letra que crea es única y no una copia de alguna que exista actualmente, sin embargo, no permite customizar la letra.	Gratuito
DeepBeat	Aplicación web que por medio de IA genera letras de música enfocada principalmente en el género rap. Si una línea no es del agrado se puede sustituir por alguna de las otras propuestas de las que ofrece.	Gratuito
Premium Lyrics	Aplicación web que proporciona versos compuestos en distintos idiomas por artistas independientes que se escogen manualmente de acuerdo a su originalidad y calidad.	$3 a $75 Dólares por letra musical
Nuestra propuesta	Aplicación web que haciendo uso de una IA va a generar letras musicales originales a partir de un género musical en exclusivo, lo que asegurará un resultado original con coros y versos distintos cada vez que se utilice.	Gratuito

Cuadro 1.1: Resumen de productos similares comparados con nuestra propuesta

1.1. Objetivos

1.1.1. Objetivo general

Crear una herramienta de apoyo para estudiantes o aficionados interesados en este rubro que se les dificulte componer nuevas letras musicales de un solo género musical debido a la carencia de creatividad, la falta de conocimientos en la estructura del género o que no tengan inspiración suficiente para poder crear nuevas canciones, esto con el fin de impulsar la carrera de futuros artistas en la industria musical que no tengan los suficientes recursos para poder contratar servicios particulares de compositores.

1.1.2. Objetivos particulares

- Generar un conjunto de datos (dataset) con letras musicales de un género musical para efecto de entrenamiento de la red semántica.
- Hacer uso de alguna herramienta de aprendizaje automático (machine learning) e implementar su uso en la nube para ayudar a procesar las letras musicales de un género en específico.
- Implementar un módulo analizador de semántica para entrenar redes neuronales.
- Desarrollar una interfaz web intuitiva en versión prototipo que utilice una aplicación web alojada en la nube para la visualización del verso musical generado a partir de un género.

1.2. Justificación.

El crear nuevas composiciones musicales puede llegar a ser muy difícil, estresante e incluso agotador para cualquier aficionado o incluso algunos expertos en este medio, esto se debe a la falta de creatividad y/o tiempo de quien lo quiera realizar [10]. En ocasiones se pueden contratar servicios particulares para la producción de la letra de una canción, sin embargo, puede ser muy costoso y en ocasiones el resultado final no alcanza a llenar las expectativas de la inversión que se hace; por ende, se pretende crear una herramienta para estudiantes, aficionados o cualquier persona interesada en este rubro que se les dificulte componer nuevas letras musicales.

Normalmente las letras musicales creadas por el humano, tienden a estar

compuestas por patrones de acuerdo al género musical [11]. Algunos ejemplos de estos patrones pueden ser las rimas, enunciados, frases cortas y que tengan una semántica correcta, estos pueden ser encontrados por medio de procesamiento de lenguaje natural y una investigación profunda en la composición de letras de estos géneros.

Se eligió el idioma inglés debido a que existe una gran cantidad de bases de datos para procesar, al igual que herramientas y documentación para este idioma.

Nos proponemos orientar esta solución en un entorno de nube, donde la información de configuración, servicios y datos necesarios pueden mantenerse de forma independiente a la implementación, facilitando la adaptación y flexibilidad de la plataforma.

Nuestro proyecto ayudará al usuario utilizando herramientas como el procesamiento de lenguaje natural, redes neuronales, aprendizaje de máquina (machine learning) y servidores en la nube. Se hará uso de un conjunto de datos (dataset) y herramientas alojadas en la nube (Google Cloud Platform o Amazon Web Services) para procesar estos datos; se pretende utilizar un módulo que encuentre patrones por medio de redes neuronales para analizar la semántica mediante técnicas y herramientas ya existentes de procesamiento de lenguaje natural. Se van a realizar pruebas y experimentos con estas herramientas antes de la implementación (Bert [12], spaCy[13]) para poder generar versos. A su vez se va a desarrollar una interfaz web intuitiva en versión prototipo donde el usuario va a poder utilizar esta herramienta la cual le va a mostrar la letra musical que se va a generar en ese momento.

A diferencia de los proyectos señalados en la Tabla 1 nuestra propuesta se va a centrar en generar letras musicales con métodos y tecnologías distintas a los que se usaron, esto es, aunque se utilicen los mismos géneros musicales se tendrán resultados completamente diferentes con propuestas distintas.

En el desarrollo de este proyecto haremos uso de los conocimientos adquiridos durante el transcurso de la carrera. Se van a utilizar técnicas de diseño de proyectos aprendidas en el curso de Ingeniería de Software, se van a aplicar los conocimientos de programación adquiridos en unidades de aprendizaje como Inteligencia Artificial, Procesamiento de Lenguaje Natural, Web Application Development, Programación Orientada a Objetos, Análisis de Algoritmos, así como técnicas de construcción de documentos y análisis de semántica vistas en Análisis y Diseño orientado a Objetos y Comunicación Oral y Escrita.

1.3. Metodología

Para el desarrollo de este trabajo terminal se utilizará la metodología ágil Scrumban, que combina algunas partes de la metodología Scrum y Kanban, debido a que este proceso de gestión reduce la complejidad al momento de desarrollar un producto el cual tratará de satisfacer las necesidades de los clientes. Además, permite trabajar colaborativamente de manera eficiente, es decir, en equipo, para obtener el mejor resultado posible.

Scrumban combina la estructura utilizada por Scrum, con los métodos basados en el flujo y visualización de Kanban. Es decir, que permite a los equipos trabajar de manera ágil usando la metodología Scrum y la simplicidad de Kanban sin tener que utilizar las actualizaciones de roles y es más sencillo de adoptar.

En la siguiente tabla se pueden observar las principales diferencias entre las tres metodologías:

	Scrum	Kanban	Scrumban
Procesos	Iterativo e incremental desarrollando sprints	Continuo	Iterativo e incremental de forma continua desarrollando iteraciones
Personas	Las personas son el centro	Las personas son el pilar	Equipo motivado con personas como pilar y en el centro
Producto	Foco en la efectividad	Foco en la eficiencia	Balance entre efectividad y eficiencia
Organización	Mejora continua del producto	Mejora continua del proceso	Mejora continua del producto y del proceso
Equipo	De 3 a 9 personas	No hay limitaciones	El equipo no requiere de un número específico de integrantes
Roles	Scrum master, Product owner, Scrum team	Pueden incluir especialistas o integrantes generalizados	No requiere un rol específico

Cuadro 1.2: Tabla comparativa de las distintos metodologías ágiles contempladas

Scrumban hace uso de iteraciones, las cuales monitorea con el apoyo de un recurso visual, como lo puede ser un tablero. Las reuniones para planificar se llevan a cabo cuando son necesarias para determinar las tareas a implementar hasta la próxima iteración. Para que estas iteraciones se mantengan cortas, se utiliza un límite de trabajo en progreso (WIP por sus siglas en inglés Work in Progress). Cuando WIP desciende de cierto nivel, se establece una acción para que el equipo sepa cuándo y qué tarea planificar a continuación.

Iteración
En Scrumban, las iteraciones son cortas para garantizar que el equipo pueda adaptarse al entorno cambiante durante el proyecto. La duración de estas iteraciones en este proyecto se medirán como máximo en lapsos de dos semanas.

Priorización
La priorización se da de tal forma que las tareas más importantes se colocan en la parte superior de la tabla de planificación seguidas por las tareas menos importantes.
Antes de llegar al tablero las tareas deben pasar por 3 etapas donde se van depurando las tareas a realizar para largo plazo (1 año), medio plazo (6 meses) y corto plazo (3 meses) siendo de esta última de donde salen las tareas más claras que se pueden completar y que ganan mayor prioridad para la próxima iteración.

Principio de Elección
Cada miembro del equipo elige qué tarea de la sección "Tareas Pendientes" va a completar a continuación.

Congelación de Funciones
Se utiliza cuando se aproxima la fecha límite del proyecto, significando que sólo pueden trabajar sobre las tareas previamente pensadas sin cabida para implementar nuevas características.

Triaje
Ocurre después de la congelación de funciones, y es el punto donde el líder del proyecto decide cuáles características en desarrollo se completarán y cuáles quedarán sin terminar.

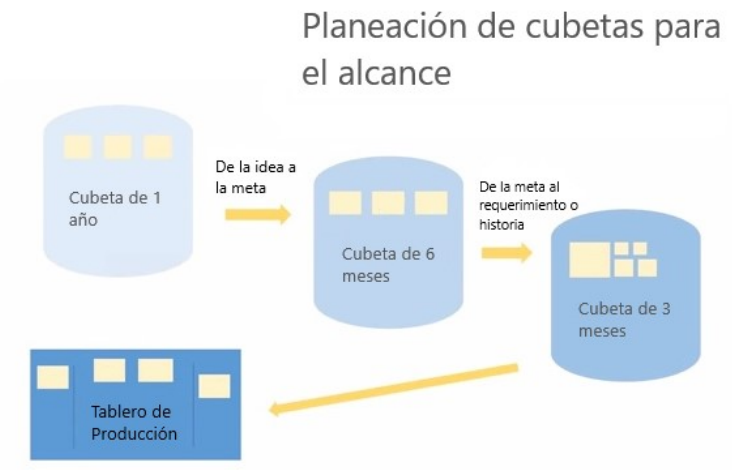


Figura 1.1: Funcionamiento de la planificación por cubetas

1.4. Organización del documento

Para dar inicio a este trabajo terminal, presentamos de manera breve la estructura de este reporte, con el objetivo de que el lector pueda tener un mejor entendimiento del trabajo.

1.4.1. Capítulo 2. Marco teórico

En esta parte del documento, se describen puntos esenciales de nuestro trabajo como definiciones, técnicas, herramientas, servicios, así como investigación realizada para llevar a cabo en la implementación dentro del trabajo.

1.4.2. Capítulo 3. Análisis.

Dentro de este capitulo se analiza el estudio de factibilidad tanto técnico como operativo y económico con la finalidad de conocer los recursos necesarios para la elaboración de este trabajo terminal. Se mencionan resumidamente las herramientas a utilizar y se explica de manera general la arquitectura del sistema y el diagrama de casos de uso general. Finalmente se muestra el análisis de los tres componentes que tenemos en el sistema.

1.4.3. Capítulo 4. Diseño

En el cuarto capítulo, nos adentraremos en el desarrollo de los prototipos, es decir, se encuentran los diagramas pertinentes para poder modelar nuestro trabajo terminal y proceder a la etapa de implementación. En este capítulo se desarrollan y explican los siguientes diagramas: 'Casos de uso', 'Flujo', 'Flujo de datos', 'Clases', 'Secuencia' y 'Actividades' y se muestra la interfaz de usuario propuesta junto con los requisitos de diseño.

Capítulo 2

Marco teórico

2.1. Inteligencia Artificial

Por Inteligencia Artificial se entiende a una simulación de procesos computacionales cognitivos para que simulen el comportamiento de una mente humana, estos comportamientos abarcan diferentes áreas de investigación, como el razonamiento, aprendizaje, percepción, comunicación y la capacidad de desarrollarse en entornos más complejos. “La inteligencia no es una dimensión única, sino un espacio profusamente estructurado de capacidades diversas para procesar la información. Del mismo modo, la Inteligencia Artificial utiliza muchas técnicas diferentes para resolver una gran variedad de tareas que se encuentran en todas partes.” [14]

En ciencias de la computación, el término de inteligencia artificial hace referencia a cualquier inteligencia semejante a la humana presentado por una computadora o un equipo electrónico. De manera popular, la inteligencia artificial hace referencia a la capacidad de una computadora o máquina de imitar las facultades del cerebro humano, es decir, que es capaz de aprender de ejemplos y experiencias, así como reconocer objetos y clasificarlos, tomar decisiones, resolver problemas, entender y responder al lenguaje, igualmente combinar estas y otras capacidades para realizar funciones similares que un ser humano podría realizar. [15]

Como objetivo, la Inteligencia Artificial pretende hacer un uso de los recursos tecnológicos para desarrollar modelos computacionales y poder resolver problemas del mundo real evolucionando constantemente.

2.2. Redes neuronales

Una red neuronal es una herramienta derivada de la inteligencia artificial que utiliza modelos matemáticos para ser utilizada como un mecanismo de predicción de texto.

Las redes neuronales son una forma de hacer que las computadoras aprendan, donde la computadora aprende a realizar alguna tarea analizando ejemplos de entrenamiento. Por lo general, estos ejemplos han sido etiquetados previamente.

Modelado vagamente en el cerebro humano, una red neuronal consiste en miles de millones de nodos de procesamiento los cuales están densamente interconectados. En la actualidad las redes neuronales están organizadas como capas de nodos, y estas capas están "alimentadas hacia adelante" (feed-forward), es decir, la información que se mueve a través de ellas solo fluye en una sola dirección. Un solo nodo puede estar conectado a muchos nodos en capas inferiores de las cuales recibe información y a su vez, puede estar conectado a nodos en capas superiores a los cuales envía información.

Cuando una red neuronal está siendo entrenada la información de entrenamiento pasa a alimentar las capas inferiores (capas de entrada), donde será procesada y pasada a capas posteriores donde será transformada hasta llegar a las capas superiores (capas de salida).[16]

2.2.1. Neuronas

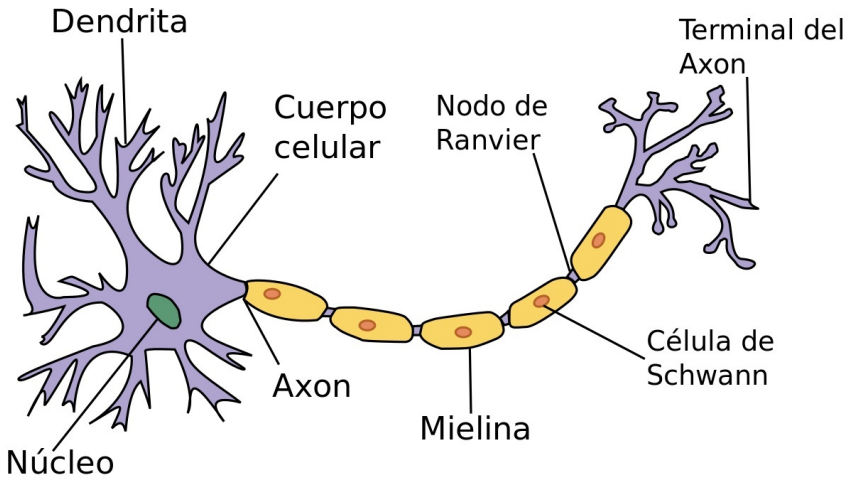
Neurona Natural

La neurona es una de muchas células la cual representa a la unidad estructural y funcional del sistema nervioso. Esta trabaja transmitiendo la información a través de impulsos nerviosos o químicos, desde un lugar del cuerpo hacia otra parte de este.

Dicho impulso nervioso o impulso eléctrico viaja siempre en un mismo sentido, es decir, llega a la neurona a través de las dendritas, se procesa en el soma y posteriormente se transmite al axón, las neuronas no están en contacto entre sí, entre ellas existe un espacio de separación denominado sinapsis o espacio sináptico. Una vez que el impulso nervioso llega al extremo final del axón este libera neurotransmisores al espacio sináptico transformando esta señal eléctrica en una señal química la cual se introduce en la dendrita

de la neurona contigua, desencadenando un impulso eléctrico en la neurona receptora, repitiendo este proceso n veces hasta llegar a su destino final.[17]

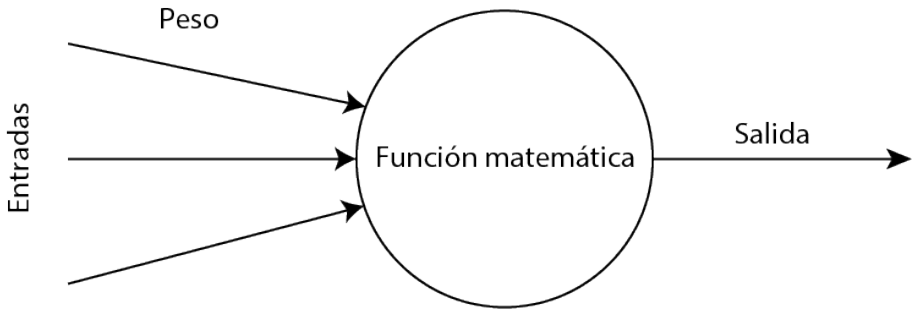
Figura 2.1: Estructura de una neurona [18]



Neurona Artificial

La complejidad de las neuronas reales se abstrae mucho cuando se realiza el modelado de neuronas artificiales. Estas consisten básicamente en entradas, que se multiplican por pesos (fuerza de las respectivas señales), y luego calculada por una función matemática que determina la activación de la neurona. Otra función (que puede ser la identidad) calcula la salida de la neurona artificial (a veces en dependencia de un cierto umbral). Las redes neuronales combinan neuronas artificiales para procesar información.[19]

Figura 2.2: Neurona Artificial



2.2.2. Aprendizaje de las Redes Neuronales

Si observamos la naturaleza, podemos ver que los sistemas que pueden aprender son altamente adaptables. En su búsqueda por adquirir conocimientos, estos sistemas utilizan información del mundo exterior y modifican la información que ya han recopilado o modifican su estructura interna. Eso es exactamente lo que hacen las redes neuronales. Adaptan y modifican su arquitectura para aprender. Para ser más precisos, las redes neuronales cambian los pesos de las conexiones según la entrada y la salida deseada.

¿Por qué tienen un peso?, bueno, si observamos la estructura de las redes neuronales, hay algunos componentes que podríamos cambiar, si queremos modificar su arquitectura. Por ejemplo, podríamos crear nuevas conexiones entre neuronas, o eliminarlas, o agregar y eliminar neuronas. Incluso podríamos modificar la función de entrada o la función de activación. Resulta que cambiar los pesos es el enfoque más práctico. Además, la mayoría de los otros casos podrían cubrirse cambiando los pesos. La eliminación de una conexión, por ejemplo, se puede hacer estableciendo el peso en 0. Y una neurona se puede eliminar si establecemos los pesos de todas sus conexiones en cero.[20]

El entrenamiento es un proceso necesario para toda red neuronal, y es un proceso en el que la red se familiariza con el problema que necesita resolver. En la práctica, generalmente se tienen algunos datos recopilados en función de los cuales necesitamos crear nuestras predicciones, clasificación, o cualquier otro procesamiento. Estos datos se denominan conjunto de entrenamiento. Según el comportamiento durante el entrenamiento y la naturaleza del conjunto de entrenamiento, tenemos algunos tipos de aprendizajes:

- **Aprendizaje no supervisado:** el conjunto de entrenamiento solo contiene entradas. La red intenta identificar entradas similares y clasificarlas en ciertas categorías.
- **Aprendizaje reforzado:** el conjunto de entrenamiento contiene entradas, pero la red también recibe información adicional durante la formación. Lo que sucede es que una vez que la red calcula la salida para una de las entradas, proporcionamos información que indica si el resultado fue correcto o incorrecto y, posiblemente, la naturaleza del error que cometió la red.
- **Aprendizaje supervisado:** el conjunto de entrenamiento contiene entradas y salidas deseadas. De esta manera, la red puede verificar su

salida calculada con la salida deseada y tomar las acciones pertinentes para reformular su cálculo.

2.2.3. Recurrent Neural Network – Long Short Term Memory

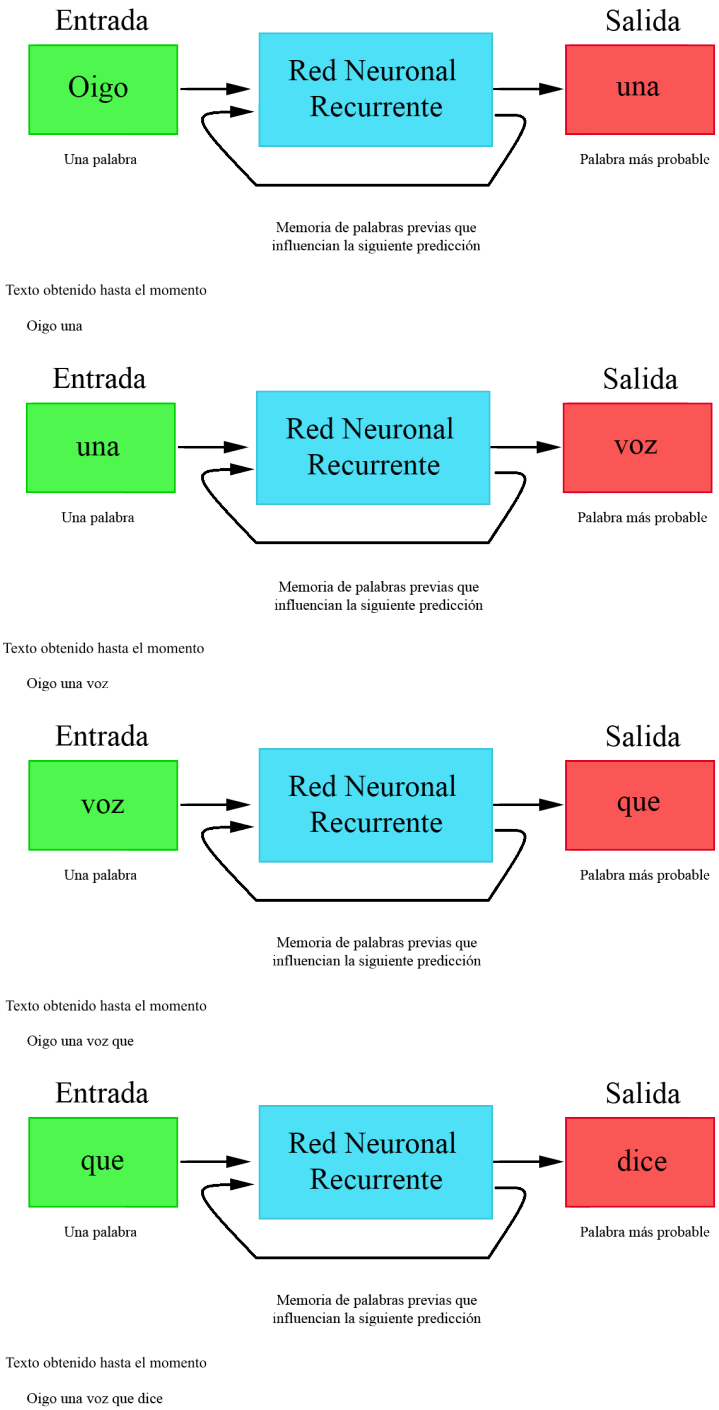
Una red neuronal recurrente es un tipo de red neuronal artificial donde la salida de alguna capa en particular es salvada y sirve para retroalimentar la entrada de esta capa, lo cual ayuda a predecir futuras salidas de esta.

La primera capa esta formada de la misma manera que la Feedforward Neural Network, es decir, solo pasa la información que entra a la siguiente capa inmediata, posteriormente la siguiente capa con el paso del tiempo comenzará a retroalimentarse, pero manteniendo la propagación frontal. Haciendo uso de esta retroalimentación la capa en futuras operaciones puede realizar predicciones, si estas predicciones no son los resultados esperados, el sistema aprende y trabaja para corregir sus futuras predicciones.[21]

Se distinguen por su "memoria", ya que toman información de entradas anteriores para influir en la entrada y salida actuales. Mientras que las redes neuronales profundas tradicionales asumen que las entradas y salidas son independientes entre sí, la salida de las redes neuronales recurrentes depende de los elementos anteriores dentro de la secuencia. Si bien los eventos futuros también serían útiles para determinar la salida de una secuencia dada, las redes neuronales recurrentes unidireccionales no pueden tener en cuenta estos eventos en sus predicciones.[24]

Estos algoritmos de aprendizaje profundo se utilizan comúnmente para problemas relacionados con la traducción de idiomas, el procesamiento del lenguaje natural, el reconocimiento de voz y los subtítulos de imágenes.

Figura 2.3: Diagrama Red Neuronal Recurrente



2.3. Procesamiento de Lenguaje Natural

El Procesamiento del Lenguaje Natural (PLN) se refiere a una rama de la informática, específicamente, a una rama de la inteligencia artificial, la cual se ocupa de dar a las computadoras la capacidad de comprender texto y palabras de la misma manera que los seres humanos.

El Procesamiento del Lenguaje Natural (PLN) combina la lingüística computacional con modelos estadísticos, machine learning y deep learning. Juntas, estas tecnologías permiten a las computadoras procesar el lenguaje humano en forma de texto o datos de voz y 'comprender' su significado completo, es decir, la intención y el sentimiento del hablante o escritor. [25]

2.3.1. Tareas del Procesamiento de Lenguaje Natural

- **Reconocimiento de voz:** La tarea pertinente al reconocimiento de voz es la conversión de la voz a texto, requiere convertir de manera confiable datos de voz en datos de texto. Lo que hace que el reconocimiento de voz sea especialmente desafiante es la forma en que las personas hablan: velocidad, arrastrando las palabras, con diferentes énfasis y entonación, así como con diferentes acentos.
- **Etiquetado de palabras:** Es el proceso de determinar la parte gramatical de una palabra o fragmento de texto en particular, en función de su uso y contexto.
- **Desambiguación de las palabras:** Es la selección del significado de una palabra con múltiples significados a través de un proceso de análisis semántico que determina la palabra que tiene más sentido en el contexto dado.
- **Reconocimiento de entidad nombrada:** Identifica palabras o frases como entidades útiles, es decir, identifica nombres propios.
- **Resolución de correferencia:** Es la tarea de identificar si dos palabras se refieren a la misma entidad. El ejemplo más común es determinar la persona u objeto al que se refiere un determinado pronombre.
- **Análisis de sentimientos:** Intenta extraer del texto cualidades subjetivas (actitudes, emociones, sarcasmo, confusión, sospecha).
- **Generación de lenguaje natural:** Es la tarea de convertir información estructurada a lenguaje humano.

2.3.2. Usos del Procesamiento de Lenguaje Natural

El procesamiento del lenguaje natural es el cerebro detrás de la inteligencia artificial en muchas aplicaciones del mundo real. Algunos ejemplos son los siguientes:

- **Aplicaciones dedicadas a la traducción:** un ejemplo de estos es Google Translate, el cual hace uso de BERT para el procesamiento del lenguaje natural así como para el etiquetado de las palabras que requiere traducir.
- **Asistentes virtuales y chatbots:** Asistentes como Siri o Alexa requieren de un reconocimiento de voz para distinguir comandos, así como para responder adecuadamente a estos. En el caso de los chatbots realizan un análisis contextual de las preguntas para proveer respuestas relacionadas a las mismas.
- **Análisis de sentimientos en redes sociales:** Consta de analizar el lenguaje utilizado en publicaciones en redes sociales, respuestas, reseñas y más para extraer actitudes y emociones en respuesta a productos, promociones y eventos; información que las empresas pueden usar en diseños de productos, campañas publicitarias y más
- **Resúmenes de textos:** Consta de realizar resúmenes y sinopsis de distintos textos.

2.4. Transformers

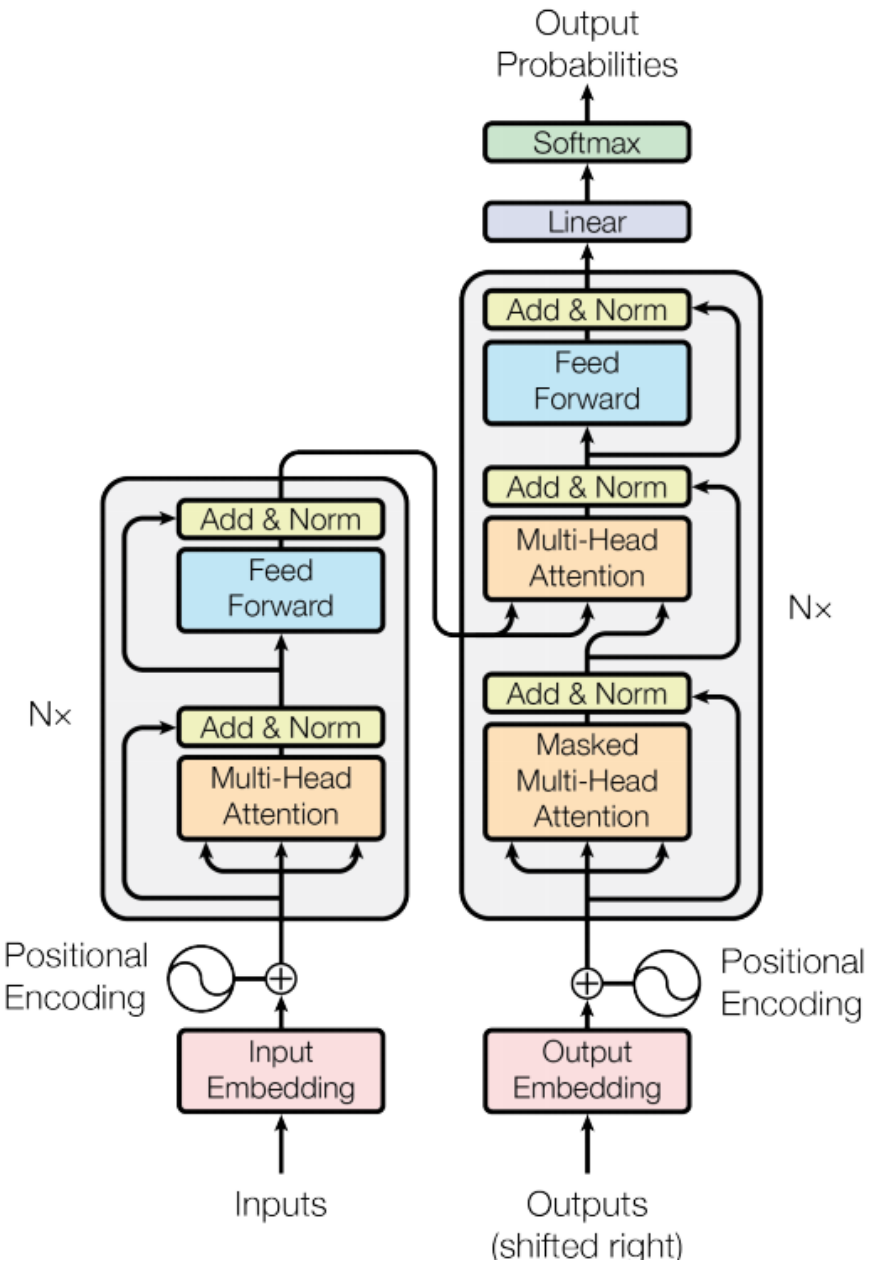
El Transformer en Procesamiento del Lenguaje Natural (PLN) es una arquitectura que tiene como objetivo codificar cada palabra que compone una frase en función a la secuencia que cada palabra sigue, permitiendo así agregar un contexto o una representación matemática dentro del texto.

Se suele trabajar con los transformadores en 2 etapas:

- **Pre-entrenamiento:** En esta etapa, el modelo aprende cómo se estructura el lenguaje de forma general, así como de obtener un conocimiento general del significado de las palabras dentro de la frase trabajada.[26]
- **Afinado:** En esta etapa se añaden ciertas capas o funciones a la arquitectura para adaptar los modelos y que estos realicen ciertas tareas, para después estos modelos ser reentrenados y cumplan con las tareas establecidas de manera correcta.[26]

2.4.1. Arquitectura de un Transformer

Figura 2.4: Arquitectura general de un Transformer [27]



En esta arquitectura el codificador mapea una secuencia de entrada de representaciones de símbolos (x1, ..., xn) a una secuencia de representaciones continuas z = (z1, ..., zn). Dado z, el decodificador genera una salida

secuencia (y1, ..., ym) de símbolos de un elemento a la vez. En cada paso, el modelo es autorregresivo, consumiendo los símbolos generados previamente como entrada adicional para generar el siguiente.

El transformer sigue esta arquitectura general usando auto-atención, es decir, las capas están completamente conectadas tanto para el codificador como para el decodificador, que se muestran en las mitades izquierda y derecha de la Figura anterior, respectivamente. [27]

2.5. BERT

2.5.1. ¿Qué es Bert?

Bidirectional Encoder Representations from Transformers (BERT) es un marco de aprendizaje automático de código abierto para el Procesamiento del Lenguaje Natural (PLN). BERT, significa Representaciones de codificador bidireccional de transformers, basado en transformers, el cual es un modelo de aprendizaje profundo en el que cada elemento de salida está conectado a cada elemento de entrada y las ponderaciones entre ellos se calculan dinámicamente en función de su conexión. [28]

Al contar con la capacidad bidireccional, BERT está previamente entrenado en dos tareas de PLN diferentes, pero relacionadas: el modelado de lenguaje enmascarado y la predicción de la siguiente oración.

El objetivo del entrenamiento del Modelado de Lenguaje Enmascarado (MLM) es ocultar una palabra en una oración y luego hacer que el programa prediga qué palabra se ha ocultado en función del contexto de la palabra oculta. El objetivo del entrenamiento de predicción de la siguiente oración es que el programa prediga si dos oraciones dadas tienen una conexión lógica secuencial o si su relación es simplemente aleatoria.

2.5.2. ¿Cómo funciona BERT?

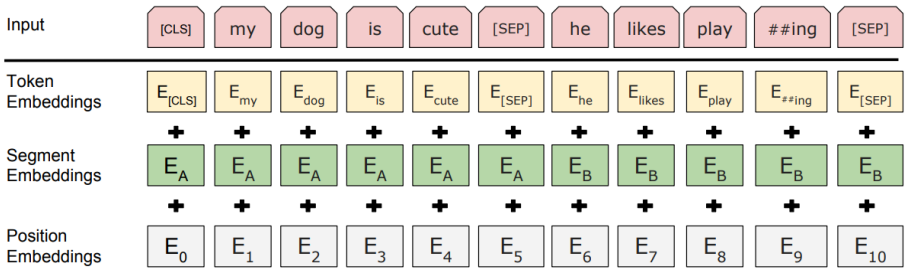
El objetivo de cualquier técnica de PLN es comprender el lenguaje humano tal como se habla de forma natural. En el caso de BERT, esto normalmente significa predecir la siguiente palabra. Para hacer esto, los modelos normalmente necesitan entrenarse usando un gran repositorio de datos de

entrenamiento especializados y etiquetados.

BERT, sin embargo, fue entrenado previamente usando solo un corpus de texto plano sin etiquetar (textos de Wikipedia en inglés). Continúa aprendiendo sin supervisión del texto sin etiquetar y mejorando incluso cuando se usa en aplicaciones prácticas. Su entrenamiento previo sirve como una capa base de "conocimiento". A partir de ahí, BERT puede adaptarse al creciente cuerpo de contenido y consultas para ajustarse a las especificaciones del usuario. Este proceso se conoce como aprendizaje por transferencia.

BERT es posible gracias a la investigación de Google sobre transformers. El transformer es la parte del modelo que le da a BERT su mayor capacidad para comprender el contexto y la ambigüedad en el lenguaje. El transformer hace esto procesando cualquier palabra dada en relación con todas las demás palabras en una oración, en lugar de procesarlas una a la vez.

Figura 2.5: Representación de entrada de BERT y la separación de la oración por palabras y la asignación de valores.[28]



En la figura anterior, lo que se realiza es que se inserta un token [Classification (CLS)] al principio de la primera oración y un token [Separate (SEP)] al final de cada oración, cada palabra es separada en una ficha, a cada ficha se agrega una inserción que indica si se trata de la oración A o la oración B, finalmente se agrega una incrustación posicional a cada token para indicar su posición en la secuencia. [29]

Esto contrasta con el método tradicional de procesamiento del lenguaje, conocido como incrustación de palabras, en el que los modelos mapeaban cada palabra en un vector, que representa sólo una dimensión, es decir, el significado de esa palabra.

2.5.3. BERT en la actualidad

BERT se utiliza actualmente en Google para optimizar la interpretación de las búsquedas de los usuarios.

Así como en la realización de tareas tales como: [30]

- Generación de lenguaje basadas en secuencia a secuencia (Respuesta a preguntas, resúmenes de documentos, predicción de siguiente oración, chatbots).
- Comprensión del lenguaje natural (Entendimiento de la polisemia, la correferencia, la desambiguación de las palabras, clasificación de la palabras por sentimientos).

2.6. GPT-2

2.6.1. ¿Qué es GPT-2?

Generative Pretrained Transformer (GPT) es un transformer aprovechado para realizar tanto aprendizaje supervisado como aprendizaje no supervisado para el Procesamiento del Lenguaje Natural (PLN).

GPT-2 es el sucesor de GPT, es un gran modelo de lenguaje basado en transformers con 1500 millones de parámetros, entrenado en un conjunto de datos de 8 millones de páginas web. GPT-2 se entrena con un objetivo simple: predecir la siguiente palabra, dadas todas las palabras anteriores dentro de un texto. [31]

El conjunto de datos no requiere ningún paso de procesamiento previo. En otras palabras, se omiten las mayúsculas y minúsculas, la tokenización y otros pasos, ya que los autores creen que estos pasos de preprocesamiento restringen la capacidad del modelo y así ser capaz de evaluar todos los puntos de referencia del lenguaje.

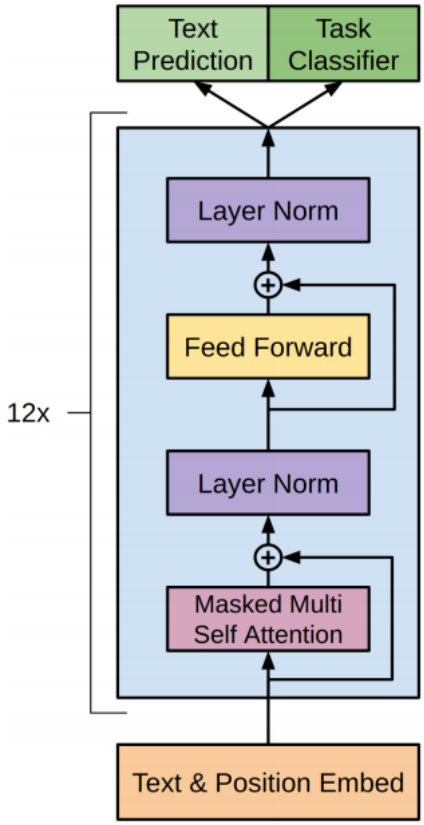
Ya que en GPT-2 no se aplica una representación de las palabras ni a nivel de palabra ni a nivel de carácter. Se elige una opción que se encuentra situada en medio, que es la subpalabra. La subpalabra se puede obtener mediante el algoritmo Byte Pair Encoding (BPE).

BPE es una forma de compresión donde se calculará una lista de subpalabras utilizando el siguiente algoritmo: [32]

- Dividir palabra en secuencia de caracteres.
- Unirse al patrón de frecuencia más alta
- Continuar con el paso anterior hasta alcanzar el número máximo predefinido de subpalabras de iteraciones.

2.6.2. Arquitectura GPT-2

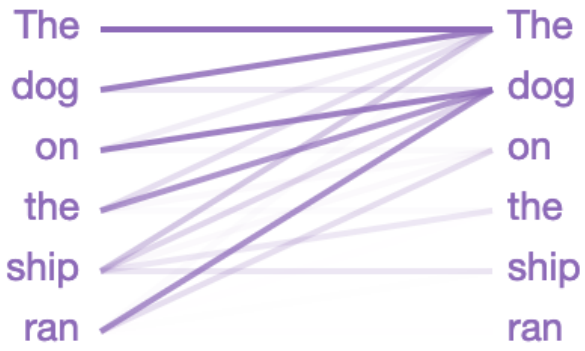
Figura 2.6: Arquitectura del modelo de GPT-2 [33]



El modelo de GPT-2 es un transformer decodificador de 12 capas, cada una con 12 mecanismos de atención independientes, llamados "cabezas"; el resultado es 12 x 12 = 144 patrones de atención distintos. Cada uno de estos patrones de atención corresponde a una propiedad lingüística capturada por el modelo. [34]

Lo que le permite identificar la relación que existe entre las palabras, entendiendo la oración y permitiendo predecir la siguiente palabra en relación a la oración.

Figura 2.7: Ejemplo análisis de una oración por GPT-2 [35]



Las líneas, leídas de izquierda a derecha, muestran dónde presta atención el modelo prediciendo la siguiente palabra en la oración (la intensidad del color representa la fuerza de la atención). Entonces, al tratar de predecir la siguiente palabra después de correr (ran), el modelo presta mucha atención al perro (dog) en este caso. Esto ya que necesita saber quién o qué está corriendo para predecir qué viene a continuación. [35]

2.7. BERT vs GPT-2

Dentro de esta sección mostramos una pequeña comparación entre las tecnologías BERT y GPT-2, visualizada en la siguiente tabla: Con base en lo anterior y de las pruebas realizadas se determino que se va a utilizar BERT el resto del trabajo terminal para ayudarnos en la generación de los textos.

BERT	GPT-2
Es de naturaleza bidireccional.	Es de naturaleza autorregresivo.
El usuario puede entrenar sus propios modelos.	Puede resolver distintos problemas de Procesamiento del Lenguaje Natural (PLN) sin necesidad volver a entrenar la red neuronal.
La separación de palabras es manejada en tokens.	La separación de palabras es realizada por medio del algoritmo Byte Pair Encoding (BPE) generando subpalabras.
Su uso es enfocado al análisis y generación de textos.	Sus principales usos son para la generación de textos.

Cuadro 2.1: Comparación entre las tecnologías BERT y GPT-2

2.8. Base de datos

Una base de datos es un conjunto organizado de datos o información, los cuales pertenecen a un mismo contexto, se encuentran almacenados de forma física o digital con la finalidad de realizar distintas acciones como consultas futuras, ingreso de nuevos datos, actualización o eliminación de estos.

Las bases de datos se componen de una o más tablas divididas en columnas y filas y son las encargadas de guardar un conjunto de datos.

Una base de datos generalmente es manejada por un Sistema de gestión de base de datos (DBMS). En conjunto, el DBMS y los datos o información, unido a las aplicaciones asociadas a ellos, se les conoce como un sistema de base de datos.[36]

2.8.1. Base de datos NoSQL o no relacionales

Estas bases permiten que los datos no estructurados y/o semiestructurados se almacenen y manipulen, a diferencia de la base de datos relacional donde se define como deben de componerse todos los datos insertados en esta. Generalmente los registros de este tipo de base de datos suelen almacenarse como un documento de tipo JSON.

2.9. Canción

Una canción es una composición literaria, generalmente escrita en versos, a la cual se le puede acompañar con música para poder ser cantada.[37]

2.9.1. Elementos de una canción

Los elementos que conforman una canción son los siguientes:

Introducción

Generalmente es una parte única la cual se encuentra al inicio de una canción, acompañada de una armonia o melodia compuesta solo para este inicio. El objetivo de la introducción es de atraer la atención y producir un ambiente.[38]

Verso

Es la parte encargada de comenzar a desarrollar la idea a transmitir, trata de contarnos el tema de la canción, y ya cuenta con una armonia bien establecida.

Pre-estribillo

Es un arreglo que permite realizar una transición, su función principal es conectar el verso con el estribillo. También ayuda a evitar que el estribillo se estanque en la monotonía.

Estribillo

Es una estrofa la cual se repite varias veces dentro de una composición. Su función principal es acentuar la idea principal de la canción tanto en su letra como en lo musical. Se considera como una de las partes más importantes dentro de una canción y en algunas ocasiones este es repetido al inicio y al final.

Puente

Es un interludio el cual conecta dos fragmentos de una canción, permitiendo construir una armonia entre ellas, suele ser usado para llevar a la canción a su clímax, para prepararlo para el desarrollo final de la canción.

Cierre

Este busca terminar o concluir una canción, una de las formas puede ser un ruptura brusca generada por un silencio imprevisto o por una secuencia de acordes. Pero la manera más ordinaria de cerrar es haciendo uso de la repetición de un estribillo.

2.9.2. Estructura de una canción

La estructura mínima de una canción está compuesta de:

- Verso
- Estribillo
- Verso
- Estribillo

La estructura más empleada en una canción es la siguiente:

- Introducción
- Verso
- Pre-estribillo
- Estribillo
- Verso
- Estribillo
- Puente
- Cierre

2.10. Cadenas de Markov

Las cadenas de Markov son un sistema matemático el cual experimenta con las transiciones de un estado a otro de acuerdo con ciertas reglas probabilísticas. La característica que define a una cadena de Markov es que no importa cómo llegó el proceso a su estado actual, y sus posibles estados futuros son fijos. En otras palabras, la probabilidad de pasar a cualquier otro estado depende únicamente del estado actual y del tiempo transcurrido. El espacio de estados o conjunto de todos los posibles estados, pueden ser cualquier cosa: letras, números, puntuaciones de un partido, acciones, etc.

Son procesos estocasticos, con la diferencia de que estos deben ser "sin memoria", es decir, la probabilidad de las acciones futuras no depende ni se ve afectada de los pasos que la condujeron al estado actual. A esto se le denomina una propiedad de Markov.

En la teoría de probabilidad, el ejemplo más inmediato es el de una cadena de Markov homogénea en el tiempo, en la que la probabilidad de que ocurra cualquier transición de estado es independiente del tiempo.

En el lenguaje de probabilidad condicional y variables aleatorias, una cadena de Markov es una secuencia X0, X1, X2, ... de variables aleatorias que satisfacen la regla de independencia condicional. En otras palabras, el conocimiento del estado anterior es todo lo que se necesita para determinar la distribución de probabilidad del estado actual. Esta definición es más amplia que la explorada anteriormente, ya que permite probabilidades de transición no estacionarias y, por lo tanto, cadenas de Markov no homogéneas en el tiempo; es decir, a medida que pasa el tiempo los pasos aumentan y la probabilidad de pasar de un estado a otro puede cambiar. [39]

Las cadenas de Markov pueden ser modeladas mediante máquinas de estados finitos.

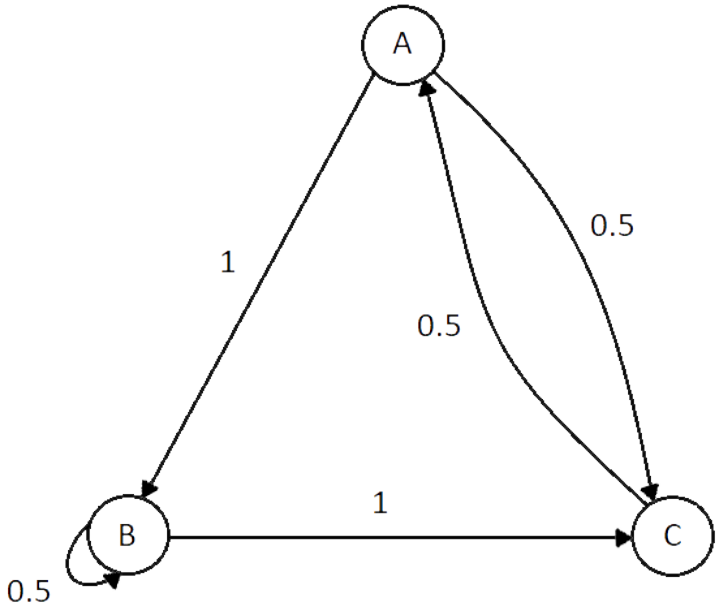


Figura 2.8: Diagrama De Estados finitos De Una Cadena De Markov [39]

2.11. Flask

Flask es un "micro" marco (framework) el cual nos permite crear de manera sencilla aplicaciones web utilizando Python, bajo el patrón de arquitectura Modelo-Vista-Controlador (MVC); cabe mencionar que soporta otros lenguajes como PHP y Java.[40]

No cuenta con un Manejador de Objetos Relacionales u ORM por sus siglas en inglés, pero si cuenta con características como el enrutamiento de URLS y un motor de plantillas. Flask solo nos da las herramientas necesarias para poder crear una aplicación web funcional, pero si se requieren de otras herramientas para añadir alguna funcionalidad, se puede adaptar añadiéndole ciertos plugins.

En general es un marco de aplicación web WSGI. Web Server Gateway Interface (WSGI) es una especificación que describe como se va a comunicar un servidor web con una aplicación web, y como se pueden llegar a enlazar distintas aplicaciones web para procesar una solicitud o una petición.

2.12. Apache

Es un servidor web gratuito y de código abierto el cual permite que los desarrolladores de sitios web puedan desplegar el contenido de ellas, este no es un servidor que se encuentre físicamente, sino que se trata de un software el cual ejecuta un servidor. Su función es instaurar la conexión entre un servidor y los usuarios del sitio web mientras estos intercambian archivos entre ellos (siguiendo una estructura cliente-servidor). El servidor y el cliente interactúan mediante el protocolo HTTP, y el software de Apache es el encargado de mantener una comunicación ágil y segura entre las 2 partes.

Apache trabaja sin problemas con otros sistemas de gestión de contenido (Drupal, Joomla, etc.), marcos de trabajo (Django, Laravel, etc.), y lenguajes de programación. Por estas razones se vuelve una opción sólida al momento de escoger entre los distintos tipos de plataformas de alojamiento web, como serían Virtual Private Server (VPS) o alojamiento compartido.

2.13. NGINX

NGINX es un software de código abierto para servicio web, almacenamiento en caché, equilibrio de carga, proxy inverso, transmisión de medios y más. Inició como un servidor web diseñado para un máximo rendimiento y estabilidad. Además de sus capacidades de servidor HTTP, NGINX también puede funcionar como un servidor proxy para correo electrónico (IMAP, POP3 y SMTP) y un proxy inverso, así como un equilibrador de carga para servidores HTTP, TCP y UDP.[41]

Apache	NGINX
Es fácil de configurar, cuenta con muchos módulos, así como un entorno sencillo.	Usa hilos para manejar solicitudes del usuario.
De código abierto y gratuito, inclusive si se usa de manera comercial.	Nginx es uno de los servidores web que soluciona el problema c10k y seguramente de los más exitoso al hacerlo.
Software estable y confiable.	No crea un nuevo proceso por cada solicitud.
Frecuentemente actualizado incluyendo actualizaciones regulares a los parches de seguridad.	Maneja toda solicitud entrante en un único hilo.
Funciona de forma intuitiva, con sitios web hechos con WordPress.	El modelo basado en eventos de Nginx reparte las solicitudes de los usuarios entre procesos de trabajo de manera eficiente.
Comunidad grande y posibilidad de contactar con soporte disponible de manera sencilla en caso de cualquier problema.	Cuenta con mejor escalabilidad.
Presenta inconvenientes de rendimiento con sitios web los cuales tienen alto tráfico.	Es capaz de manejar un sitio web con demasiado tráfico con un uso de recursos mínimo.
Al contar con muchas preferencias de configuración puede generar vulnerabilidades de seguridad.	

Cuadro 2.2: Tabla comparativa de los distintos servidores web contemplados para nuestro proyecto

En ambos casos, se mantienen en desarrollo, con actualizaciones constantes, así como frecuentemente sacando parches para evitar ataques de DDos.

2.14. Servidor Web

La función es mostrar sitios web en internet. A fin de conseguir este objetivo, actúa como un mediador entre el servidor y el dispositivo del cliente. Obtiene el contenido del servidor en cada petición hecha por el usuario y lo entrega al sitio web.

Los servidores web son capaces de procesar archivos escritos en distintos lenguajes de programación como Java, PHP, Python, y otros.

Podría decirse que un servidor web es la herramienta responsable por la correcta comunicación entre el cliente y el servidor.

2.15. Certificado SSL

Conocido como Secure Socket Layer (SSL) o (Capa de Conexión Segura) es un estándar de seguridad global que fue originalmente creado por Netscape en los 90's. SSL crea una conexión encriptada entre tu servidor web y el navegador web de tu visitante permitiendo que la información privada sea transmitida sin que ocurran problemas como serían espionaje, manipulación de la información, y falsificación de los datos del mensaje. Básicamente, la capa SSL permite que dos partes tengan una "conversación" privada.

Para establecer esta conexión segura, se instala en un servidor web un certificado SSL (también llamado 'certificado digital') que cumple dos funciones:

- Autentificar la identidad del sitio web, garantizando a los visitantes que no están en un sitio falso.
- Cifrar la información transmitida.

Hay varios tipos de certificados SSL según la cantidad de nombres de dominio o subdominios que se tengan, como por ejemplo:

- Único: Asegura un nombre de dominio o subdominio completo (Fully Qualified Domain Name (FQDN) por sus siglas en inglés).
- Comodín: Cubre un nombre de dominio y un número ilimitado de sus subdominios.
- Multidominio: Asegura varios nombres de dominio.

2.16. Plataforma en la Nube de Aprendizaje Automático

El entrenamiento de aprendizaje automático y de modelos de aprendizaje profundo involucra miles de iteraciones. Se necesitn esta gran cantidad de iteraciones para producir el modelo más preciso.

El cómputo en la nube permite modelar capacidad de almacenamiento y manejar cargas a escala, o escalar el procesamiento a través de los nodos. Por ejemplo, AWS ofrece instancias de GPUs con capacidad de memoria que va de los 8Gb's a los 256Gb's, estas instancias son cobradas a ritmos por hora.

Las GPUs son procesadores especializados diseñados para procesado complejo de imágenes. Azure de Microsift ofrece GPUs de alto rendimiento de la serie NC para aplicaciones o algoritmos de cómputo de alto rendimiento.

2.16.1. Amazon Web Services

Dentro de las bondades que ofrecen los Servicios Web de Amazon (Amazon Web Services (AWS) por sus siglas en inglés) y que nos pueden ser de utilidad para las nacesidades de nuestro proyecto se pueden encontrar:

- **SageMaker:** Es una plataforma de aprendizaje automático completamente administrado para científicos de datos y desarrolladores. La plataforma corre en Cómputo Elástico en la Nube (Elastic Compute Cloud (EC2) por sus siglas en inglés), y permite construir modelos de aprendizaje automático, organizar la información y escalar sus operaciones.
  Algunas aplicaciones de aprendizaje automático en SageMaker van desde reconocimiento de voz hasta visión de computadora, e incluso recomendaciones basadas en el comportamiento aprendido del usuario.

  El mercado de AWS ofrece modelos que se pueden usar en lugar de empezar desde cero, posterior a eso se puede entonces empezar a entrenar y optimizar el modelo; las elecciones más comunes son frameworks como Keras, TensorFlow, y PyTorch; Sagemaker puede optimizar y configurar estos frameworks automáticamente, o pueden ser entrenadas de manera personal.

Uno mismo puede incluso desarrollar su propio algoritmo construyéndolo en un contenedor de Docker o se puede hacer uso de una "Jupyter notebook" para construir un modelo propio de aprendizaje automático y visualizar su información usada para el entrenamiento del modelo siendo este punto lo que más nos interesa para nuestro proyecto.

Cuadro 2.3: Tabla comparativa de las diversas plataformas contempladas

Característica	GCP [74]	AWS [75]	Azure [76]
Jupyter Notebook alojado de manera local o remota	Plataforma de IA	SageMaker Studio IDE	- Estudio de Notebooks de Azure de Aprendizaje Automático - Databrick de Azure
Entrenamiento Distribuido	Si	Si	Si
Versionado de Modelos	Si	Si	Si
Seguimiento de Experimentos	Si	Si	Si
AutoML (UI y API)	Tabla de AutoML	Autopiloto de SageMaker	AutoML
Análisis de Errores	Tabla de AutoML con BigQuery	Debugger de SageMaker	Aprendizaje Profundo de Azure

Los tres proveedores de la tabla anterior han alojado servicios de "Jupyter Notebook" (contienen tanto código de computadora como elementos ricos en texto como ecuaciones, figuras, etc.), experimentando seguimientos y control de versiones, y métodos de despliegue sencillos.

Dentro de las características únicas de cada una de las plataformas podemos encontrar que la Plataforma de la Nube de Google (GCP) usa un paquete llamado "what if tool" el cual se puede integrar junto a un "Jupyter Notebook" y de esa manera jugar con el modelo cambiando el umbral o un valor característico de un ejemplo dado, esto permite checar como es que ciertos cambios afectan el resultado predicho con anterioridad previo al cambio.

El debbuger de SageMaker de AWS permite analizar cómo es que la ingeniería de características y el refinado del modelo son hechos, o de forma más concisa, permite ver qué sucede durante el entrenamiento del modelo.

Azure, por su parte, provee un módulo propio en su SDK el cual parece tener la mejor integración de entre las tres plataformas.

Costo y Rendimiento del Modelo

Azure y GCP puntúan ligeramente mejor que AWS en términos de rendimiento, esto no necesariamente significa que una plataforma es mejor que otra.

El costo de AWS fue considerablemente menor que GCP y Azure. Por supuesto, cada una tiene sus fuertes como puede verse en la tabla presentada anteriormente.

El diseñador de aprendizaje automático de Azure cuenta con una interfaz de arrastar y soltar el cual es bastante amigable con un usuario novato, esto es, porque requiere menos codeo y antecedentes técnicos. AWS y GCP parecen ser más enfocados a desarrolladores aunque puede resultar en más de trabajo ensamblar una cadena de procesos (pipeline), estos resultan más personalizables con los diferentes componentes disponibles. Estos componentes y la conexión de la cadena de procesos son usualmente desarrollados usando código y configuraciones, en vez de utilizar una interfaz.

Tanto GCP como AWS ofrecen un modelo de pago "pay-as-you-go". Este modelo es el mejor para aquellos individuos que puedan llegar a esperar un uso intermitente de la nube, ya que permite un enfoque flexible para añadir y

remover servicios cuando se necesite. Por supuesto, este nivel de flexibilidad tiene un costo, haciendo al modelo "pay-as-you-go" el más caro por hora.

Cuadro 2.4: Tabla comparativa GCP vs AWS (en dólares)

Tipo de Instancia	Precio de EC2 (por hora)	Precio de Google (por hora)
Propósito General	$0.134	$0.15
Cómputo Optimizado	$0.136	$0.188
Optimizado de Memoria	$0.201	$0.295
GPU	$0.526	$1.4

En conclusión, la opción ideal para las necesidades de nuestro proyecto es AWS debido a que es más económico que las otras dos plataformas y cuenta con herramientas más útiles como SageMaker que nos permitirá saber cómo se está entrenando el modelo.

Capítulo 3

Análisis

3.1. Estudio de Factibilidad

El estudio de factibilidad es un instrumento que sirve para orientar la toma de decisiones, así como para determinar la posibilidad de desarrollar un negocio o un proyecto; corresponde a la última fase de la etapa pre-operativa del ciclo del proyecto. Se formula con base en información que tiene la menor incertidumbre posible para medir las posibilidades de éxito o fracaso de un proyecto, apoyándose en el resultado se tomará la decisión de proceder o no con su implementación.

Este estudio establecerá la viabilidad, si existe, del trabajo planteado previamente.

- **Factibilidad Técnica:** Este aspecto evalúa que la infraestructura, es decir, los equipos, el software, el conocimiento, la experiencia, etc. que se poseen son los necesarios para efectuar las actividades requeridas para la realización del trabajo terminal.
- **Factibilidad Operativa:** Analiza si el personal posee las competencias necesarias para el desarrollo del proyecto.
- **Factibilidad Económica:** Consiste en el análisis de los recursos financieros necesarios para llevar a cabo la elaboración de este proyecto.

3.1.1. Factibilidad Técnica

Dentro de este apartado se explican detalladamente las tecnologías que se utilizarán, así como las características de nuestros equipos de cómputo actualmente. La elección de estas herramientas estuvo basada tanto en las

tecnologías que más se utilizan en la actualidad como las que cuentan con el mayor soporte para su trabajo en la nube, esto se debe a que los equipos con los que contamos actualmente no soportarían todo el trabajo.

Equipo de cómputo ideal.	
Procesador	Intel i9 10900KF
Tarjeta de video	RTX 3080 10Gb
Memoria RAM	32Gb DDR4
Disco duro	2Tb HDD y 512Gb SSD

Cuadro 3.1: Equipo de cómputo ideal

En la tabla anterior se muestra el equipo que requeriríamos para poder trabajar sin mayor dificultad al momento de realizar los entrenamientos de los modelos de redes neuronales que necesitamos para la realización de este trabajo, debido a la situación actual de la pandemia, en el mercado actual los componentes para el equipo son escasos y su precio en el mercado aumento. Por esa razón se utilizarán servicios de la nube para la realización de estos entrenamientos y nuestros equipos para el desarrollo del resto del proyecto.

Herramientas de Software a utilizar	
Sistema Operativo	Linux, Mac, Windows
Navegador Web	Google Chrome
Lenguaje de Programación	Python
Servidor	Apache y Gunicorn
Servicio Nube	Amazon Web Services

Cuadro 3.2: Herramientas de Software a utilizar

Además de las herramientas de software a utilizar, es necesario mencionar el equipo de hardware que se utiliza, tanto para desarrollar, como para probar e implementar cada uno de los prototipos a lo largo de este trabajo terminal.

Equipo de cómputo utilizado. [1]	
Procesador	Ryzen 5 3600
Tarjeta de video	Amd Radeon Rx580
Memoria RAM	32 Gb
Disco duro	1Tb HDD y 512Gb SSD

Cuadro 3.3: Equipo de cómputo 1

Equipo de cómputo utilizado. [2]	
Marca	Apple
Modelo	iMac Late 2012
Procesador	Intel Core i5
Tarjeta de video	NVIDIA GeForce GT 640M 512 Mb
Memoria RAM	8 Gb
Disco duro	1Tb y 256 Gb SSD

Cuadro 3.4: Equipo de cómputo 2

Equipo de cómputo utilizado. [3]	
Procesador	Amd FX-8350
Tarjeta de video	Nvidia Geforce 1050ti
Memoria RAM	16 Gb
Disco duro	1Tb HDD

Cuadro 3.5: Equipo de cómputo 3

Junto con las herramientas de hardware y software a utilizar es necesario mencionar los servicios básicos que son relevantes para el desarrollo de este trabajo terminal como lo son:

- Luz Eléctrica
- Agua Potable
- Internet

Estos servicios forman parte de la factibilidad técnica ya que sin ellos no se podría realizar este proyecto y por eso mismo generan un costo, dicho costo se menciona en la Factibilidad Económica.

3.1.2. Factibilidad Operativa

A continuación se presenta una tabla con los recursos operativos del trabajo terminal que se calcularon con base en los recursos humanos con los que se cuenta actualmente y un análisis de las horas en las que el personal estará en operación:

Horas a trabajar en el desarrollo del trabajo terminal					
Mes	No. de Días	Sábado y Domingo	Días hábiles	Horas de trabajo por día	Horas Totales
Marzo	31	8	22	2	44
Abril	30	8	15	2	30
Mayo	31	10	19	2	38
Junio	30	10	17	2	34
Agosto	31	10	18	2	24
Septiembre	30	8	20	2	40
Octubre	31	10	20	2	40
Noviembre	31	8	18	2	36

Cuadro 3.6: Relación de horas de trabajo estimadas para la realización de este trabajo terminal

Con esto podemos concluir que contamos con 286 horas, suficiente tiempo para el desarrollo de este trabajo terminal, ya que las horas totales de trabajo están contempladas para cada uno de los integrantes del equipo

3.1.3. Factibilidad Económica

Luego de haber realizado el estudio de factibilidad técnica así como el operacional es necesario tomar en cuenta un estudio de factibilidad económica el cual desglosará todo el gasto económico realizado para la elaboración de este trabajo terminal:

- **Capital Humano:** Se tienen contemplados aproximadamente 36 días laborales, es decir, 288 horas para la elaboración de este trabajo terminal en el cual participaremos los tres integrantes
- **Capital Técnico:** Se cuenta con las viviendas y el equipo de cómputo principal de cada uno de los integrantes.

Respecto a los costos monetarios de todo el proyecto se toma a consideración lo siguiente:

- **Servicios**
  Se considera un gasto mensual aproximado de $1,600.00 que al ser multiplicado por todo el tiempo de elaboración contemplado nos da un total de $14,400.00.
- **Software**
  Durante algunos periodos se va a hacer uso principalmente de herramientas gratuitas y de software libre, en cuanto al servicio en la nube se pretende hacer uso de AWS y trabajar inicialmente con los planes gratuitos que ofrece, en caso de que se lleguen a consumir los recursos de este plan, entonces se procederá a cambiarnos a otro plan superior donde el costo actual se cobra en $ 0.15 centavos de dólar por hora de uso.
- **Hardware**
  Se utilizarán los equipos de cómputo personal de cada integrante, lo que da un costo aproximado de $24,012.00 aplicando los parámetros de vida útil de un equipo de cómputo y su depreciación anual del hardware.
- **Recursos Humanos**
  Se estima un gasto aproximado de $15,000.00 por cada integrante del equipo para la elaboración del proyecto con lo que se generará un gasto total de $45,000.00

Tomando en cuenta el salario promedio de un desarrollador de software el cual es de $15,000.00 mensuales (este dato fue obtenido de mx.indeed.com), se consideró este salario para el desarrollo del proyecto, el cual tendría una

duración de desarrollo de 6 meses aproximadamente, dando como resultado un salario total para los integrantes del equipo de $270,000.00; el costo final del desarrollo de este trabajo terminal sería de:

$353,420.00

Conclusión
Tras realizar el análisis el estudio de factibilidad de este proyecto es pertinente decir que los integrantes no cuentan con el apoyo financiero y que el hardware mencionado ya es propiedad de los integrantes, por lo que el trabajo terminal se califica como *"Viable"* iniciando de esta manera su implementación acorde con las fechas mencionadas.

3.2. Análisis de riesgo

Dentro de este apartado determinaremos la probabilidad de que surja alguna problemática o riesgo el cual pueda impactar el desarrollo del proyecto, ya sea en términos del cronograma, la calidad o los costos.

Algunos riesgos que enfrentaremos durante el desarrollo del proyecto son los siguientes:

- **Riesgo de costo**
  Si sobrepasamos el costo de desarrollo previsto, quizá nos enfrentemos a problemas relacionados con el alcance y los requerimientos funcionales y no funcionales del proyecto.

Cuadro 3.7: Riesgo de costo

Riesgo: Costos Elevados	
Impacto: Medio alto (7).	Nivel: Alto (49)
Probabilidad: Medio alto (7)	

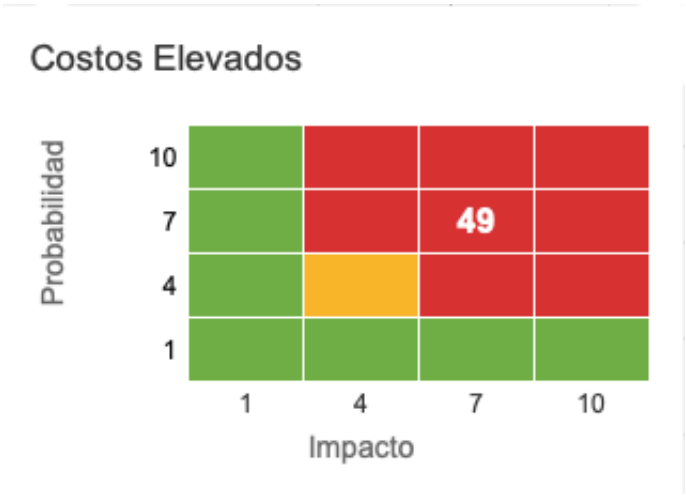


Figura 3.1: Matriz de Riesgo de Costo

- **Riesgo de calendario**
  Si se realizó una mala estimación del tiempo necesario para el desarrollo, si se asignaron mal los recursos, si hubo alguna afectación dentro del recurso humano, lo más probable es que el proyecto no se termine de desarrollar en el tiempo establecido.

Cuadro 3.8: Riesgo de calendario

Riesgo: Falta de Tiempo	
Impacto: Medio bajo (4).	Nivel: Medio Medio (16)
Probabilidad: Medio bajo (4)	

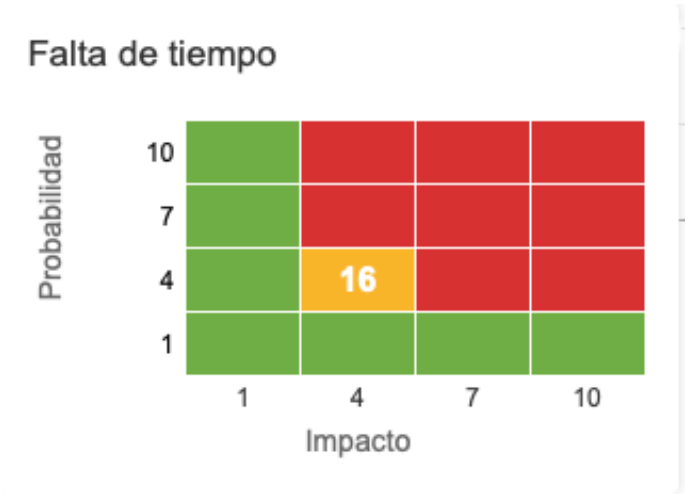


Figura 3.2: Matriz de Riesgo de calendario

- **Riesgo tecnológico**
  Si no logramos entender la complejidad de la herramientas que utilizaremos para el desarrollo del proyecto o nos toma demasiado tiempo que son nuevas para nosotros, como lo son la herramientas de AWS para la creación de los modelos de redes neuronales y su entrenamiento dentro de la misma plataforma, se podrían presentar problemas como: su integración a la página web, creación fallida de los modelos de redes neuronales, posible no adaptación con el hardware con el que contamos, entre otros.

Cuadro 3.9: Riesgo tecnológico

Riesgo: Fallas de Equipos	
Impacto: Bajo (1).	Nivel: Bajo (7)
Probabilidad: Medio alto (7)	

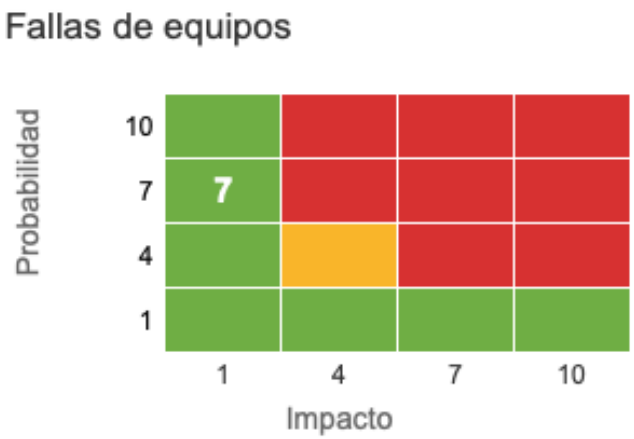


Figura 3.3: Matriz de Riesgo tecnológico

- **Riesgo operacional**
  Si no se plantean posibles soluciones a posibles problemas que se pueden presentar durante el proyecto, si no se presenta liderazgo por parte de alguno de los miembros, si existe una mala comunicación, falta de motivación y si no hay una monitorización, se puede ver mermado el avance del proyecto.

Cuadro 3.10: Riesgo operacional

Riesgo: Desorganización en el equipo	
Impacto: Medio bajo (4).	Nivel: Medio (16)
Probabilidad: Medio bajo (4)	

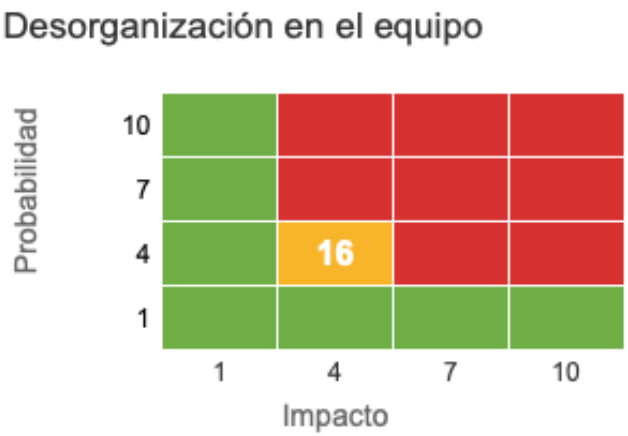


Figura 3.4: Matriz de Riesgo operacional

- **Riesgos externos**
  Factores externos que pueden afectar el desarrollo del proyecto pueden ser algún cambio en las leyes, cambio en alguna norma, desastre natural.

Cuadro 3.11: Riesgos externos

Riesgo: Factores Externos	
Impacto: Alto (10).	Nivel: Bajo (10)
Probabilidad: Bajo (1)	

Figura 3.5: Matriz de Riesgos externos

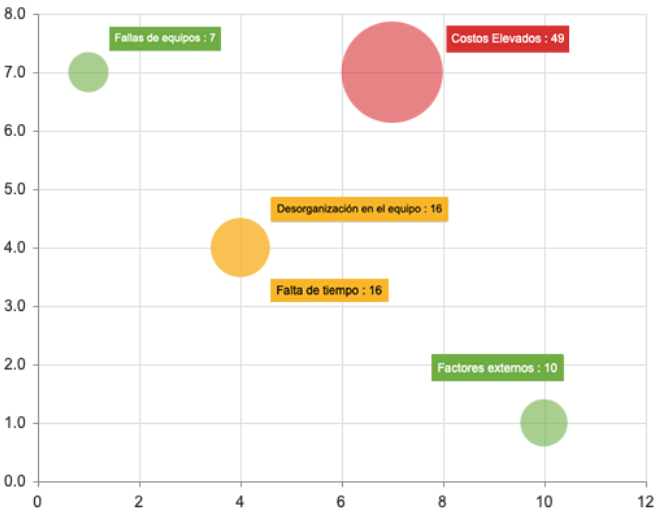


Figura 3.6: Matriz General de Riesgos

Se utilizó la herramienta llamada Matriz de Evaluación de Riesgos de ITM Platform para generar este análisis de riesgo. [77]

3.3. Herramientas a usar

3.3.1. Software

Para el desarrollo de software de este prototipo, es necesario hacer mención de algunas de las siguientes herramientas, para tener una idea clara sobre qué herramientas estamos utilizando y porque es que las estamos utilizando:

HTML

Hypertext Markup Language (HTML) es un lenguaje de marcado que define la estructura de una página web y su contenido. HTML consta de una serie de elementos que se utilizan para encerrar o envolver diferentes partes del contenido para que estos se visualicen de cierta manera o actúe de cierta manera. Las etiquetas adjuntas pueden hacer que una palabra o imagen sea un hipervínculo a otro lugar, pueden poner palabras en cursiva, pueden hacer que la fuente sea más grande o pequeña, etc. [42]

HTML5 es la versión mas reciente de html, la cual integra nuevos elementos, atributos y comportamientos. Permite describir de mejor manera el contenido de la página web, así como mejora su conectividad con el servidor y almacenamiento, posibilita que las paginas web puedan operar sin conexión usando los datos almacenados localmente del lado del cliente, otorga un mejor soporte al contenido multimedia así como una mejor integración a APIs y un mejor diseño usando CSS3. [43]

CSS

Cascading Style Sheet (CSS) es el lenguaje para describir la presentación de las páginas web así como hacerlas más atractivas. Permite adaptar la presentación a diferentes tipos de dispositivos. CSS es independiente de HTML y puede ser empleado con cualquier otro lenguaje de marcado basado en XML o SVG. Usando CSS se pueden controlar con precisión cómo se ven los elementos HTML en el navegador, que presentará para las etiquetas de marcado el diseño que cada uno desee. La separación de HTML de CSS facilita el mantenimiento de los sitios, compartir las hojas de estilo entre páginas y adaptarlas a distintos ámbitos. [44]

Es un lenguaje basado en reglas: cada usuario define las reglas que especifican los grupos de estilos que van a aplicarse a elementos particulares o grupos de elementos de la página web.

Antes de CSS, las etiquetas como fuente, color, estilo de fondo, alineación,

borde y tamaño tenían que repetirse en cada elemento de una página web. Ahora con los CSS, podemos definir cómo se van a comportar las etiquetas, al ser guardado en un archivo por separado, esta misma configuración puede usarse en otra página web ahorrando tiempo diseñándola. Además de que CSS provee de mejor y más detallados atributos para cada etiqueta.

JavaScript

JavaScript es un lenguaje de programación o secuencias de comandos que permite implementar funciones complejas en las páginas web. Estos scripts pueden ser desarrollados en el mismo HTML para que sean ejecutados automáticamente cuando se carga dicha páginas web, estos scripts se proporcionan y ejecutan como texto sin formato. No necesitan una preparación especial ni una compilación para ejecutarse. [45]

JavaScript puede ejecutarse no solo en un navegador, sino también en un servidor, o en cualquier dispositivo que tenga un programa especial llamado JavaScript engine, el cual permite interpretar y ejecutar los scripts.

JavaScript permite crear contenido dinámico dentro de las páginas web, reaccionar ante algunas acciones realizadas por los usuarios como lo son los clicks del ratón, el movimiento del puntero o el presionar cierta tecla, permite enviar peticiones al servidor, así como descargar y subir archivos, además es posible obtener y configurar cookies, mostrar mensajes o alertas al usuario.

OpenSSL

Consiste en un paquete robusto de herramientas de administración y bibliotecas las cuales están relacionadas con la criptografía, estas suministran funciones criptográficas a otros paquetes como OpenSSH y navegadores web (permitiendo un acceso seguro a sitios HTTPS). Estas herramientas contribuyen al sistema a instaurar el protocolo Secure Socket Layer (SSL), de igual manera como otros protocolos relacionados con la seguridad, como el Transport Layer Security (TLS). OpenSSL posibilita la creación certificados digitales los cuales pueden aplicarse a un servidor, por ejemplo a Apache [46].

MongoDB

MongoDB es un sistema de base de datos multiplataforma dirigido a documentos, de esquema libre, es decir, que cada registro o entrada es capaz de contar con un esquema de datos distinto, con columnas o atributos que pueden variar o no de un registro a otro.

Sus características más destacadas son su sencillo sistema de consulta de la base de datos y la velocidad. Alcanzando así un balance perfecto entre rendimiento y funcionalidad. MongoDB utiliza un modelo NoSQL el cual es un modelo de agregación que se basan en la noción de agregado, entendiendo el agregado como una colección de objetos relacionados que se desean tratar de forma semántica e independiente [62].

Las ventajas que ofrece MongoDB como herramienta de desarrollo de base de datos no relacionales son:

- La base de datos no tiene un esquema de datos predefinido.
- El esquema puede variar para instancias de datos que pertenecen a una misma entidad.
- En ocasiones el gestor de la base de datos no es consciente del esquema de la base de datos.
- Permite reducir los problemas de concordancia entre estructuras de datos usadas por las aplicaciones y la base de datos.
- Frecuentemente se aplican técnicas de desnormalización de los datos.

MySQL

Es un sistema de gestión de bases de datos desarrollado por Oracle Corporation, es considerada como la base de datos de código abierto más popular del mundo y una de las más populares, sobre todo para entornos de desarrollo web [66].

El modelo más común utilizado en este sistema de gestión de base datos, es el modelo relacional, donde se almacenan y proporcionan acceso a puntos de datos relacionados entre sí. Esto es posible a que cada fila de la tabla cuenta con un registro y este tiene un ID único, las columnas contienen atributos de los datos, y cada registro en su mayoría cuenta con un valor para cada atributo, lo que favorece el establecimiento de las relaciones entre los datos.

Las principales ventajas son:

- Evita la duplicidad de registros, habilitando ciertas configuraciones.
- Asegura una integridad referencial, es decir, al eliminar un registro elimina todos los registros relacionados.

- Favorece la normalización al ser más comprensible y aplicable.

Mientras que las principales desventajas son:

- Presentan deficiencias con datos gráficos, multimedia, CAD y sistemas de información geográfica.
- Dificulta la manipulación de bloques de texto como un tipo de dato.

Flask

Flask es un mini marco (framework) web, esto es, un modulo de Python el cual permite desarrollar aplicaciones web. No cuenta con un Manejador de Objetos Relacionales u ORM por sus siglas en inglés, pero si cuenta con características como el enrutamiento de URLS y un motor de plantillas. En general es un marco de aplicación web WSGI.

La Web Server Gateway Interface (WSGI) es una especificación que describe como se va a comunicar un servidor web con una aplicación web, y como se pueden llegar a enlazar distintas aplicaciones web para procesar una solicitud o una petición.

Las principales ventajas de Flask son:

- Permite escalar con facilidad la aplicación.
- Es de fácil desarrollo.
- Es flexible con la platillas que te da por default.
- Es modular.

Principales desventajas son:

- No cuenta con muchas herramientas o librerias de apoyo.
- No puede manejar peticiones multiples al mismo tiempo.

Gunicorn

Gunicorn, también conocido como unicornio verde “Green Unicorn”, es una de las muchas implementaciones de un Web Server Gateway Interface (WSGI) y se usa comúnmente para ejecutar aplicaciones web hechas con Python. Esta implemente la especificacion WSGI de frameworks como Django, Flask o Bottle.

Python

Python es un lenguaje de programación orientado a objetos y de alto nivel con semántica dinámica. Sus estructuras de datos integradas de alto nivel, combinadas con tipado dinámico y enlace dinámico, lo hacen muy atractivo para el desarrollo rápido de aplicaciones, así como para su uso en scripts o para conectar componentes ya existentes. La sintaxis simple y fácil de aprender de Python enfatiza la legibilidad y, por lo tanto, reduce el costo de mantenimiento del programa. Python admite módulos y paquetes, lo que fomenta la modularidad del programa y la reutilización del código.[61]

Las principales ventajas de Python son:

- Es fácil y rápido desarrollar una aplicación.
- Cuenta con una gran cantidad de librerias.
- Es fácil de entender el código y darle mantenimiento.

Principales desventajas son:

- En ocasiones la ejecución del código es lenta.
- Consume mucha memoria.

BERT

Bidirectional Encoder Representations from Transformers (BERT) es un marco de aprendizaje automático de código abierto para el Procesamiento del Lenguaje Natural (PLN). BERT, significa Representaciones de codificador bidireccional de transformers, basado en transformers, el cual es un modelo de aprendizaje profundo en el que cada elemento de salida está conectado a cada elemento de entrada y las ponderaciones entre ellos se calculan dinámicamente en función de su conexión. [28]

Al contar con la capacidad bidireccional, BERT está previamente entrenado en dos tareas de PLN diferentes, pero relacionadas: el modelado de lenguaje enmascarado y la predicción de la siguiente oración.

El objetivo del entrenamiento del Modelado de Lenguaje Enmascarado (MLM) es ocultar una palabra en una oración y luego hacer que el programa prediga qué palabra se ha ocultado en función del contexto de la palabra oculta. El objetivo del entrenamiento de predicción de la siguiente oración es que el programa prediga si dos oraciones dadas tienen una conexión lógica secuencial o si su relación es simplemente aleatoria.

GPT-2.

Generative Pretrained Transformer (GPT) es un transformer aprovechado para realizar tanto aprendizaje supervisado como aprendizaje no supervisado para el Procesamiento del Lenguaje Natural (PLN).

GPT-2 es el sucesor de GPT, es un gran modelo de lenguaje basado en transformers con 1500 millones de parámetros, entrenado en un conjunto de datos de 8 millones de páginas web. GPT se entrena con un objetivo simple: predecir la siguiente palabra, dadas todas las palabras anteriores dentro de un texto. [31]

Amazon EC2.

Amazon Elastic Compute Cloud (EC2) proporciona una infraestructura de tecnologías de información que se ejecuta en la nube y funciona como un centro de datos que se ejecuta en su propia sede. Es ideal para empresas que necesitan rendimiento, flexibilidad y potencia al mismo tiempo.

Amazon EC2 es un servicio que permite alquilar un servidor o máquina virtual de forma remota para ejecutar aplicaciones. Las principales ventajas de Amazon EC2 son:

- Cuenta con plantillas predeterminadas de maquinas virtuales y servidores.
- Permite configurar la memoria, almacenamiento, CPU y otras caracteristicas dependiendo las necesidades del usuario.
- Precios dependiendo del uso y características del equipo o servidor alquilado.

Las principales desventajas son:

- Tiene una curva de aprendizaje alta para los nuevos usuarios.
- Los costos de soporte técnico son muy elevados..

Amazon CLI

Amazon Command Line Interface (CLI) es una herramienta que reúne todos los servicios de AWS en una consola central, lo que le brinda un control sencillo de varios servicios de AWS con una sola herramienta. Como su nombre indica, los usuarios operan los distintos servicios desde la línea de comandos.

Amazon SageMaker

Amazon SageMaker es un servicio que ayuda a científicos y desarrolladores a construir, entrenar e implementar de manera rápida y sencilla modelos de machine learning.

Para construir el modelo, este servicio cuenta con algoritmos de machine learning más utilizados que vienen preinstalados. También está preconfigurado para que pueda ejecutar Apache MXNet y TensorFlow.

Para el entrenamiento, con un solo clic en la consola de servicio, es fácil comenzar a entrenar su modelo. Amazon SageMaker se encarga de cada infraestructura y facilita la escalbilidad lo que permite entrenar los modelos a escala de petabytes. Si se desea acelerar y simplificar el proceso de entrenamiento, se puede ajustar automáticamente el modelo para obtener la mejor precisión.

Para desplegar el modelo entrenado, el modelo se aloja en un clúster de escalado automático de Amazon EC2.

Amazon S3.

Amazon Simple Storage Service (S3), como su nombre lo indica, es un servicio web proporcionado por Amazon Web Services (AWS) que proporciona almacenamiento. Este almacenamiento es altamente escalable en la nube.

Apache

Es un servidor web gratuito y de código abierto el cual permite que los desarrolladores de sitios web puedan desplegar el contenido de ellas, este no es un servidor que se encuentre físicamente, sino que se trata de un software el cual ejecuta un servidor. Su función es instaurar la conexión entre un servidor y los usuarios del sitio web mientras estos intercambian archivos entre ellos (siguiendo una estructura cliente-servidor). El servidor y el cliente interactúan mediante el protocolo HTTP, y el software de Apache es el encargado de mantener una comunicación ágil y segura entre las 2 partes.

Apache trabaja sin problemas con otros sistemas de gestión de contenido (Drupal, Joomla, etc.), marcos de trabajo (Django, Laravel, etc.), y lenguajes de programación. Por estas razones se vuelve una opción sólida al momento de escoger entre los distintos tipos de plataformas de alojamiento web, como serían Virtual Private Server (VPS) o alojamiento compartido.

3.3.2. Hardware

Se usarán los equipos de cómputo con los que los integrantes del equipo contamos actualmente, los cuales se especifican a continuación:

Equipo de cómputo utilizado. [1]	
Procesador	Ryzen 5 3600
Tarjeta de video	Amd Radeon Rx580
Memoria RAM	32 Gb
Disco duro	1Tb HDD y 512Gb SSD

Cuadro 3.12: Equipo de cómputo 1

Equipo de cómputo utilizado. [2]	
Marca	Apple
Modelo	iMac Late 2012
Procesador	Intel Core i5
Tarjeta de video	NVIDIA GeForce GT 640M 512 Mb
Memoria RAM	8 Gb
Disco duro	1Tb y 256 Gb SSD

Cuadro 3.13: Equipo de cómputo 2

Equipo de cómputo utilizado. [3]	
Procesador	Amd FX-8350
Tarjeta de video	Nvidia Geforce 1050ti
Memoria RAM	16 Gb
Disco duro	1Tb HDD

Cuadro 3.14: Equipo de cómputo 3

Capítulo 4

Diseño

4.1. Arquitectura del sistema

4.1.1. Descripción de la arquitectura del sistema

El sistema se compone de tres bloques los cuales se comunicarán vía red:

1. **Base de datos:** En este módulo se almacenan los datos de canciones, se trabajará especialmente con una tabla que contenga como principales campos:

   - Artista
   - Género
   - Nombre de la Canción
   - Letra
   - Idioma

   Dicha base de datos se almacenará en un sistema especializado que se conecte a la nube para poder trabajar y almacenar estos datos sin mayor problema.

2. **Modelado de la red neuronal:** Este bloque va ser el encargado de generar las letras musicales para cada usuario que ingrese . Para la generación de dichas letras musicales, se utiliza un servicio en la nube para procesar la solicitud del usuario.

3. **Página Web**: Este primer bloque es el que se encuentra directamente enlazado con el usuario de nuestro sistema y al mismo tiempo con el servicio en la nube que aloja a las redes neuronales, el usuario utilizando su navegador podrá realizar peticiones al servicio en la web.

4.1.2. Funcionamiento General del Sistema

A continuación se presentan unos diagramas que muestran el funcionamiento general de nuestro proyecto:

Figura 4.1: Diagrama de obtención del dataset

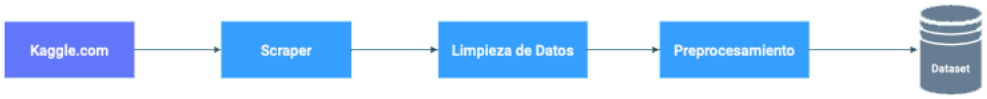
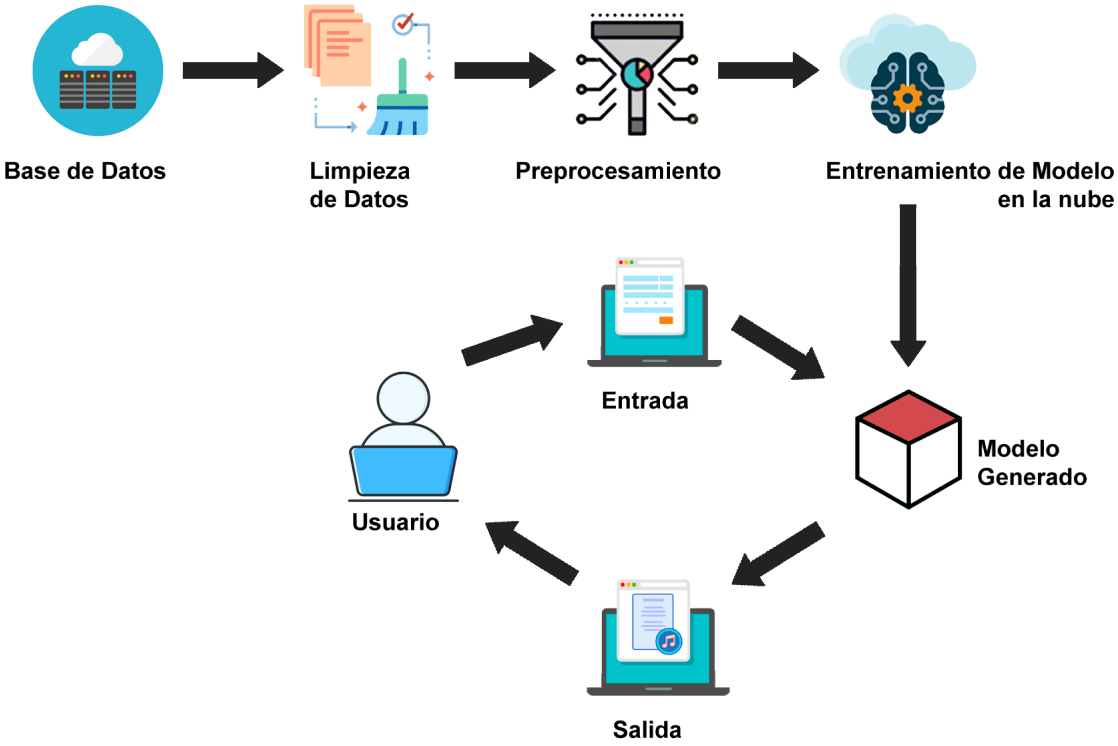


Figura 4.2: Diagrama del entrenamiento y generación del modelo

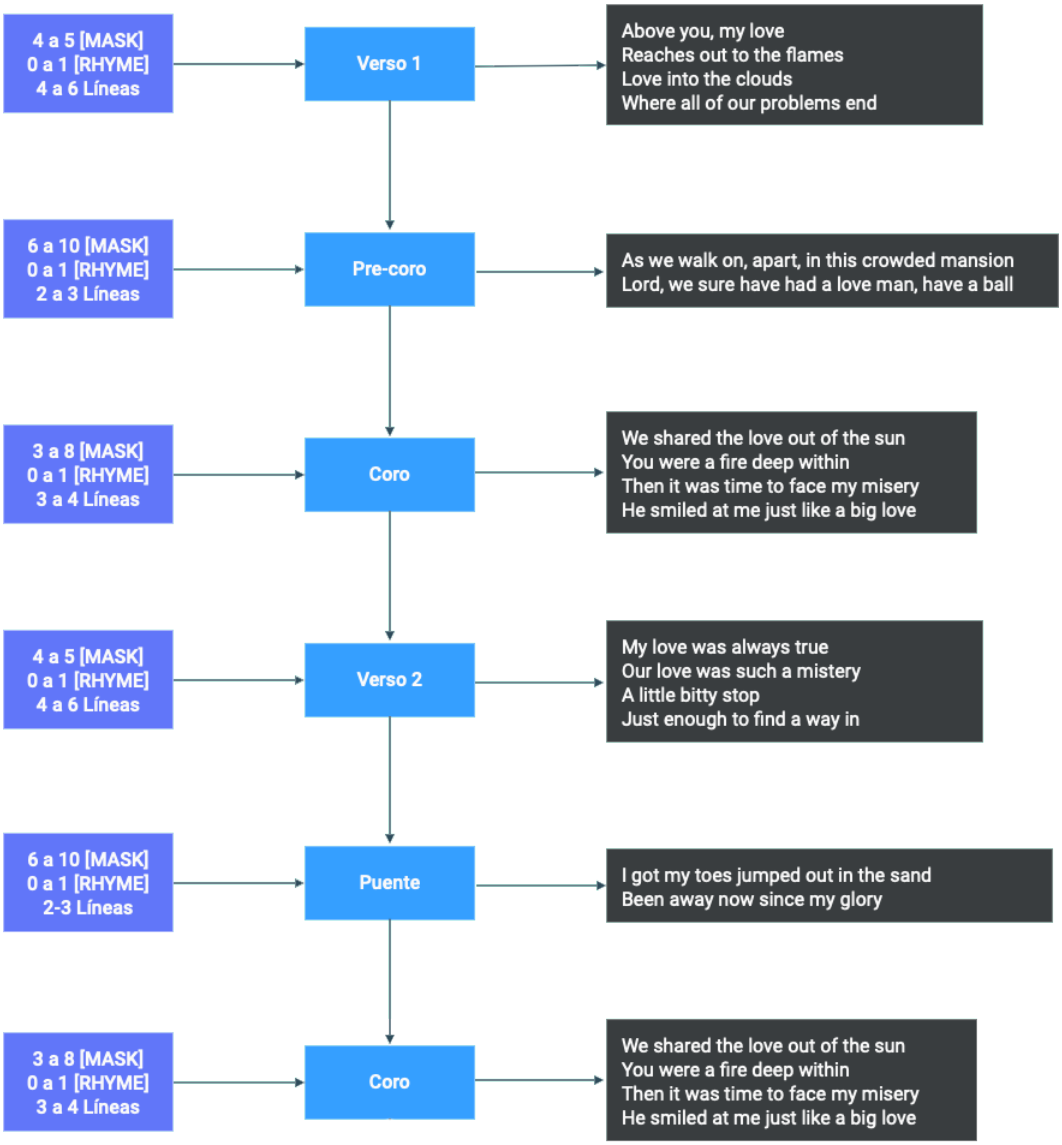
El siguiente diagrama muestra como se espera que quede el funcionamiento general del proyecto, una vez este implementado el modelo entrenado en la página web:

Figura 4.3: Diagrama del funcionamiento general



Se pretende trabajar la estructura de la generación de letras musicales de la siguiente forma:

Figura 4.4: Diagrama de la estructura para la generación de las letras musicales



4.1.3. Historias de Usuario

Cuadro 4.1: Historia de Usuario 1

Historia de Usuario 1	
Nombre: Generar una letra de canción.	Usuario: Visitante
Descripción: Como un visitante de la página web quiero generar una letra de una canción a partir de una palabra o palabras que proporcione.	Validación: Con la palabra o palabras proporcionadas por el usuario con ayuda del modelo entrenado se generará una letra musical relacionada a estas.

Cuadro 4.2: Historia de Usuario 2

Historia de Usuario 2	
Nombre: Descargar una letra canción.	Usuario: Visitante
Descripción: Como un visitante de la página web quiero poder descargar la letra de la canción que se generó a partir de la palabra o palabras que proporcioné.	Validación: El usuario podrá descargar un archivo de texto que contendrá la letra de la canción generada.

Cuadro 4.3: Historia de Usuario 3

Historia de Usuario 3	
Nombre: Definir que tanto deseo que rime una canción	Usuario: Visitante
Descripción: Me gustaría tener una opción en la cual pueda definir que tanto quiero que rime una canción, desde que no rime nada hasta que rime toda.	Validación: Se genera una letra de canción con lo parámetros de rima que establece el usuario.

Cuadro 4.4: Historia de Usuario 4

Historia de Usuario 4	
Nombre: Generar otra letra de canción con los mismos parámetros	Usuario: Visitante
Descripción: Me gustaría generar otra letra de canción con los mismo parámetros y palabra(s) que ya había introducido.	Validación: Brindar una opción al usuario que una vez generada y desplegada la letra de la canción pueda generar otra usando los mismo parámetros que previamente había introducido.

Cuadro 4.5: Historia de Usuario 5

Historia de Usuario 5	
Nombre: Validación de la palabra o palabras introducidas.	Usuario: Sistema
Descripción: La página web validará la palabra o palabras proporcionadas por el usuario para con ellas hacer la petición al servidor y generar una letra de canción relacionada a estas.	Validación: Con ayuda del modelo y la palabra o palabras introducidas y procesadas por la página web se genera y despliega la letra generada.

4.1.4. Requerimientos funcionales y no funcionales

A continuación, se presentan los requerimientos funcionales mas importantes de nuestro proyecto:

- Permitirá a cualquier usuario que acceda a la página, generar uno o más versos musicales.
- Permitirá al usuario descargar la letra generada como archivo de texto.
- Permitirá al usuario generar cuantas veces quiera una letra musical.
- Solo se generarán las letras musicales si se cumplen todos los requisitos obligatorios proporcionados por el usuario, los cuales serán: una palabra y el porcentaje de rima de la canción.

Ahora, se presentan algunos de los requerimientos no funcionales más importantes de nuestro proyecto

- El servicio web debe ser capaz de procesar más de diez solicitudes simultaneas.
- La página debe ser capaz de operar adecuadamente con hasta cincuenta usuarios simultáneamente.
- El tiempo que le toma a un usuario promedio en el aprendizaje del sistema de la interfaz deberá de ser menor a cinco minutos.
- La página proporcionará alertas de error que sean informativos y que ayuden a la experiencia de usuario.
- La aplicación web debe tener un diseño responsivo, es decir, que garantice una a adecuada visualización tanto en computadoras, tabletas y dispositivos móviles.

4.1.5. Aspectos Económicos

Las herramientas que usaremos en el trabajo terminal y las cuales validarán la viabilidad del proyecto se presentan a continuación:

Herramientas OpenSource a utilizar en el presente trabajo terminal

- Datamuse API [67]
- MongoDB - SSPL* [68]
- BERT - Apache 2.0 [69]
- Apache - Apache 2.0 [70]
- React - MIT [71]
- Python - PSFL [72]
- Amazon - Elastic 2.0 y SSPL [73]

*SSPL: Aunque es clasificado como OpenSource por parte de MongoDB, al estar basado en la licencia de software libre de GPL3, la Iniciativa de Open Source (OSI) no la reconoce como licencia OpenSource al describirla como una licencia OpenSource "no genuina".

Como se puede apreciar en el software presentado, las licencias son del tipo OpenSource con lo cual validamos la viabilidad económica del proyecto el cual no representará gasto monetario más allá del necesitado para el entrenamiento de la red neuronal, para el cual haremos uso de Amazon SageMaker que tiene un costo de:

Cuota gratuita primeros 2 meses:

- Instancia de bloc de notas: 250 horas.
- Entrenamiento: 50 horas.

Cuotas:

- Instancia de bloc de notas (4 CPU's virtuales y 16Gb de memoria): 0.20 centavos de dólar/hora.
- Entrenamiento (4 CPU's virtuales y 16Gb de memoria): 0.23 centavos de dólar/hora.
- Procesamiento (4 CPU's virtuales y 16Gb de memoria): 0.20 centavos de dólar/hora.

4.1.6. Aspectos Legales

Copyright

Los derechos de autor son una forma de protección para obras originales de autoría fijadas en un medio de expresión tangible. Los derechos de autor cubren tanto las obras publicadas como las que no o son inéditas.[58]

Los derechos de autor protegen un gran rango de obras como lo son: obras literarias, obras musicales, obras pictóricas o escultóricas, coreografías, escenas dramáticas, producciones cinematográficas y demás audiovisuales, programas de cómputo, fotografías, entre muchas otras.

Originalidad

Los derechos de autor protegen la forma en que se expresan las ideas. Esta manera única de expresar las palabras, notas musicales, colores, formas, etc. son elegidas y organizadas u ordenadas. Es la expresión lo que hace que una obra sea original. Esto significa que puede haber muchos trabajos diferentes sobre la misma idea y todos de ellos estarán protegidos por derechos de autor, siempre y cuando expresen esta idea de una manera original.[59]

Según la Ley Federal del Derecho de Autor, la forma en cómo se expresa un autor al momento de crear su obra es lo que es protegido por la ley ante acciones fraudulentas.[60]

4.2. Base de Datos

4.2.1. Descripción

En este apartado se hará el diseño de la base de datos para poder generar un modelo acorde a los requerimientos funcionales del proyecto.

4.2.2. Obtención de la Base de Datos

Para iniciar la búsqueda de la base de datos primero diseñamos un diagrama de datos esenciales que se muestra a continuación para poder generar el modelo:

Se hizo dicha búsqueda en páginas que contienen datasets con licencia del tipo open source, tales como Kaggle [79], Google datasets [80], AWS Open Data [81] y en papers de investigación.

Encontramos diferentes datasets que contaban con algunas de las necesidades del sistema pero por una u otra razón no se podían utilizar, tal fue el caso del dataset de Music4All [82] que contaba con las necesidades del sistema pero el uso de los datos era muy restringido.

En Kaggle [79] se encontró un dataset llamado "Song lyric for 6 musical genres"[78] el cual contiene todos los datos necesarios para el sistema con un total de 160790 letras de canciones.

Dicha base de datos se sacó originalmente de la página de Vagalume [83] e incluye 6 tipos de géneros musicales mencionados a continuación, los cuales son:

- Rock
- Pop
- Hip Hop
- Samba
- Sertanejo
- Funk Carioca

Esta base de datos cuenta con dos tablas que cumplen con las características ideales para realizar el modelo acorde a los requerimientos funcionales.

A continuación se muestran las columnas de la base de datos en la primer tabla, como se puede observar, se utilizarán las columnas “Genre” para separar el dataset en géneros musicales y “Link” que servirá como enlace a la segunda tabla.

En esta segunda tabla que se muestra a continuación se utilizarán las columnas de “Lyric” para poder entrenar el modelo, “Idiom” para seleccionar y utilizar las canciones en el idioma inglés y “ALink” que nos servirá para fusionar la tabla mencionada anteriormente con la actual y así poder usar esas columnas.

4.2.3. Género Musical

El género musical que elegimos es Pop en inglés debido a que es un género musical flexible en su estructura y en su léxico, otros géneros contemplados fueron Hip Hop y Rock que también cuentan con un extenso catálogo de datos pero se encontraron modelos que ya utilizan esos géneros.

4.2.4. Resultados

Al necesitar extraer información precisa para el entrenamiento del modelo no es necesario que la base de datos sea muy compleja, razón por la cual las tablas mostradas en el diagrama entidad-relación se ve así, siendo lo de mayor relevancia el mostrar que un artista puede tener muchas canciones pero una canción le pertenece únicamente a un artista.

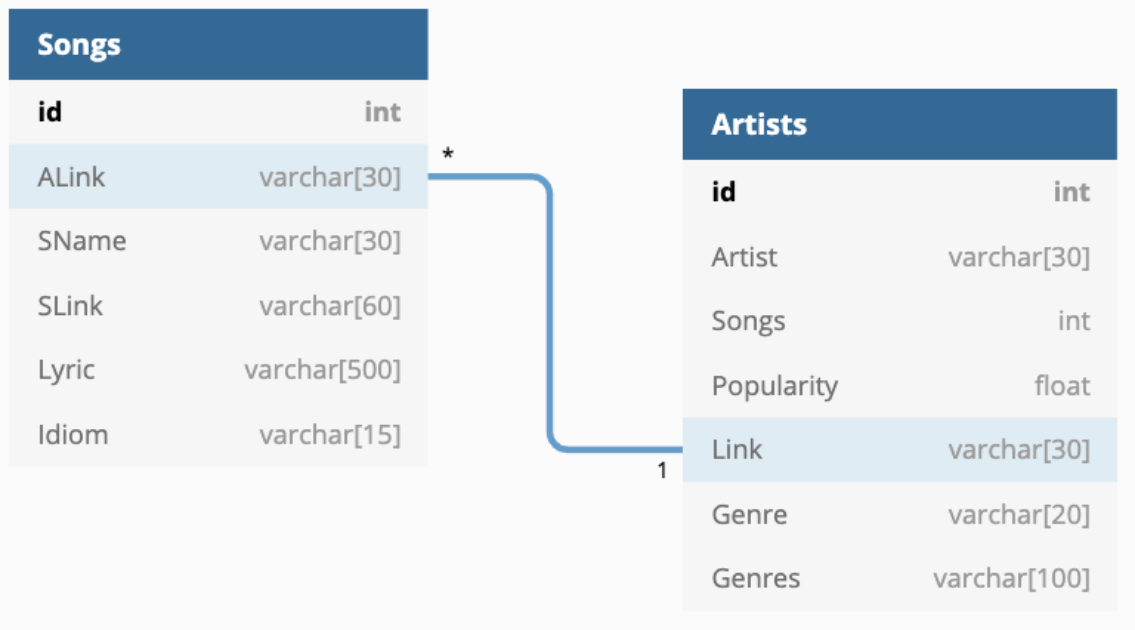


Figura 4.5: Diagrama Entidad Relación de la Base de Datos

4.3. Modelo para Neural Network

4.3.1. Descripción

En este apartado expondremos las herramientas pensadas para el desarrollo del modelo de red neuronal, así como la manera en que se piensa que se puedan integrar a nuestro proyecto y las conexiones que puede tener tanto con la base de datos como con el servidor en la nube o la interfaz web.

4.3.2. Herramientas usadas para generar el modelo

BERT

Como ha sido expuesto con anterioridad BERT (Bidirectional Encoder Representations from Transformers) el cual se usarás para clasificar oraciones, se puede entender de forma más sencilla de la siguiente manera:

- Bidirectional: Para entender el texto es necesario dar un vistazo hacia atrás (a las palabras previas) y hacia adelante (a las palabras que siguen).
- Transformers: El Transformer lee secuencias enteras de tokens de una vez. En un sentido, el modelo es no-direccional, mientras que las redes del tipo LSTM (Long-Short Term Memory) leen de forma secuencial (izquierda-derecha o derecha-izquierda). El mecanismo de atención permite aprender relaciones contextuales entre palabras.

4.3.3. Modelo

Para el diseño del modelo se tiene contemplado el siguiente modelo del cual se puede decir que una vez se tiene la salida del transformer se pasa por una capa de clasificación, posterior a ello se usa la función "Softmax"de Python para calcular la probabilidad de aparición de cada palabra que se tiene en el vocabulario y cuyo resultado final determinará la palabra usada para la máscara (marcada de color rojo).

En última instancia, y de acuerdo a la probabilidad de rima dada por el usuario en la interfaz web, la última palabra puede o no contener una rima que haga juego con alguna de las palabras de la oración previa del verso.

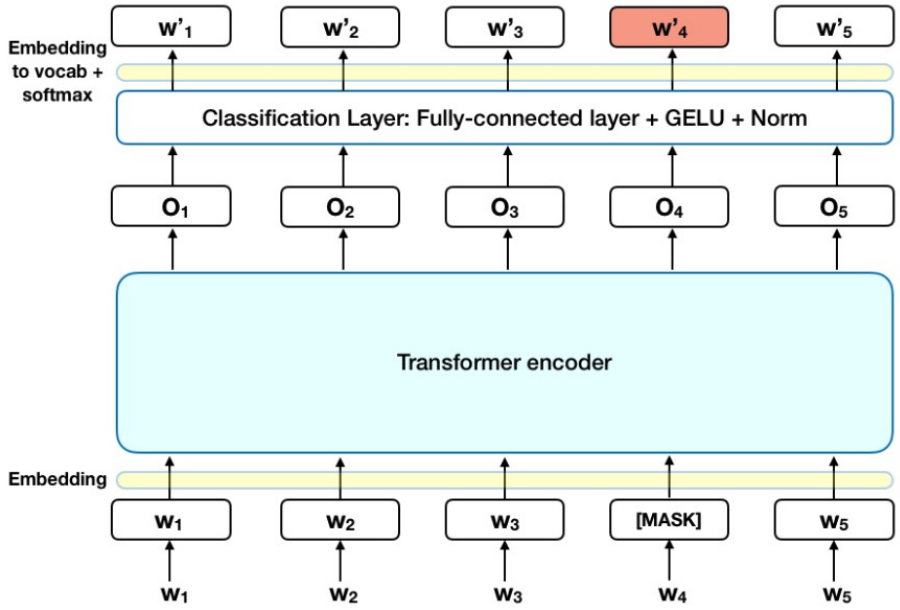


Figura 4.6: Diagrama de diseño del modelo

4.3.4. Resultados

Habiendo hecho el análisis del modelo anterior y tomándolo como referencia para nuestro proyecto podemos adaptarla de forma tal que las entradas sean en su mayoría [MASK] las cuales serán las palabras predichas siendo la razón por la cual todas las palabras derivadas de ésto están en rojo ya que representan una entradas del tipo MASK, y la última palabra [RHYME] estará dada por el porcentaje de coincidencia seleccionado por el usuario y que tiene como objetivo, valga la redundancia, de rimar con la última palabra de la penúltima oración previa:

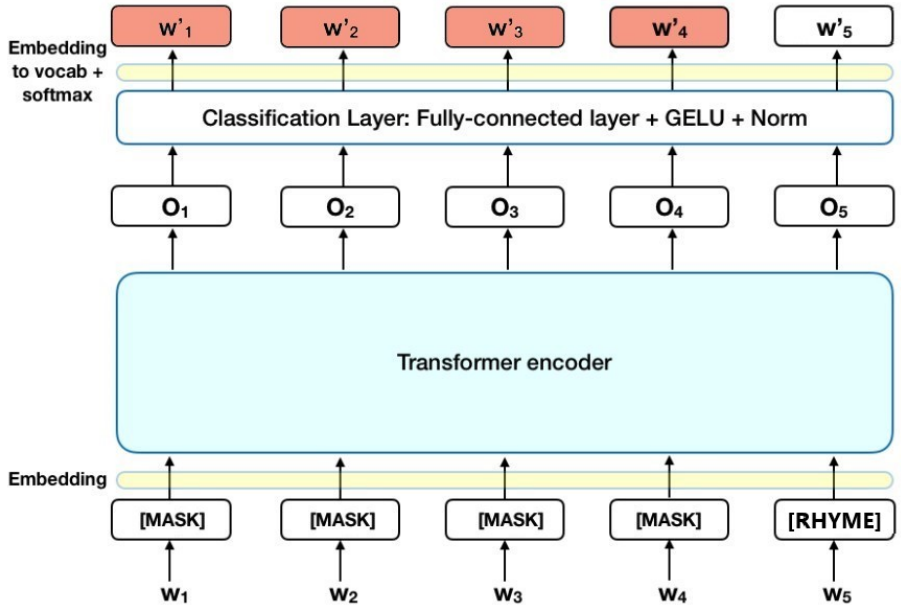


Figura 4.7: Diagrama de diseño del modelo

Siendo el resultado esperado uno muy similar a la estructura que se muestra a continuación y en la que se puede ver lo siguiente:

En todos los casos la palabra que rime tendrá un valor entre 0 y 1 (valor flotante) el cual determinará qué tan alta sería la posibilidad de que la palabra rime, siendo el valor 0 que no rime y el valor 1 representaría una rima segura.

4.3.4.1. Verso

- Estará compuesto de 4 a 6 oraciones y el valor será decidido por un número al azar.
- Las palabras que serán predichas en todo el verso serán de 4 a 5 y el valor será decidido por un número al azar.

4.3.4.2. Pre-coro y puente

- Estará compuesto de 2 a 3 oraciones y el valor será decidido por un número al azar.
- Las palabras que serán predichas en todo el verso serán de 6 a 10 y el valor será decidido por un número al azar.

4.3.4.3. Coro

- Estará compuesto de 3 a 4 oraciones y el valor será decidido por un número al azar.
- Las palabras que serán predichas en todo el verso serán de 3 a 8 y el valor será decidido por un número al azar.

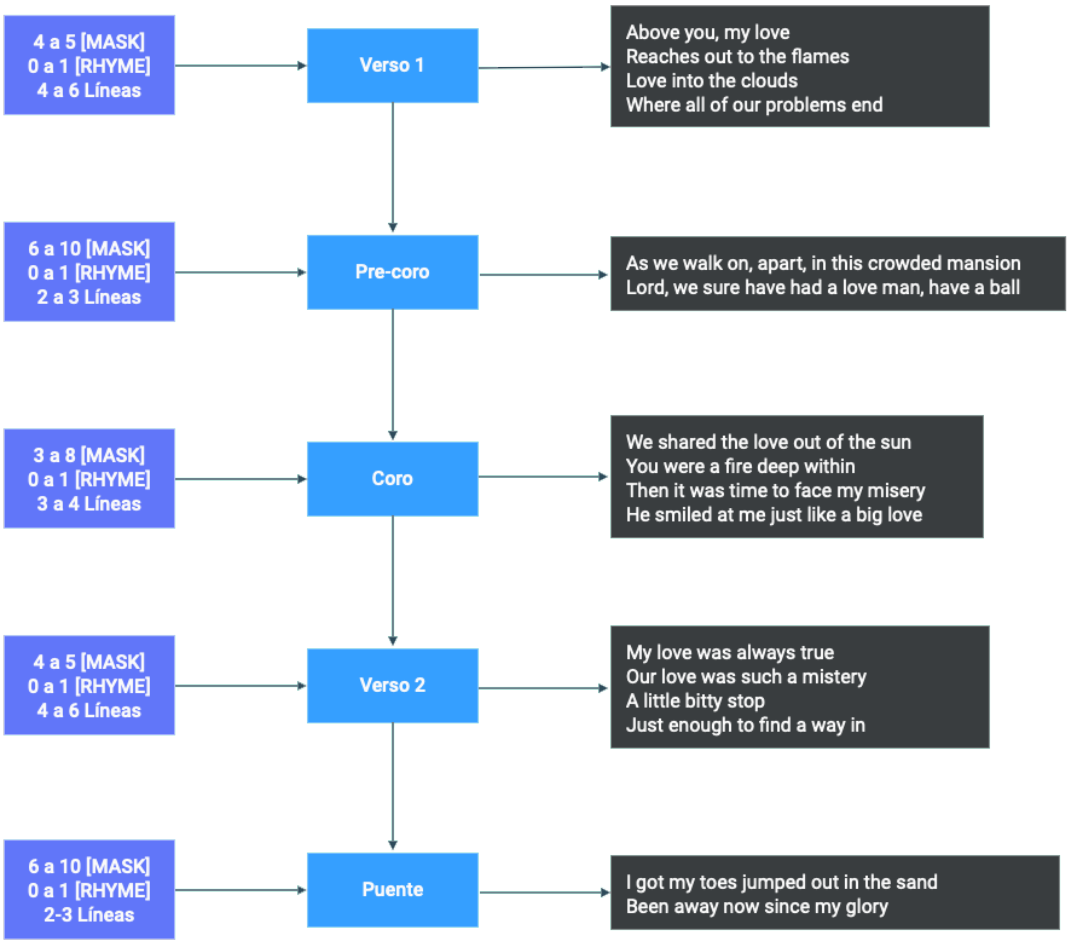


Figura 4.8: Diseño de la estructura del modelo

4.4. Generación del Modelo en la Nube

4.4.1. Descripción

En este apartado se tocarán herramientas que utilizaremos, la interacción entre nosotros y el sistema para suministrar los datos a Amazon Sagemaker, el desarrollo que seguirá AWS para ir desde el entrenamiento del modelo hasta el despliegue del mismo para poder ser utilizado.

4.4.2. Amazon Sagemaker

Amazon SageMaker es un servicio que ayuda a científicos y desarrolladores a construir, entrenar e implementar de manera rápida y sencilla modelos de machine learning de forma rápida. SageMaker suprime las tareas difíciles y tediosas de cada paso del proceso de machine learning para que sea más fácil crear modelos de alta calidad.

Dentro de las virtudes de SageMaker, y razón por la cual lo preferimos sobre la compentencia, es que cuenta con 2 herramientas escenciales las cuales nos servirán al momento de entrenar el modelo:

- SageMaker Experiments: Ayuda a organizar y rastrear iteraciones en los modelos de aprendizaje automático al capturar los parámetros de entrada, configuraciones y resultados y guardarlos como .ºexperimentos", permitiendo que de forma posterior se pueda llevar una línea de tiempo de lo que ha sucedido y comparar los resultados de los experimentos de una manera visual.
- SageMaker Debugger: Registra métricas del entrenamiento en tiempo real, genera advertencias y consejos de solución cuando se detectan problemas de entrenamiento comunes, así como monitorea los recursos del sistema para birndar recomendaciones respecto a cómo reasignar recursos y así reducir los costos.
- Managed Spot Training: Es una herramienta que permite entrenar los modelos de aprendizaje automático con instancias de Amazon EC2 que pueden demorar más al entrenar a cambio de abaratar los costos de entrenamiento hasta en un 90 %.

4.4.3. Suministrado de Datos

Al ingresar a SageMaker lo primero que nos es requerido hacer es una instancia bloc de notas (Jupyter Notebook) dentro del cual escribiremos el código en Python que ocuparemos para ejecutar todo el procedimiento visto en la iteración tres.

4.4.3.1. Instancia de bloc de notas

Dentro de Amazon Sagemaker creamos un bloc de notas (Jupyter Notebook) dentro del cual podremos operar con código Python y el cual tiene como ventaja principal el correr código por bloques evitando de esta manera el tener que reescribir ese código y pudiendo correr todo de manera secuencial nuevamente o insertar nuevos bloques entre los previamente creados según veamos más conveniente.

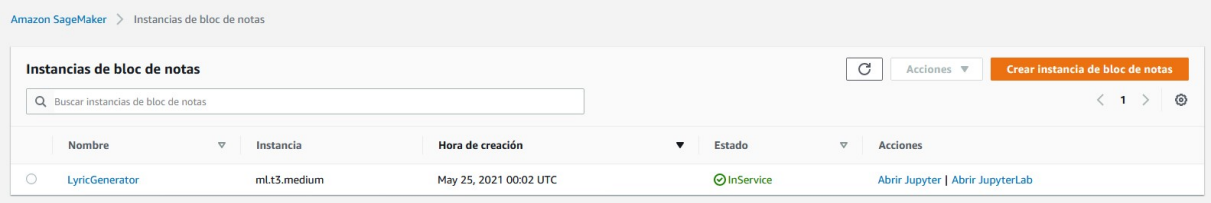


Figura 4.9: Instancia creada en AWS

Habiendo visto el proceso de alimentación de datos para el modelo podemos ahora mostrar de manera general el proceso que tomaremos para el entrenamiento de nuestro modelo, proceso que puede resumirse en el siguiente diagrama:

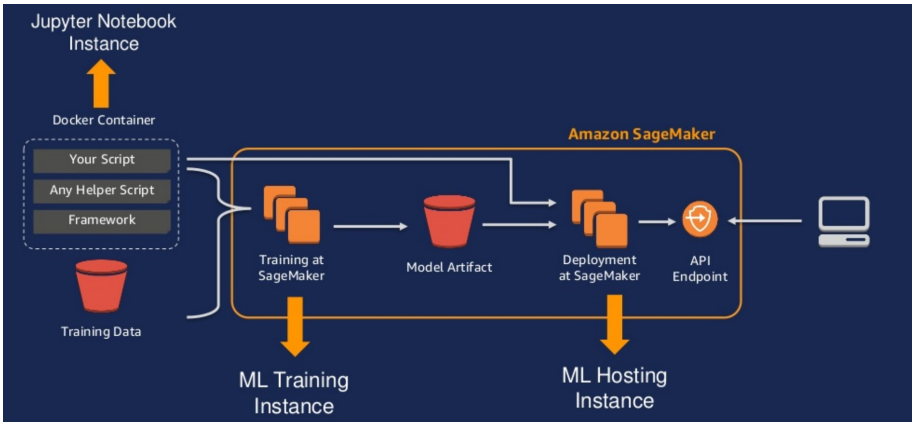


Figura 4.10: Diagrama de interacción general del entrenamiento con SageMaker

A continuación se muestra el proceso seguido para el entrenamiento del modelo:

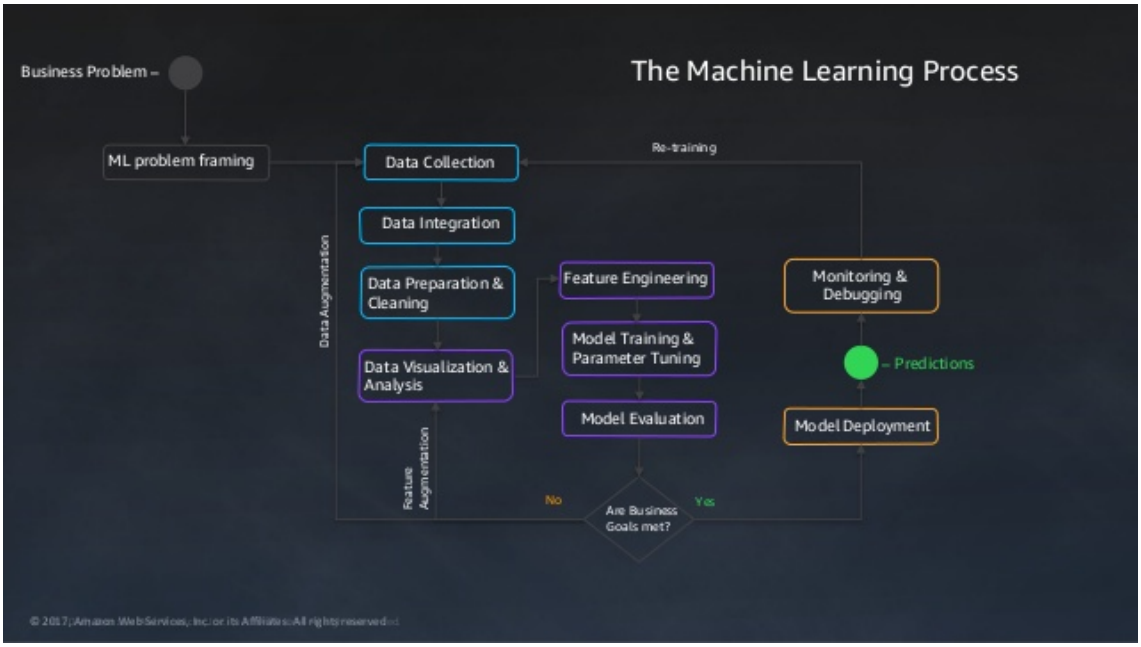


Figura 4.11: Diagrama de diseño del modelo

Dentro del apartado podemos asignar las siguientes actividades como parte de la primer y segunda iteración, las cuales se centran en:

- Colección de datos

- Integración de los datos
- Preparación y limpieza de los datos

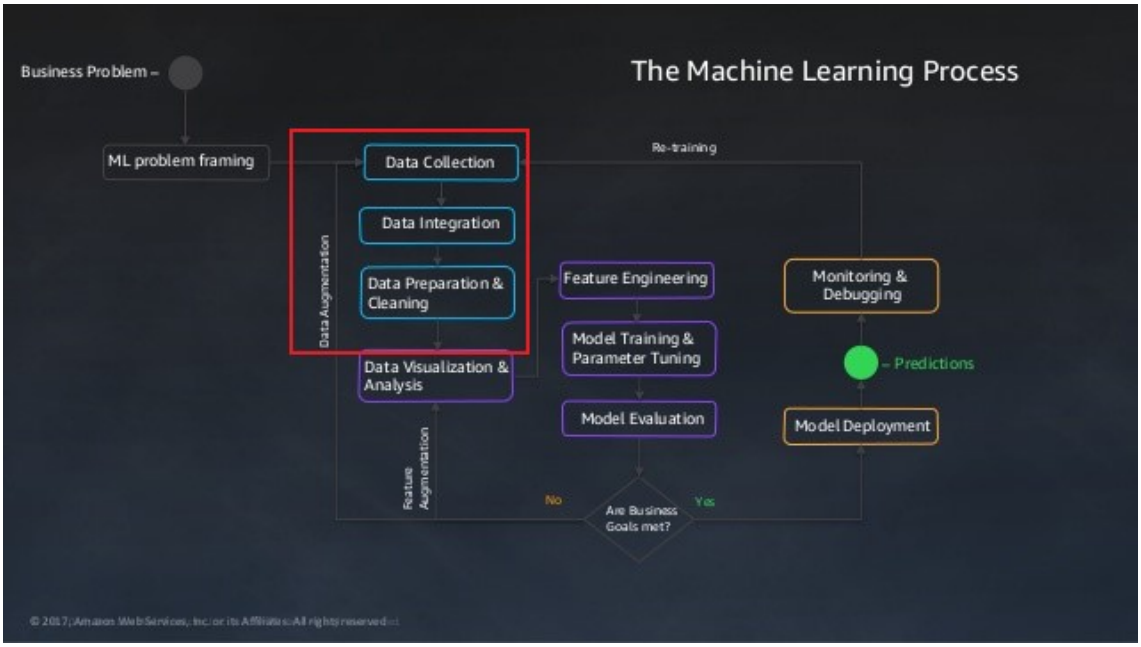


Figura 4.12: Diagrama de diseño del modelo

El ejemplo anterior está pensado para el uso de un contenedor Docker dentro del cual operar una instancia de Jupyter, sin embargo y sólo de ser necesario también se podría manejar el proceso de manera local de forma que el proceso seguido para hacerlo se ve resumido en el diagrama mostrado a continuación:

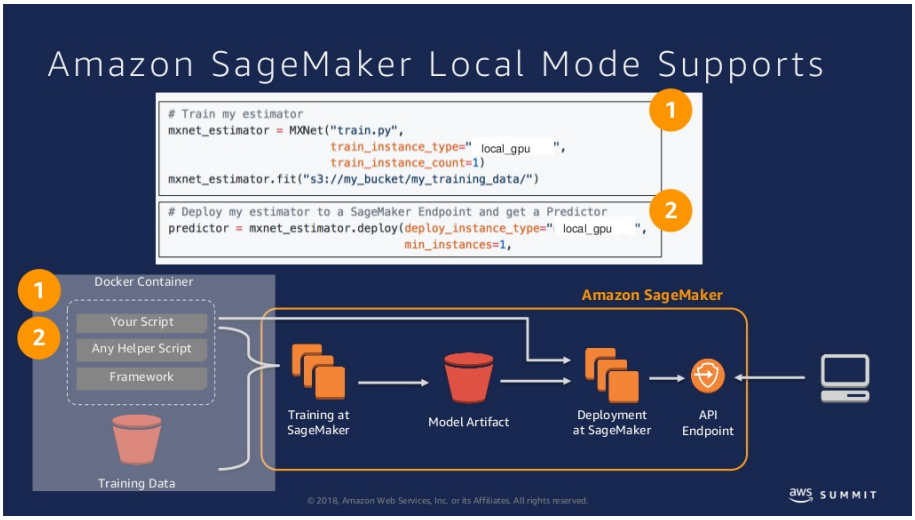


Figura 4.13: Diagrama

Una vez se tienen los pasos necesarios para la generación del modelo podemos describir la interacción del administrador con el sistema de la siguiente manera:

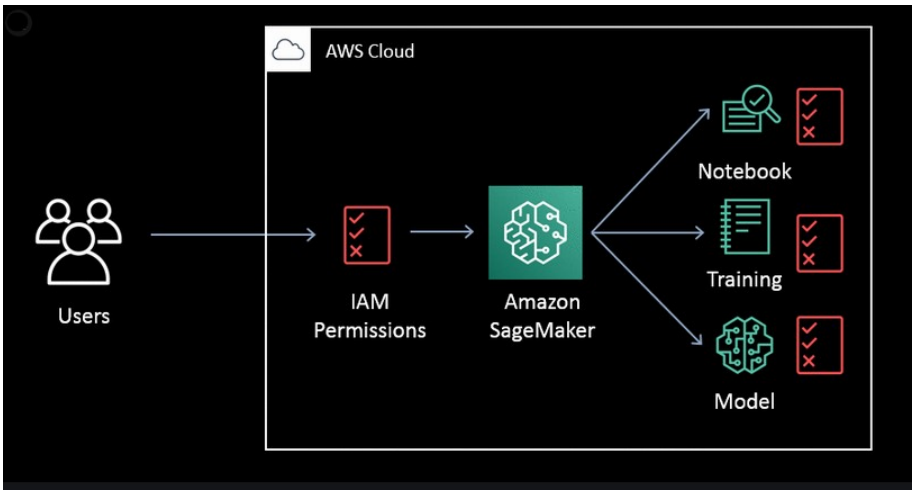


Figura 4.14: Diagrama interacción del administrador con AWS

Instancia de bloc de notas

Dentro de Amazon Sagemaker creamos un bloc de notas (Jupyter Notebook) dentro del cual podremos operar con código Python y el cual tiene como ventaja principal el correr código por bloques evitando de esta manera el tener que reescribir ese código y pudiendo correr todo de manera secuencial

nuevamente o insertar nuevos bloques entre los previamente creados según veamos más conveniente.

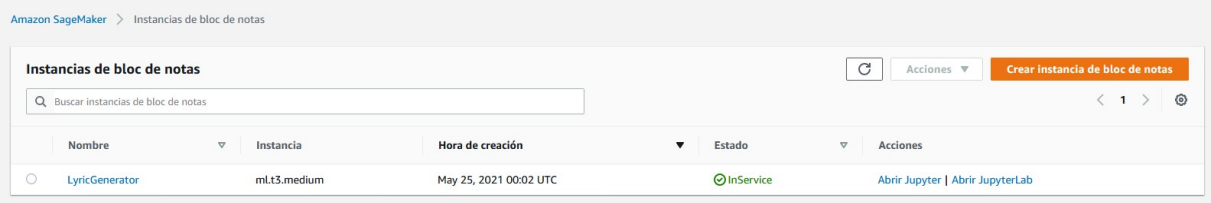


Figura 4.15: Instancia creada en AWS

4.5. Resultados

El resultado final al que podemos llegar dado todo lo anterior es que el camino que debemos de seguir en cuanto al diseño es uno muy similar al mostrado a continuación, puesto que va desde los datos suministrados por nosotros, el entrenamiento del modelo, la información proporcionada al usuario y la retroalimentación a los datos:

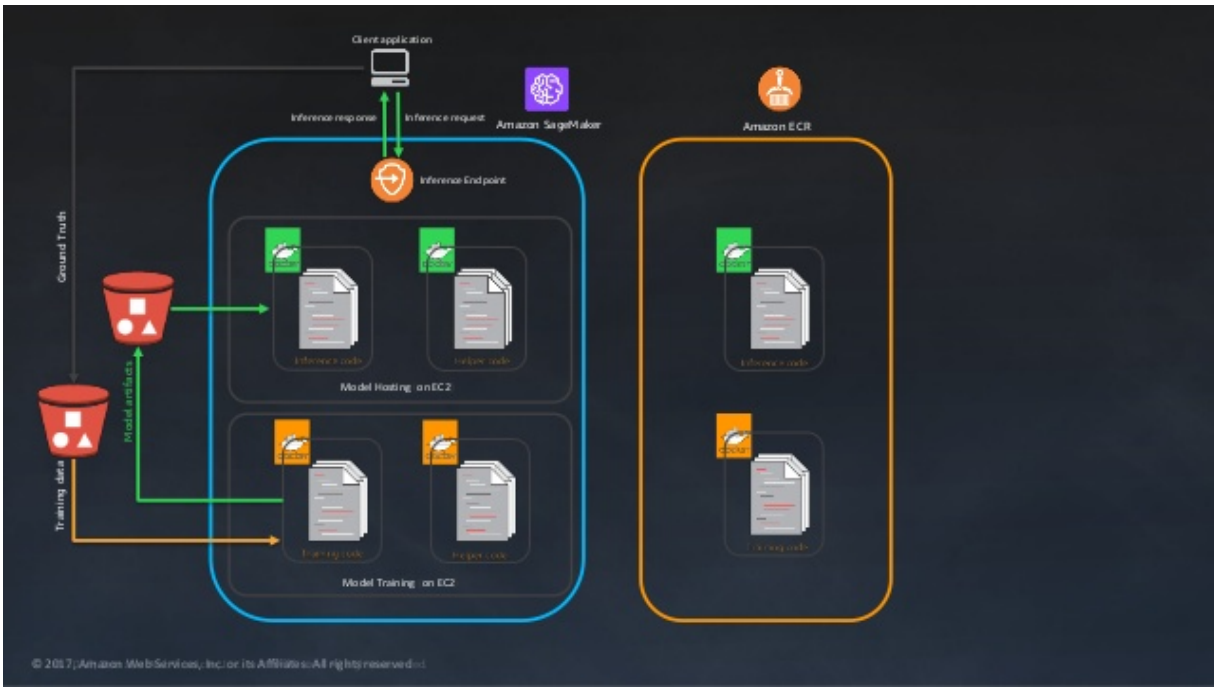


Figura 4.16: Diagrama del posible proceso a seguir para la generación del modelo

4.6. Aplicación web

En este apartado se mostrará un primer boceto de la aplicación web una vez terminada y funcionando.

Figura 4.17: Aplicación web, pagina inicial.

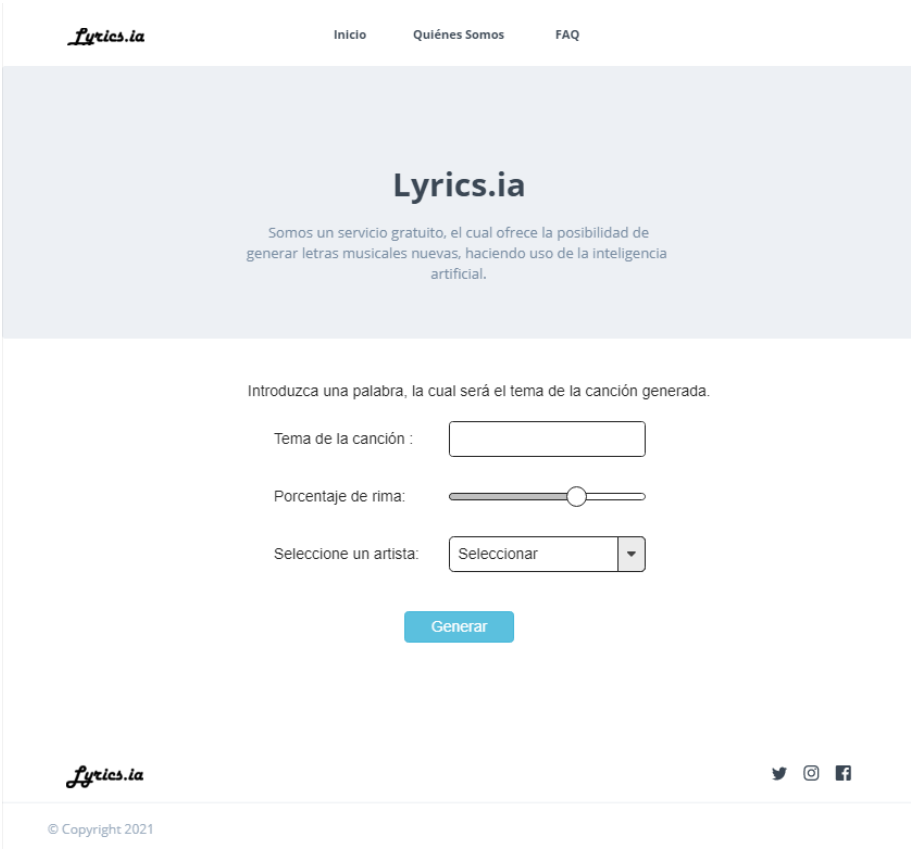
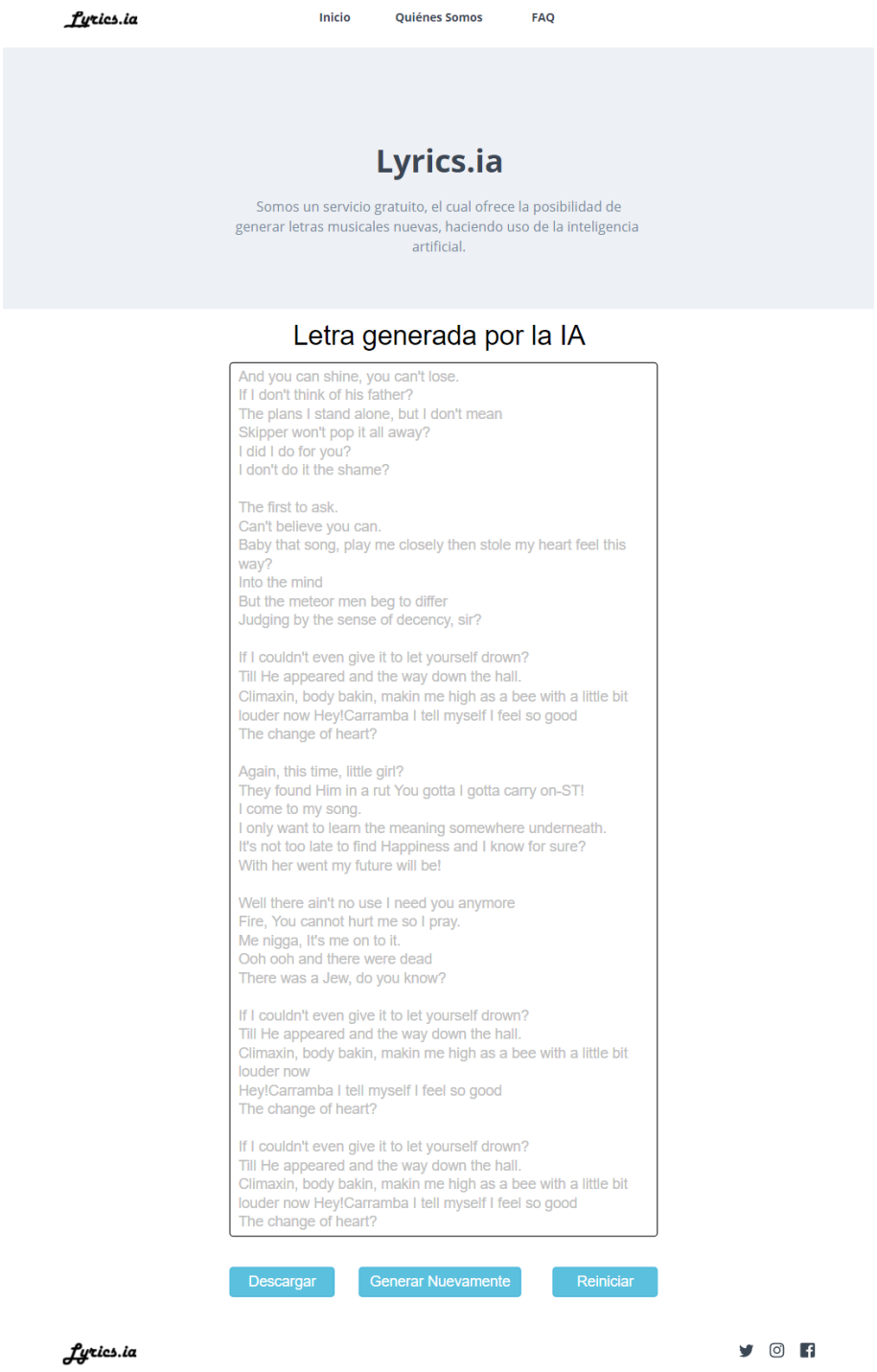


Figura 4.18: Aplicación web, pagina una vez generada una letra musical.



Anexos

Glosario.

Acordes Conjunto de notas musicales sonando al mismo tiempo.[1]. 31

Aprende Adquirir el conocimiento de algo por medio del estudio, análisis o de la experiencia.[1]. 20

Armonia Unión y combinación de sonidos simultáneos y diferentes, pero acordes.[1]. 31

Estado Situación en que se encuentra alguien o algo, y en especial cada uno de sus sucesivos modos de ser o estar.[1]. 33

Estocástico Que está sometido al azar y que es un objeto de análisis estadístico.[1]. 33

Estrofa Conjunto de versos que generalmente se ajustan a una medida y a un ritmo determinado y constante.[1]. 31

Melodia Es una sucesión lineal ordenada y coherente de sonidos musicales los cuales forman una unidad estructurada con un sentido musical, independiente del acompañamiento.
Sucesión de sonidos que por su manera de combinarse resulta agradable de oír.[2]. 31

Máquinas de Estados Finitos Conocidas también como Finite State Machines por su traducción en inglés, nos sirven para realizar procesos bien definidos en un tiempo discreto. Reciben una entrada, realizan un proceso y nos entregan una salida.[3]. 33

Recurrente Algo que vuelve a ocurrir o a aparecer después de un intervalo.[1]. 20

Sistema Conjunto de elementos con relaciones de interacción e interdependencia las cuales le confieren entidad propia al formar un todo unificado.[4]. 20, 33

Acrónimos.

API Application Programm Interface. 39

AWS Amazon Web Services. 5, 38–41, 46, 53, 54

BERT Bidirectional Encoder Representations from Transformers. 4, 23, 25–27, 29, 30, 52

BPE Byte Pair Encoding. 27, 30

CAD Computer Aided Design. 50

CLI Command Line Interface. 53

CLS Classification. 26

CSS Cascading Style Sheet. 48, 49

DBMS Sistema de gestión de base de datos. 30

EC2 Elastic Compute Cloud. 38, 41, 53, 54

FQDN Fully Qualified Domain Name. 37

GCP Google Cloud Platform. 5, 39–41

GPT Generative Pretrained Transformer. 4, 27–30, 52

GPU Graphic Processor Unit. 38, 41

HDD Hard Drive Disk. 43, 44, 55

HTML Hypertext Markup Language. 48, 49

HTTP Hypertext Transfer Protocol. 35, 36, 54

HTTPS Hypertext Transfer Protocol Secure. 49

JSON Javascript Object Notation. 30

MLM Modelado de Lenguaje Enmascarado. 25, 52

MVC Modelo-Vista-Controlador. 34

ORM Object Relational Manager. 34, 50

PLN Procesamiento del Lenguaje Natural. 22, 23, 25, 27, 30, 52

S3 Simple Storage Service. 54

SDK Software Development Kit. 40

SEP Separate. 26

SQL Structured Query Language. 49, 50

SSD Solid State Drive. 43, 44, 55

SSL Secure Socket Layer. 37, 49

TLS Transport Layer Security. 49

UI User Interface. 39

URL Uniform Source Locator. 34, 50

VPS Virtual Private Server. 35, 54

WIP Work In Progress. 12

WSGI Web Server Gateway Interface. 34, 50, 51

XML Extensible Markup Language. 49

Bibliografía

[1] Real Academia Española (2020, noviembre), Diccionario de la lengua española, [En línea]. Disponible: https://dle.rae.es [Último acceso: 10 de diciembre del 2020].

[2] Oxford Lexico (2020, noviembre), Definitions, Meanings, Synonyms, and Grammar by Oxford, [En línea]. Disponible: https://www.lexico.com [Último acceso: 2 de diciembre del 2020].

[3] J. A. Gutiérrez Orozco, Escuela Superior de Cómputo (2008, septiembre 15), Máquinas de Estados Finitos, [En línea]. Disponible: http://delta.cs.cinvestav.mx/ mcintosh/cellularautomata/Summer_Research_files/maquinasef.pdf [Último acceso: 15 de diciembre del 2020].

[4] O. Jaramillo, Universidad Nacional Autonoma de México (2007, mayo 03), El concepto de Sistema, [En línea]. Disponible: https://www.ier.unam.mx/ ojs/pub/Termodinamica/node9.html [Último acceso: 2 de diciembre del 2020].

[5] Chaney, D. (2012). The Music Industry in the Digital Age: Consumer Participation in Value Creation. International Journal of Arts Management, (1), pp. 15.

[6] Sutskever, I., Martens, J., Hinton, G. E. (2011, January). Generating text with recurrent neural networks. In ICML.

[7] Kim, Y. (2014). Convolutional neural networks for sentence classification. arXiv preprint arXiv:1408.5882.

[8] Monteith, K., Martinez, T. R., & Ventura, D. (2012, May). Automatic Generation of Melodic Accompaniments for Lyrics. In ICCC, pp. 87-94.

[9] Nikolov, N. I., Malmi, E., Northcutt, C. G., Parisi, L. (2020). Conditional Rap Lyrics Generation with Denoising Autoencoders. arXiv preprint arXiv:2004.03965.

[10] Baker, F. A. (2015). What about the music? Music therapists' perspectives on the role of music in the therapeutic songwriting process. Psychology of Music, 43(1), pp. 122-139.

[11] Guerrero, J. (2012). El género musical en la música popular: algunos problemas para su caracterización. Trans. Revista transcultural de música, (16), pp. 1-22.

[12] Wang, A., & Cho, K. (2019). Bert has a mouth, and it must speak: Bert as a markov random field language model. arXiv preprint arXiv:1902.04094.

[13] Honnibal, M.,& Montani, I. (2017). spaCy 2: Natural language understanding with Bloom embeddings, convolutional neural networks and incremental parsing.

[14] V. Advani (2021, febrero 11), What is Artificial Intelligence? How does AI work, Types and Future of it?, [En línea]. Disponible:https://www.mygreatlearning.com/blog/what-is-artificial-intelligence/ [Último acceso: 6 de abril del 2021].

[15] IBM Corporation (2021, marzo 31), Artificial Intelligence (AI), [En línea]. Disponible: https://www.ibm.com/cloud/learn/what-is-artificial-intelligence [Último acceso: 6 de abril del 2021].

[16] L. Hardesty (2017, abril), Explained: Neural networks, [En línea]. Disponible: https://news.mit.edu/2017/explained-neural-networks-deep-learning-0414 [Último acceso: 15 de noviembre del 2020].

[17] amBientech (2019, julio 30), ¿Qué es la neurona?, [En línea]. Disponible: https://ambientech.org/la-neurona [Último acceso: 6 de abril del 2021].

[18] National Cancer Institute (2021), Neurons & Glial Cells, [En línea]. Disponible: https://training.seer.cancer.gov/brain/tumors/anatomy/neurons.html [Último acceso: 30 de abril del 2021].

[19] C. Gershenson (2012, octubre), Artificial Neural Networks for Beginners, [En línea]. Disponible: https://www.uv.mx/mia/files/2012/10/Artificial-Neural-Networks-for-Beginners.pdf [Último acceso: 6 de abril del 2021].

[20] Rubik's Code (2018, febrero), How do Artificial Neural Networks learn?, [En línea]. Disponible: https://rubikscode.net/2018/01/15/how-artificial-neural-networks-learn/ [Último acceso: 7 de abril del 2021].

[21] S. Leijnen, F. van Veen (2020, mayo), The Neural Network Zoo, [En línea]. Disponible: https://www.researchgate.net/publication/341373030_The_Neural_Network_Zoo [Último acceso: 20 de noviembre del 2020].

[22] A. Mehta (2019, enero 25), A Comprehensive Guide to Types of Neural Networks, [En línea]. Disponible: https://www.digitalvidya.com/blog/types-of-neural-networks/ [Último acceso: 15 de noviembre del 2020].

[23] P. Shukla, R. Iriondo (2020, agosto 11), Main Types of Neural Networks and its Applications, [En línea]. Disponible: https://medium.com/towards-artificial-intelligence/main-types-of-neural-networks-and-its-applications-tutorial-734480d7ec8e [Último acceso: 20 de noviembre del 2020].

[24] IBM Cloud Education (2020, septiembre 14), What are Recurrent Neural Networks?, [En línea]. Disponible: https://www.ibm.com/cloud/learn/recurrent-neural-networks [Último acceso: 26 de marzo del 2021].

[25] ref IBM Cloud Education (2020, julio 2), Natural Language Processing [En línea]. Disponible: https://www.ibm.com/cloud/learn/natural-language-processing [Ultimo acceso: 20 de abril del 2021].

[26] ref A. Vaca (2020, mayo), Transformers en Procesamiento del Lenguaje Natural, [En línea]. Disponible: https://www.iic.uam.es/innovacion/transformers-en-procesamiento-del-lenguaje-natural/ [Ultimo acceso: 15 de abril del 2021].

[27] ref A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. Gomez, L. Kaiser & I. Polosukhin (2017, diciembre), Attention Is All You Need, [En línea]. Disponible: https://arxiv.org/pdf/1706.03762.pdf [Ultimo acceso: 15 de abril del 2021].

[28] ref J. Devlin, M. Chang, K. Lee, K. Toutanova (2019, mayo), BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding, [En línea]. Disponible: https://arxiv.org/pdf/1810.04805.pdf [Ultimo acceso: 11 de abril del 2021].

[29] ref R. Horev (2018, noviembre), BERT Explained: State of the art language model for NLP, [En línea]. Disponible:

https://towardsdatascience.com/bert-explained-state-of-the-art-language-model-for-nlp-f8b21a9b6270 [Ultimo acceso: 11 de abril del 2021].

[30] ref B. Lutkevich (2020, enero), BERT language model, [En línea]. Disponible: https://searchenterpriseai.techtarget.com/definition/BERT-language-model [Ultimo acceso: 12 de abril del 2021].

[31] ref A. Radford, J. Wu, D. Amodei, D. Amodei, J. Clark, M. Brundage & I. Sutskever (2020, enero), Better Language Models and Their Implications, [En línea]. Disponible: https://openai.com/blog/better-language-models/ [Ultimo acceso: 14 de abril del 2021].

[32] ref M. Edward (2019, febrero), Too powerful NLP model (GPT-2), [En línea]. Disponible: https://towardsdatascience.com/too-powerful-nlp-model-generative-pre-training-2-4cc6afb6655 [Ultimo acceso: 14 de abril del 2021].

[33] ref A. Radford, K. Narasimhan, T. Salimans & I. Sutskever (2018), Improving Language Understanding by Generative Pre-Training, [En línea]. Disponible: https://s3-us-west-2.amazonaws.com/openai-assets/research-covers/language-unsupervised/language_understanding_paper.pdf [Ultimo acceso: 14 de abril del 2021].

[34] ref J. Montantes(2019, mayo), Examining the Transformer Architecture, [En línea]. Disponible: https://towardsdatascience.com/examining-the-transformer-architecture-part-1-the-openai-gpt-2-controversy-feceda4363bb [Ultimo acceso: 14 de abril del 2021].

[35] ref J. Vig(2019, marzo), GPT-2: Understanding Language Generation through Visualization, [En línea]. Disponible: https://towardsdatascience.com/openai-gpt-2-understanding-language-generation-through-visualization-8252f683b2f8 [Ultimo acceso: 14 de abril del 2021].

[36] Oracle México (2020, noviembre), ¿Qué es una base de datos?, [En línea]. Disponible: https://www.oracle.com/mx/database/what-is-database/ [Último acceso: 20 de noviembre del 2020].

[37] Escribir Canciones (2008), Estructura y elementos de una canción, [En línea]. Disponible: https://dle.rae.es [Último acceso: 2 de diciembre del 2020].

[38] Swing this Music (2008), ¿QUÉ SECCIONES PUEDE TENER UNA CANCIÓN?española, [En línea]. Disponible: https://sites.google.com/site/swingthismusiccast/interpretacio/estructura-cancion/secciones-de-una-cancion [Último acceso: 2 de diciembre del 2020].

[39] J. R. Norris (1997), Markov Chains, [En línea]. Disponible: https://cape.fcfm.buap.mx/jdzf/cursos/procesos/libros/norris.pdf [Último acceso: 15 de diciembre del 2020].

[40] Flask (2019, julio 04), Flask Documentation, [En línea]. Disponible: https://flask.palletsprojects.com/en/1.0.x/ [Último acceso: 16 de diciembre del 2020].

[41] NGINx (2016, mayo 17), What is NGINX?, [En línea]. Disponible: https://www.nginx.com/resources/glossary/nginx/ [Último acceso: 27 de mayo del 2021].

[42] Mozilla.org (2021, febrero 19), HTML basics, [En línea]. Disponible: https://developer.mozilla.org/en-US/docs/Learn/Getting_started_with_the_web/HTML_basics [Último acceso: 15 de mayo del 2021].

[43] Mozilla.org (2021, mayo 14), HTML5, [En línea]. Disponible: https://developer.mozilla.org/es/docs/Web/Guide/HTML/HTML5 [Último acceso: 15 de mayo del 2021].

[44] W3C (2016), HTML & CSS, [En línea]. Disponible: [Último acceso: 15 de mayo del 2021].

[45] Mozilla.org (2021, abril 27), What is JavaScript?, [En línea]. Disponible: https://developer.mozilla.org/en-US/docs/Learn/JavaScript/First_steps/What_is_JavaScript [Último acceso: 15 de mayo del 2021].

[46] OpenSSL (2018), OpenSSL, [En línea]. Disponible: https://www.openssl.org [Último acceso: 15 de mayo del 2021].

[47] Documentation Group. (n.d.). Welcome! - The Apache HTTP Server Project. Apache. Retrieved April 6, 2021, from https://httpd.apache.org/

[48] G., D. (2021, March 9). What is Apache? An In-Depth Overview of Apache Web Server. Hostinger Tutorials. https://www.hostinger.com/tutorials/what-is-apache

[49] What is a web server? - Learn web development — MDN. (2021, January 27). MDN Web Docs. https://developer.mozilla.org/en-US/docs/Learn/Common_questions/What_is_a_web_server

[50] https://letsencrypt.org/es/

[51] What is SSL? (2017, May 15). SSLSHOPPER. https://www.sslshopper.com/what-is-ssl.html

[52] Verisign. (2015, August 30). ¿Qué es un certificado SSL? – Verisign. Verisign NameStudio. https://www.verisign.com/es_LA/website-presence/online/ssl-certificates/index.xhtml

[53] Bavati, I. (2020, September 22). Data Science in the Cloud - Towards Data Science. Medium. https://towardsdatascience.com/data-science-in-the-cloud-239b795a5792 (September 2020)

[54] G. (2021, March 5). Comparing Google Cloud Platform, AWS and Azure - Georgian Impact Blog. Medium. https://medium.com/georgian-impact-blog/comparing-google-cloud-platform-aws-and-azure-d4a52a3adbd2

[55] J. (2020, July 30). Choosing Your Deep Learning Infrastructure: The Cloud vs. On-Prem Debate. Determined AI. https://determined.ai/blog/cloud-v-onprem

[56] I., A. (2015, October 2). Jupyter/IPython Notebook Quick Start Guide. Jupyter Notebook Beginner Guide. https://jupyter-notebook-beginner-guide.readthedocs.io/en/latest/what_is_jupyter.html

[57] Jones, E. (2021, March 25). Google Cloud vs AWS in 2021 (Comparing the Giants). Kinsta. https://kinsta.com/blog/google-cloud-vs-aws/

[58] Copyright.gov (2021), Copyright in General, [En línea]. Disponible: https://www.copyright.gov/help/faq/faq-general.html#what [Último acceso: 04 de mayo del 2021].

[59] World Intellectual Property Organization (2007), THE ARTS AND COPYRIGHT, [En línea]. Disponible: https://www.wipo.int/edocs/pubdocs/en/copyright/935/wipo_pub_935.pdf [Último acceso: 04 de mayo del 2021].

[60] CÁMARA DE DIPUTADOS DEL H. CONGRESO DE LA UNIÓN (2014, julio 14), LEY FEDERAL DEL DERECHO DE AUTOR, [En

línea]. Disponible: https://www.ucol.mx/content/cms/13/file/federal/LEY_FED_DEL_DERECHO_DEL_AUTOR.pdf [Último acceso: 05 de mayo del 2021].

[61] Python (2021), What is Python? Executive Summary, [En línea]. Disponible: https://www.python.org/doc/essays/blurb/ [Último acceso: 24 de abril del 2021].

[62] MongoDB. (2019), ¿Qué es MongoDB?, [En línea]. Disponible: https://www.mongodb.com/es. [Último acceso: 26 Marzo 2021].

[63] Universitat de Valéncia (2016), ¿Qué son las cookies? [En línea]. Disponible: https://www.uv.es/uvweb/universidad/es/politica-privacidad/politica-cookies/son-cookies–1285919089226.html [Último acceso: 26 Marzo 2021]

[64] Noteworthy Programming Masterpiece (2019), openssl-nodejs [En línea]. Disponible: https://www.npmjs.com/package/openssl-nodejs [Último acceso: 26 Marzo 2021]

[65] The Apache Software Foundation (2019), Apache Tomcat [En línea]. Disponible: http://tomcat.apache.org/index.html [Último acceso: 26 Marzo 2021]

[66] Oracle (2018), MySQL [En línea]. Disponible: https://www.oracle.com/mysql/ [Último acceso: 26 Marzo 2021]

[67] OneLook Dictionary Search (2016), Privacy Policy [En línea]. Disponible: https://www.onelook.com/about.shtml [Último acceso: 19 Mayo 2021]

[68] MongoDB Inc (2018), MongoDB Licensing [En línea]. Disponible: https://www.mongodb.com/community/licensing [Último acceso: 19 Mayo 2021]

[69] Google Research: BERT (2018), LICENSE [En línea]. Disponible: https://github.com/google-research/bert/blob/master/LICENSE [Último acceso: 19 Mayo 2021]

[70] Apache (2004), The Apache License, Version 2.0 [En línea]. Disponible: https://httpd.apache.org/docs/current/license.html [Último acceso: 19 Mayo 2021]

[71] Facebook: React (2018), LICENSE [En línea]. Disponible: https://github.com/facebook/react/blob/master/LICENSE [Último acceso: 19 Mayo 2021]

[72] Python (2001), History and License [En línea]. Disponible: https://docs.python.org/3/license.html [Último acceso: 19 Mayo 2021]

[73] Elastic (2021), Elastic License 2.0 [En línea]. Disponible: https://www.elastic.co/es/licensing/elastic-license [Último acceso: 19 Mayo 2021]

[74] Google Cloud Platform (2021), Descripción general de Google Cloud [En línea]. Disponible: https://cloud.google.com/docs/overview [Último acceso: 25 Mayo 2021]

[75] Amazon Web Services (2021), Informática en la nube con AWS [En línea]. Disponible: https://aws.amazon.com/es/what-is-aws/ [Último acceso: 25 Mayo 2021]

[76] Azure (2021), Conozca Azure [En línea]. Disponible: https://azure.microsoft.com/es-mx/overview/ [Último acceso: 25 Mayo 2021]

[77] ITM Platform (2010), Matriz de Evaluación de Riesgos [En línea]. Disponible: https://www.itmplatform.com/es/recursos/matriz-de-evaluacion-de-riesgos /#RiskSetId=rh25q/6IQWFCFvYIwSUF6A== [Último acceso: 26 Mayo 2021]

[78] Kaggle (2019, Noviembre 17), Song lyrics from 6 musical genres [En línea]. Disponible: https://www.kaggle.com/neisse/scrapped-lyrics-from-6-genres [Último acceso: 25 Mayo 2021]

[79] Kaggle (2010), How to Use Kaggle [En línea]. Disponible: https://www.kaggle.com/docs/datasets [Último acceso: 25 Mayo 2021]

[80] Google (2018, Septiembre 5), Dataset Search [En línea]. Disponible: https://datasetsearch.research.google.com [Último acceso: 25 Mayo 2021]

[81] Amazon, Open Data on AWS [En línea]. Disponible: https://aws.amazon.com/es/opendata/?wwps-cards.sort-by=item.additionalFields.sortDate&wwps-cards.sort-order=desc [Último acceso: 25 Mayo 2021]

[82] Music4All (2020), A New Music Database and Its Applications [En línea]. Disponible: https://sites.google.com/view/contact4music4all [Último acceso: 25 Mayo 2021]

[83] Vagalume (2002), Vagalume: Music e tudo [En línea]. Disponible: https://www.vagalume.com.br/ [Último acceso: 25 Mayo 2021]