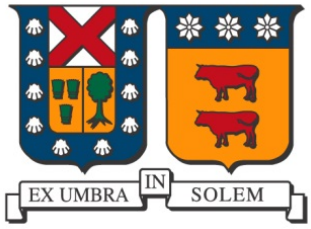
UNIVERSIDAD TÉCNICA FEDERICO SANTA MARÍA
DEPARTAMENTO DE INFORMÁTICA
VALPARAÍSO - CHILE



“ARQUITECTURA DE SOFTWARE PARA ANÁLISIS DE TEXTOS UTILIZANDO INTELIGENCIA ARTIFICIAL”

IGNACIO RODRIGO VALENZUELA ALBORNOZ

MEMORIA PARA OPTAR AL TÍTULO DE
INGENIERO CIVIL EN INFORMÁTICA

Profesor Guía: CECILIA REYES COVARRUBIAS
Profesor Correferente: HUMBERTO LOBOS SUCARRAT

Marzo - 2021

DEDICATORIA

Dedico este trabajo a mi madre, padre y hermana, quienes han sido los pilares durante mi vida y han definido la persona que soy.

Además, dedico este trabajo a mi primo Fabián por su ejemplo de vida y por su compañía durante muchos años, y a mi tío Marcos por su apoyo incondicional y considerarme un hijo más. Sé que hubieran estado orgullosos de mí con este logro.

AGRADECIMIENTOS

Agradezco infinitamente a mi núcleo familiar. A mi madre por su cariño y apoyo incondicional, a mi padre por los consejos y siempre considerarme el mejor en todo, y a mi hermana por su sinceridad y preocupación.

Agradezco a mis tíos y primos de quienes tengo hermosos recuerdos y nuevos por construir. Me transmitieron una importancia considerable respecto al concepto de familia.

Agradezco a mis amigos de colegio, con quienes tras egresar, nos seguimos reuniendo. Gracias por su amistad, apoyo y mantener la amistad, aunque la universidad me alejase de las reuniones más recurrentes con ustedes.

Agradezco a la gran cantidad de amigos (de diversas carreras) que hice en la universidad. Conocer una variedad de gente distinta hizo el transcurso por los años de universidad muy entretenidos, y conocer diversos puntos de vista que influyeron en mi forma de ver la vida.

Agradezco a Gonzalo Tello e Ítalo, por su amistad, a pesar de que fuimos por diferentes caminos académicos y hemos estado siempre uno para el otro.

Agradezco a Sebastián Gallardo, Gerardo Neumann, Macarena Andrade y Bastián Quezada, con los cuales formamos un gran equipo en la Feria del Software, donde la pasamos bien y crecimos como personas.

Agradezco a Gonzalo Valenzuela por aceptarme en su vivienda por 2 años, ayudarme en adaptarme en la vida en Santiago y aguantar la insistencia de solicitar comida a domicilio.

Agradezco a la profesora Cecilia Reyes por su pedagogía que me marcó mucho durante el transcurso de la carrera, también la voluntad de participar en este trabajo de título. También agradezco a Humberto Lobos por confiar en mis capacidades y darme todas las herramientas necesarias para realizar un buen trabajo de título.

Agradezco a la empresa QServus por permitirme realizar este trabajo con ellos, y a todos los trabajadores que siempre me ayudaron cuando solicité ayuda.

Finalmente, agradezco a Susana Fernández por su presencia, amor, apoyo y siempre mostrarme que soy capaz de muchas cosas.

RESUMEN

Resumen— Las respuestas de texto en una encuesta son una de las fuentes más complejas de analizar y automatizar, ya que los computadores no comprenden de forma directa el lenguaje y su contexto. La empresa chilena QServus está consciente de esta problemática y encuentra en los algoritmos de inteligencia artificial el camino correcto para lograrlo. En este trabajo se enfrentan dos problemas en este aspecto, que son el análisis de sentimiento y la categorización de textos. Se desarrollarán los algoritmos de aprendizaje automático que resolverán esos problemas y se implementarán en una API que se encargará de producir los análisis e indicadores para que las aplicaciones de QServus puedan acceder a ellas de forma fácil y simple. Se logra implementar la API siguiendo un diseño ordenado y estandarizado, se prueban de forma satisfactoria ambos modelos de sentimiento y categoría y se crean 6 nuevos tipos de indicadores para la plataforma.

Palabras Clave— Aprendizaje Automático, REST API, Análisis de Sentimiento, Categorización de Texto, Indicadores.

ABSTRACT

***Abstract*—** Text answers in a survey are one of the most complex sources to analyse and automatize, because computers can't comprehend in a direct way the language and their context. The chilean company QServus is aware about this problem and believes that the artificial inteligence algorithms are the apropiate choice to achieve this goal. In this work we face two main problems in this field which are sentiment analysis and text categorization. We will develop machine learning algorithms will solve this problems and will be implemented into an API that will be responsible to produce all the analysis and different indicators (charts), with the intention to differents applications of QServus can access to all of that in a easy and simple way. We implemented the API following all the standard and proper software design, we tested with good results both models of sentiment and category. Also, we created six new types of indicators for the platform.

***Keywords*—** Machine Learning, REST API, Sentiment Analysis, Text Categorization, Indicators.

GLOSARIO

ANN: Artificial Neural Network
Un modelo de aprendizaje supervisado inspirado en su homólogo biológico.

API: Aplication Programming Interface
Diseño de software para la comunicación entre varios intermediarios de software.

BPTT: Back-propagation through time
Es el algoritmo de entrenamiento y aprendizaje para las redes neuronales recurrentes.

HTTP: Hypertext Transfer Protocol
Es un protocolo para la transmisión de documentos de *hypermedia*.

IDF: Inverse Document Frequency
Métrica que define la frecuencia de palabras respecto a la cantidad de documentos que aparece.

JSON: JavaScript Object Notation
Es un formato de datos intercambiables muy liviano.

LSTM: Long short-term memory
Es una estructura mejorada de las redes neuronales recurrentes.

NLP: Natural Language Processing
Área del lenguaje e informática que se centra en la comunicación entre máquinas y el lenguaje humano.

NPS: Net Promoter Score
Métrica que representa el porcentaje de usuarios que recomiendan un producto o servicio.

ReLU: Rectified Linear Unit
Función de activación que considera solo salidas positivas, dejando en cero las negativas.

REST: Representational State Transfer
Estándar de diseño de software para aplicaciones que se comunican con múltiples servicios web.

TF: Term Frequency
Métrica que representa la frecuencia de una palabra en un documento.

URI: Uniform Resource Identifier
Es un identificador único usado por aplicaciones web.

ÍNDICE DE CONTENIDOS

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

INTRODUCCIÓN

En la actualidad la totalidad de las empresas (de pequeñas a grandes) tienen como objetivo principal mejorar sus productos o servicios; por una parte para lograr una mejor retribución económica directa como también para poseer la satisfacción y lealtad de sus clientes. Ya que los clientes son la carta fundamental para obtener los factores de mejora, es necesario escuchar sus opiniones tanto de cómo la empresa está entregando sus servicios, y cómo los aprecian en contraste con empresas de la competencia.

QServus, una empresa chilena que busca fomentar la calidad de servicio y la felicidad de las personas, detectó esta necesidad, por lo que desarrolló una herramienta que permite a cualquier empresa obtener feedback de sus clientes, a través de la creación de encuestas con variedad de tipos de preguntas y métodos de distribución, con métricas provenientes de información analizada en tiempo real y representadas en gráficos intuitivos para la mejor detección de los focos de acción.

Tras años ayudando a empresas en este objetivo, QServus detectó el potencial que conlleva analizar las preguntas de texto abierto, ya que permiten a los clientes detallar de mejor forma sus críticas y necesidades. La complicación recae en realizar el análisis de forma automática, ya que se presentan de forma no estructurada y no etiquetadas, lo que dificulta entregar información más allá de frecuencias o detección de palabras claves.

Considerando el panorama anterior, QServus ha decidido aprovechar esta oportunidad para construir la API de inteligencia artificial de la empresa, comenzando con análisis de sentimiento y la categorización de comentarios que, a opinión de sus clientes, es la necesidad más útil para ellos.

En este trabajo se abordarán estos desafíos a través de la construcción de una API que logre comunicarse con las aplicaciones de QServus, y que pueda ofrecer funcionalidades de análisis de texto como análisis de sentimiento y categorización de texto. Además de estas tareas, se debe investigar nuevos tipos de indicadores que logren mostrar esta información de forma clara e intuitiva.

En el presente documento, se entrega el desarrollo de este trabajo de título, organizado de la siguiente forma:

1. **Definición de problema:** Se entregan antecedentes históricos de la problemática antes de definirla. Se explica qué es QServus y cómo busca solucionar la problemática. Finalmente se presenta los objetivos de este trabajo.
2. **Marco conceptual:** Se definen todos los conceptos necesarios para tener una comprensión base respecto al trabajo realizado en la memoria.
3. **Propuesta de solución:** Se define el contexto de implementación de la solución para

posteriormente explicar cómo será realizado, considerando su diseño, construcción de modelos e indicadores y los criterios de aceptación para cada uno de ellos.

4. **Validación de la solución:** Se presentan los resultados de las métricas de los algoritmos de análisis de textos e indicadores. También, se muestra cómo es la implementación de la API y su correcto funcionamiento.

5. **Conclusiones:** Se detalla el aprendizaje obtenido al realizar el trabajo, el cumplimiento de los objetivos definidos en la fase inicial y, finalmente, se plantean los desafíos futuros para este trabajo.

CAPÍTULO 1
DEFINICIÓN DEL PROBLEMA

1.1. ANTECEDENTES DEL PROBLEMA

Durante la primera mitad del siglo XX dominó una economía basada en la industria, donde las empresas establecían sus estándares de calidad para construir sus productos, lo que desembocó en un sello de identidad que les representaba en el mundo. Sobre todo con la revolución industrial, la producción en masa hizo que los productos fueran más accesibles para los consumidores y así suplir la demanda existente. Con el pasar de los años esta tendencia fue mermando hasta los años 90', donde la era tecnológica comenzó a crecer fuertemente, propiciando un cambio de paradigma gigantesco en la economía y en cómo las empresas debieran producir.

Actualmente estamos en la era del consumidor la cual produjo cambios radicales. Los dos cambios más importantes son el enfoque de la producción hacia el consumidor, es decir, los intereses y gustos de estos influyen en lo que las empresas producen y no al revés, como sucedía años atrás. El segundo es la transición de producto a servicio, debido a que el alcance de consumidores aumentó considerablemente gracias a la tecnología, que nos permite alcanzar diversos rincones del planeta. Además, el sentido de servicio representa al consumidor una satisfacción de comodidad y atención personalizada.

Gracias a este cambio, la atención al cliente se ha vuelto un factor diferenciador fundamental en la industria, y allí el por qué las empresas buscan mejorar en ese aspecto. Una buena parte de esto es que los mismos consumidores son la fuente de retroalimentación principal, ya que son los que utilizan constantemente el servicio, logrando captar los problemas a mejorar; las partes del servicio con buen desempeño o cuáles son las funcionalidades que más impactan en los consumidores.

El problema que emerge es cómo las empresas pueden recopilar los datos principales para obtener la información que ayude a mejorar su producto o servicio. La cantidad de consumidores puede ser gigante, de diversos países, edades, intereses, necesidades, etc, lo que dificulta segmentar e identificar los focos donde las empresas deben dar hincapié. Se pueden realizar estudios o encuestas masivas lo cual puede ser muy costoso, sobre todo en pequeñas y medianas empresas. La empresa QServus identificó este problema y comenzó a realizar acciones al respecto para ayudar a otras empresas a solventar este inconveniente.

QServus es una empresa chilena que ofrece un servicio para la mejora de calidad de servicio de las empresas, utilizando encuestas de satisfacción. Se basan en un principio de tres pasos:

- Escuchar: Las encuestas son creadas y divulgadas. Además poseen un seguimiento de las respuestas en tiempo real.

- Analizar: Tras obntener los resultados de las encuestas, se analizan sus resultados para entregar métricas, las cuales son representados en gráficos o *indicadores*, que es el concepto que QServus las define.
- Actuar: Tras el análisis, la plataforma ofrece diversas herramientas para actuar, tales como contacto con correo electrónico, creación de bitácoras para hacer seguimiento caso a caso, etc.

Figura 1: Esquema de tres pasos de QServus.

Las encuestas son personalizables y presentan una variedad de tipos de preguntas, tales como numéricas, alternativas, texto abierto, etc, lo que propicia una gran cantidad de indicadores posibles para facilitar la lectura de las respuestas sin necesidad de realizar una revisión manual.

1.2. DEFINICIÓN DEL PROBLEMA

Las respuestas numéricas son fácilmente tratables mediante indicadores como histogramas, gráficos de torta, gráfico de cajas, etc, ya que son datos fijos e inclusive en la mayoría de los casos, con un rango predefinido. El problema surge con el análisis de las respuestas de texto abierto, las cuales son las que representan la información más importante.

Existen indicadores para el análisis de texto como nubes de palabras, extracción de conceptos importantes, categorización, resumen formato tabla, entre otros, pero su generación automática es bastante compleja debido a la extensión variable de respuestas, idioma utilizado, faltas de ortografía y el más importante, la identificación del contexto de las palabras lo que implica un significado distinto de una misma palabra en diferentes frases.

El fin de QServus es lograr dar información a tiempo real a las empresas para que puedan tomar acción en cualquier momento, por lo que la búsqueda de un modelo automatizado de análisis de texto es menester, ya que se podrá extraer toda esa información importante de las respuestas de texto abierto que apuntan a problemas cruciales en el servicio. La

implementación de lo anterior repercutirá inmediatamente en la rapidez de acción de las empresas para la mejora continua de su producto o servicio.

1.3. CONTEXTUALIZACIÓN DEL PROBLEMA

QServus actualmente no posee un mecanismo para realizar categorización de respuestas de texto abierto. Generalmente, estas respuestas van precedidas por una nota NPS que indica la satisfacción del cliente respecto al aspecto que se está evaluando. Gracias a esta nota, se puede inferir la intención de la respuesta, por lo que en la sección de *newsfeed* de la plataforma se presenta la respuesta de este cliente con una imagen con color y rostro, expresando si el cliente esta satisfecho o insatisfecho. Lo anterior es una buena forma de representar las respuestas de forma resumida, pero existe el problema donde las respuestas de texto abierto no parecen coincidir con la nota entregada, ya que a pesar de que se responda con una nota máxima, la respuesta de texto puede aún incluir alguna crítica. Por lo tanto, si la empresa quiere revisar los comentarios de una nota baja, se perderá esta crítica ya que posee una nota máxima.

1.4. OBJETIVOS DE LA MEMORIA

1.4.1. OBJETIVO GENERAL

Construir un software basado en inteligencia artificial que sea capaz de automatizar el procesamiento de texto abierto en las encuestas y ofrezca herramientas e indicadores, con el fin de mejorar la calidad de servicio o producto de cualquier empresa.

1.4.2. OBJETIVOS ESPECÍFICOS

- Desarrollo de modelos de redes neuronales para la predicción de sentimientos en las respuestas de texto abierto.
- Evaluar e implementar un modelo no supervisado para la categorización de texto.
- Creación de indicadores adecuados para la representación correcta de los datos procesados, considerando categorías y sentimientos.
- Creación de un software eficiente para el análisis y entrega de recursos, según las necesidades de la plataforma de QServus.

CAPÍTULO 2
MARCO CONCEPTUAL

En este capítulo revisaremos los conceptos fundamentales que sustentan este trabajo, tanto para su entendimiento general, como para el diseño, construcción y evaluación de la solución. Se definen conceptos básicos de procesamiento natural de texto, arquitectura de software y modelos de aprendizaje de máquinas con los cuales la solución será validada.

2.1. REST API

REST es un estilo de arquitectura de software que define un conjunto de restricciones para la creación de servicios web, de tal forma de potenciar y asegurar la interoperabilidad entre los sistemas a través de la web [Fielding, 2000]. Por otra parte, una API es una interfaz computacional que define las interacciones entre múltiples softwares. En conjunto, obtenemos un sistema que permitirá el manejo de recursos de forma estructurada y de fácil intercambio.

La colección de restricciones que una REST API debe cumplir para su correcto uso son:

- Arquitectura de cliente-servidor: Consiste en aislar los roles del cliente y servidor. Lo anterior permite que puedan evolucionar de forma independiente, además que esta separación permite que el servidor pueda adaptarse a diversas interfaces de usuario, lo que potencia la portabilidad.
- Ausencia de estados: El servidor no necesita almacenar información alguna sobre el estado de la sesión del usuario. Toda la información necesaria estará incluida en cada petición y no es necesaria información compartida entre peticiones diferentes. Esto potencia la simplicidad de las llamadas y que el estado de la sesión siga siendo manejada desde el lado del cliente.
- Manejo de caché: La REST API debe definir la presencia o ausencia de caché en las peticiones de diversos recursos, de tal forma de evitar datos erróneos o inapropiados tras una consecusión de peticiones.
- Sistema en capas: Implica que el cliente no necesariamente puede decir si la conexión al recurso se hace directamente al servidor o hay más intermediarios en la consulta. El servidor es capaz de realizar las peticiones correctamente sin importar si hay una capa superior de seguridad, un *proxy* o alguna otra entidad.
- Interfaz uniforme: Una restricción fundamental que ayuda a simplificar y desacoplar la arquitectura. Las tres restricciones de interfaz son:

- Identificación de recursos en consulta: Se debe identificar de forma clara en la petición el recurso, por ejemplo, en una URI.
- Manipulación de recursos: Cuando el cliente mantiene una representación del recurso, tiene información necesaria para modificar o eliminar el estado de este.
- Mensajes claros y explicativos: Cada petición a un recurso incluye un mensaje con la información necesaria para procesar la respuesta. Esto incluye, además del mensaje, un estado de respuesta HTTP, por ejemplo.

Para apoyar la restricción de *identificación de recursos en consulta*, se utiliza una URI, específicamente una URL. Este debe poseer el dominio con el cual será identificada la API seguido de un identificador del recurso. Para apoyar su simplicidad, las URIs deben poseer una jerarquía clara, ser únicas, los nombres de los recursos no deben ser verbos y no se deben incluir filtros de información en ellas.

La razón del porqué los recursos definidos en la URI no deben ser verbos, es debido a que la acción se define mediante los verbos que nos ofrece el protocolo HTTP, donde los más utilizados son:

- GET: Para consultar y leer recursos.
- POST: Para la creación de un recurso.
- PUT: Para la edición de recursos.
- DELETE: Para la eliminación de un recurso.
- PATCH: Para la edición parcial de un recurso.

Por último, al realizar una acción a un recurso específico se deben respetar ciertas condiciones para que esta acción sea aceptada o rechazada. Es importante que la REST API se asegure de cumplirlas y siempre debe notificar al cliente si la acción fue realizada correctamente y, en caso contrario, detallar la causa del error para que el cliente pueda corregir la consulta. Para este fin, se utilizan los códigos de estado que ofrece el protocolo HTTP, los cuales se dividen en 5 niveles y cada uno con un código que detalla e identifica el estado de la respuesta. Los niveles son:

- 100: Son estados que van informando al navegador los pasos o casos que debe ir considerando al enviar las peticiones.
- 200: Estos estados confirman la ejecución correcta en un recurso, detallando la acción que se realizó en el servidor.
- 300: Estados que indican al navegador que debe realizar acciones adicionales para la obtención del recurso.

- 400: Estados que indican un error por parte del cliente. Detallan el tipo de error para que el cliente pueda corregir la petición y realizarla nuevamente.
- 500: Estados que indican un error por parte del servidor. Son producidos cuando el servidor no puede procesar de forma correcta la petición a pesar de poseer una estructura aparentemente correcta.

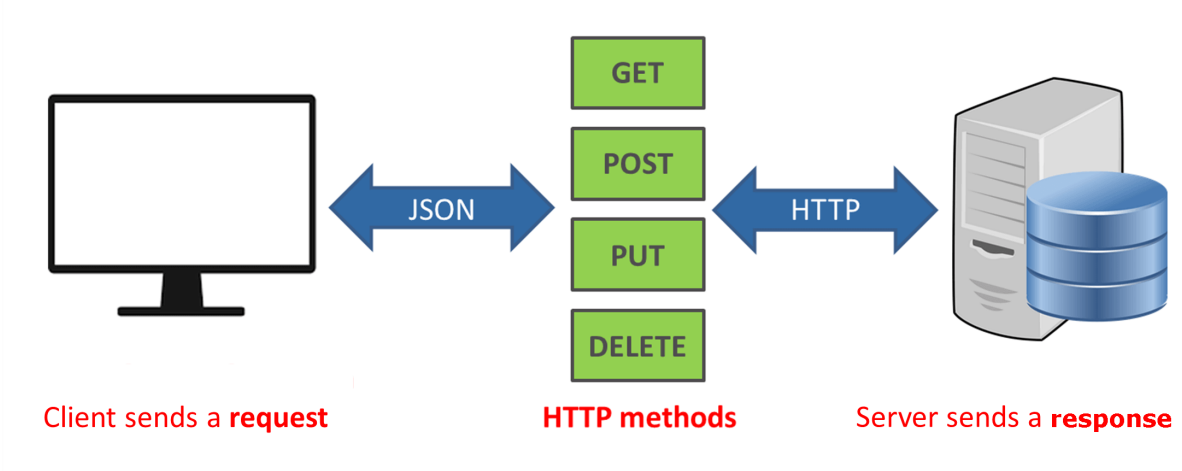


Figura 2: Esquema simple del funcionamiento de una REST API. Fuente: https://medium.com/faun/consuming-rest-apis-with-python-eb86c6b724c5

En conclusión, como se ve en la Figura 2, las REST API se basan principalmente en identificar sus recursos mediante una URI, luego se define la acción sobre este recurso gracias a los verbos HTTP y finalmente se retorna un código de estado al cliente para notificar el resultado de la petición, completando así el ciclo.

2.2. NET PROMOTER SCORE

El Net Promoter Score (NPS) es una herramienta que buscar medir la lealtad de los clientes de una empresa basado en las recomendaciones de éstos [Reichheld, 2003]. Consiste en una pregunta de tipo ”¿Cuán probable es que recomiende el producto o servicio a un familiar o amigo?” o ”¿Cómo fue el servicio/producto ofrecido por la empresa?”, donde se puede responder en un rango de 0 a 10, donde 0 implica ”Totalmente no lo recomiendo” y un 10 que es ”Lo recomiendo totalmente”. Según este resultado, los clientes son clasificados como:

- Promotores: Si entregan una nota 10 o 9
- Pasivos: Si entregan una nota 8 o 7

- Detractores: Si entregan una nota menor a 7

El cálculo del NPS consiste en considerar principalmente a los promotores y detractores con su proporción respecto al total de encuestados. Para obtener esta métrica se debe seguir la siguiente fórmula:

$$NPS = (P - D) * 100 \tag{1}$$

Esto representa un porcentaje que va desde un rango de -100 (todos son detractores) a 100 (todos son promotores).

Generalmente cuando se realizan las encuestas, uno quiere saber la opinión de la mayor cantidad de clientes posibles. Lo anterior no siempre se logra debido al gran trabajo que implica encuestar a todos los clientes, además de posibles consumidores que no contesten una encuesta entregada. Es por esto que el valor NPS puede no ser real si es que se calcula con una cantidad parcial de respuestas. Con la ayuda del error muestral, podemos definir qué tan real es el valor calculado del NPS para así poder usarlo con confianza para otros fines.

Para obtener el error muestral se requiere contar con la desviación estándar de las respuestas, la cual se calcula con la siguiente fórmula:

$$\hat{\sigma} = \sqrt{(1 - NPS_e)^2 * P + NPS_e^2 * N + (-1 - NPS_e)^2 * D} \tag{2}$$

donde N es la proporción de pasivos en las respuestas y NPS_e es el NPS estimado. Si no se quiere considerar los pasivos se debe quitar su parte en la ecuación. Con la desviación estándar muestral calculada, podemos utilizar su valor para calcular el error muestral como se muestra en la siguiente fórmula:

$$\mathrm{z}_{\frac{\alpha}{2}} * \frac{\hat{\sigma}}{\sqrt{n}} \tag{3}$$

Donde n es la cantidad de respuestas y $\mathrm{z}_{\frac{\alpha}{2}}$ es la desviación estándar de una distribución normal estándar relacionada al grado de confiabilidad α. Si α = 0.1, tendremos un grado de confiabilidad del 95 %, por lo que el valor real del NPS se encontrará dentro del rango:

$$[NPS_e - \mathrm{z}_{\frac{\alpha}{2}} * \frac{\hat{\sigma}}{\sqrt{n}}, NPS_e + \mathrm{z}_{\frac{\alpha}{2}} * \frac{\hat{\sigma}}{\sqrt{n}}] \tag{4}$$

En términos generales, una diferencia del 5 % entre el error muestral y real se considera suficiente, pero mientras menor sea, mejor. Al obtener valores de NPS más cercanos al real, podemos utilizarlos sin temor a que un gran error muestral se propague a los modelos y las etiquetas de los datos de entrenamiento.

2.3. PROCESAMIENTO DE LENGUAJE NATURAL

El procesamiento de lenguaje natural es una gama de técnicas computacionales para analizar y representar ocurrencias naturales de texto en uno o más niveles de análisis lingüísticos, con el propósito de conseguir un procesamiento de lenguaje humano en diversas tareas o aplicaciones [Liddy, 2001]. ¿Qué aspectos alberga esta definición?:

La gama de técnicas computacionales puede ser una definición imprecisa, pero es necesaria debido a la gran cantidad de técnicas para abarcar un solo problema o varios de ellos dentro del campo del lenguaje.

Ocurrencias naturales de texto ya que no necesariamente tiene que ser escrito; puede ser hablado, de diversos idiomas o dialectos pues el lenguaje a través de la historia evoluciona enormemente.

Niveles de análisis lingüísticos ya que obteniendo control de estos niveles, el sistema de NLP podrá obtener un mejor desempeño en el lenguaje. Estos niveles se componen de:

- Fonología: Se centra en la interpretación de los sonidos entre y a través de las palabras. Dentro de esta se encuentran tres reglas:
  - Fonética: En los sonidos entre palabras.
  - Fonémica: En las variaciones de pronunciaciones cuando las palabras son habladas juntas.
  - Prosódica: Por la fluctuación y acentuación en la entonación a través de una sentencia.

  Este nivel es importante cuando se quiere hacer una transcripción de voz a texto ya que en un mismo lenguaje hay diversos acentos que poseen reglas diferentes.
- Léxica: En este nivel se interpreta el significado de las palabras por si solas, es decir, de diccionario. Un ejemplo, se analiza el significado de la palabra *'merienda'*, que significa un alimento ligero. En este nivel surge una herramienta denominada *'part-of-speech'*, o *'categoría gramatical'* en español, que consiste en categorizar cada palabra en su tipo o rol, el cual puede ser artículo, pronombre, verbo, etc.
- Morfológica: Representa la composición natural de las palabras, las cuales se componen de morfemas, la unidad más pequeña de entendimiento. Un ejemplo es la palabra alimentar. Este es un verbo simple, pero gracias a ciertos morfemas podemos cambiar el sentido de la palabra. Si usamos el morfema *-ción*, obtenemos alimentación; el morfema *-se* obtenemos alimentarse y, con el morfema *-aba*, alimentaba. Cada morfema puede incluir nueva información a un mismo concepto, como tiempos verbales o incluso cambiar su rol, como pasar de un verbo a un sustantivo.

- Sintáctica: Se enfoca en analizar las palabras en una sentencia para descubrir la estructura gramatical de una sentencia. Es muy importante ya que sentencias con las mismas palabras pero con diferente orden pueden dar un significado diferente, como el ejemplo de ”El perro cazó al gato” y ”El gato cazó al perro”, los cuales solo presentan una diferencia sintáctica, pero una causalidad muy diferente.
- Semántica: Determina los posibles significados de una sentencia enfocándose en las interacciones entre los significados de cada palabra que la componen. Esto toma un rol primordial en las desambiguaciones lingüísticas y en las polisemias; palabras que toman un significado distinto dependiendo de las palabras que la acompañan en la oración. Un ejemplo es la palabra sierra, que puede ser una herramienta para cortar madera, un conjunto de montañas alineadas o una especie de pescado.
- Pragmático: Se centra en la utilidad del lenguaje en diversas situaciones y cómo usa el contexto por sobre el contenido del texto mismo para dar un nuevo entendimiento. Permite dar información adicional sin estar explícitamente dentro de la sentencia. Un ejemplo, en una conversación cuando se está hablando sobre un profesor, decir su nombre la primera vez es suficiente, y da el contexto al resto de la conversación. Con esto, podemos usar el artículo *él* para referirse al profesor sin tener que repetir su nombre cada vez.

Procesamiento de lenguaje humano o natural (NLP por sus siglas en inlgés) busca que las máquinas representen el idioma con el cual nos comunicamos. Con esta definición (aunque NLP considere otras disciplinas igualmente) se engloba este campo dentro de la inteligencia artificial.

En diversas tareas o aplicaciones implica que hay diversas tareas específicas que consideran el lenguaje humano como herramienta como, por ejemplo, análisis de sentimiento, traducción de idiomas, transcripción de voz a texto, etc.

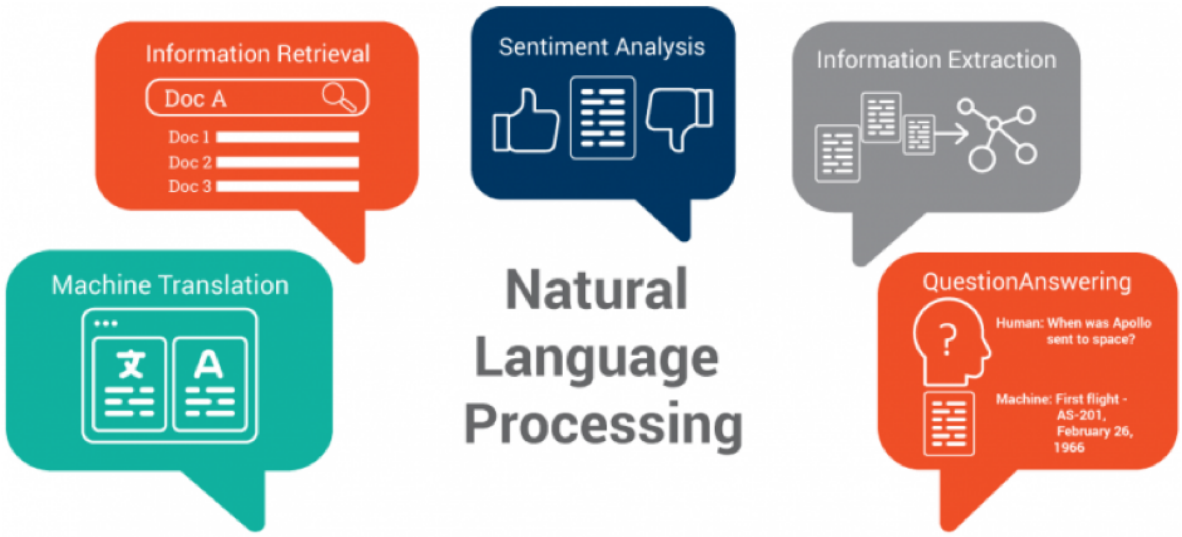


Figura 3: Diversas aplicaciones del procesamiento de lenguaje natural. Fuente: [https://medium.com/greyatom/introduction-to-natural-language-processing-78baac3c602b]

2.3.1. REPRESENTACIÓN DE TEXTO

Un computador no puede comprender de forma directa el texto escrito ya que este funciona bajo el sistema numérico binario. Debido a esta necesidad se utilizan representaciones de texto para que los programas hagan una relación numérica a cada palabra y así poder trabajar con ellas. Se utilizan variadas técnicas de preprocesamiento de texto [Jurafsky y Martin, 2019], dentro de las cuales las más destacadas son:

- Normalización: Se eliminan caracteres o elementos del texto que no ofrecen información importante para el análisis. Estos caracteres pueden ser puntos, signos de exclamación o de pregunta, llaves, corchetes, etc. Además, se lleva todo el texto a minúsculas, ya que para la máquina las palabras Comida y comida son distintas pues tienen una codificación interna diferente..
- Tokenización: Un string, que es un conjunto de caracteres, es considerado como si fuera una sola palabra para la máquina. Es importante lograr dividirlos para obtener una cadena de palabras, para que así la computadora comprenda que una sentencia está compuesta por una o varias de ellas.
- Definición y eliminación de stopwords: Las stopwords son un conjunto de palabras que poseen una gran frecuencia en los textos, pero son utilizadas principalmente para una definición sintáctica del lenguaje. Generalmente se trata de artículos, conectores o pronombres.

- Lematización y Stemming: Son dos técnicas que se enfocan principalmente en obtener un prefijo raíz de las palabras con el fin de reducir el vocabulario total de los documentos, lo que ayuda a simplificar la complejidad. Stemming busca truncar las palabras de forma más básica, mientras que la lematización considera el contexto del lenguaje y la sentencia, haciendo que una misma palabra termine en un distinto lema dependeniendo de su función en la sentencia.

Las técnicas anteriores son utilizadas para obtener el vocabulario más reducido posible, manteniendo la información que otorga el corpus. Con esto, cuando se realice la representación de las palabras, se logre una expresión menos compleja lo que implica realizar menos comparaciones en los cálculos, búsquedas más simples, etc. Las dos representaciones de texto más utilizadas son:

- Bolsa de palabras: Consiste en definir un vector inicializado en 0 y de largo total igual a la cantidad de palabras en el vocabulario. Tomando una sentencia se hace un conteo de cada palabra y se le asigna ese valor en el vector según corresponda. Este modelo funciona bien en ciertas cincurstancias, pero cuando el vocabulario es muy extenso tendremos un vector base del mismo tamaño, y generalmente las sentencias poseen pocas palabras en comparación al vocabulario. Lo anterior produce el efecto *sparse*, es decir, que hay una gran cantidad de valores 0 en los vectores, haciendo que cualquier operación entre ellas produzca pequeños cambios, complicando analizar su real impacto.
- Word embedding: Consiste en una representación de un vector con valores reales en donde se define un vector de un tamaño específico y que ciertos valores fijos de esos vectores representan directamente a una palabra. En resumen, en vez de tener una representación vectorial por una sentencia como en la bolsa de palabras, ahora se posee una representación vectorial por cada palabra del vocabulario. Esto es de mucha ayuda, ya que se pueden aprovechar las propiedades de los vectores tales como suma, resta, producto y distancias para definir las similitudes o diferencias entre las palabras.

2.3.2. TF-IDF

Esta métrica, que es el resultado de dos estadísticos, busca reflejar qué tan importantes son las palabras dentro y a través de los documentos de un *corpus*. Se entiende por corpus como un conjunto de textos extensos y estructurados.

TF es el nombre general para la métrica de *frecuencia de un término*. Esta puede ser calculada con diferentes fórmulas, pero su esencia es la misma; considerar la cantidad de veces que aparece una palabra dentro de un documento. Generalmente, la fórmula es simple, asociada al conteo de palabras, siendo del tipo:

$$tf(t,d) = f_{t,d} \tag{5}$$

Donde *t* representa un término o palabra y *d* representa el documento. La función $f_{t,d}$ representa la función de conteo habitual: ¿cuántas veces aparece el término *t* en el documento *d*?.

La función de frecuencia de términos puede utilizar otras fórmulas para determinar su peso, como por ejemplo:

- Frecuencia de término ajustado al documento: Se utiliza para dar importancia al largo del documento. En este caso será mucho más importante un término más frecuente en un documento más corto, ya que será un concepto que lo definirá mejor.

$$tf(t,d) = \frac{f_{t,d}}{n_d} \tag{6}$$

- Frecuencia con escala logarítmica: Utilizada principalmente cuando se obtienen rangos de frecuencias muy grandes y se quiere reducir esa brecha haciéndola más compacta.

$$tf(t,d) = log(1 + f_{t,d}) \tag{7}$$

- Frecuencia aumentada: Es utilizada para prevenir el sesgo presente con documentos muy grandes. En este caso, la frecuencia de cada término es dividida por la frecuencia del término más frecuente en el documento.

$$tf(t,d) = 0{,}5 + 0{,}5 * \frac{f_{t,d}}{max\{f_{t',d} : t' \in d\}} \tag{8}$$

idf es el acrónimo de *frecuencia de documento inversa* en inglés y busca detectar cuánta información la palabra o término provee a través de los documentos y en cada uno de ellos. Gracias a este concepto es podemos analizar la información cruzada en los documentos y detectar si las palabras son frecuentes porque dan un indicio respecto al tópico del documento, o solamente es un término muy frecuente en el lenguaje (como los conectores o artículos personales, por ejemplo). La fórmula para calcular el IDF es la siguiente:

$$idf(t,D) = log\left(\frac{N}{1 + |\{d \in D : t \in d\}|}\right) \tag{9}$$

en donde *N* es la cantidad de documentos en el corpus *(N = |D|)* y el denominador corresponde al número de documentos en donde el término *t* aparece. Por lo tanto la especificidad de un término puede ser cuantificado como la función inversa del número de documentos en el cual este término aparece [Sparck, 1972].

Con la definición de ambos conceptos se puede utilizar el estadístico TF-IDF, el cual simplemente se define como:

$$tfidf(t, d, D) = tf(t, d) \cdot idf(t, D) \tag{10}$$

Debido a que *idf* es una función logarítmica, tiene una cota inferior 1, la función *tdidf* será siempre mayor o igual a 0. Para obtener valores altos de *tfidf* la frecuencia de un término debe ser alta y a su vez menor a través de los documentos. En resumen, mientras menos se repita el término entre documentos, más aumenta el valor de *tfidf*.

2.4. DEEP LEARNING

Las redes neuronales artificiales se remontan a la década del 40´ asociadas al Aprendizaje de Máquinas (Machine Learning), como modelos o algoritmos que surgen de analizar los procesos biológicos que permiten a los seres humanos comprender su entorno e ir aprendiendo a través de la vida. Las redes neuronales son entonces un conjunto de algoritmos diseñados especialmente para reconocer patrones de comportamiento, emulando a las neuronas de nuestro cerebro. Recién en los últimos años han adquirido mayor popularidad, gracias a que han disminuido las exigencias de su complejidad computacional y la cantidad de datos necesarios para obtener resultados satisfactorios.

A finales de los 80’ y durante la década del 90’ se realizaron varios descubrimientos importantes, como la implementación satisfactoria del algoritmo de backpropagation para el aprendizaje [Rumelhart *et al.*, 1986] y la creación de las unidades LSTM [Hochreiter y Schmidhuber, 1997]

No fue hasta el año 2006 donde comenzó a explotar la popularidad y el uso de las redes neuronales debido a tres factores principales: el descubrimiento de nuevos y mejores modelos de redes neuronales, la rápida y potente renovación del hardware y, por último, la digitalización de la información que permite conseguir una gran cantidad de información en poco tiempo.

2.4.1. REDES NEURONALES FEEDFORWARD

Una red neuronal es un modelo matemático y estadístico inspirado en las redes neuronales biológicas presentes en los cerebros de diversos animales tanto en la estructura como en el proceso de aprendizaje vinculado.

Las redes neuronales se componen de tres partes principales [Goodfellow *et al.*, 2016]:

- Neurona: Es una estructura la cual puede intercambiar información con neuronas vecinas
- Conexiones: Es la relación entre dos neuronas que permite intercambiar la información propagada
- Capas: Es un conjunto de neuronas las cuales generalmente poseen características similares

La red neuronal feedforward se compone de capas, principalmente de tres tipos:

- Capa de entrada: Consiste en la capa que hará entrar los datos con la cual el modelo entrenará y predicirá los resultados.
- Capa escondida: Es o son la/s capa/s que se encargan de extraer las capas de abstracción de los datos. Cada neurona de la capa posee un peso que será traspasado a los valores propagados desde la capa anterior y mediante una función de activación es procesado para entregarlo a las neuronas de la capa siguiente.
- Capa de salida: Es la capa donde la red muestra el resultado siendo ésta un dato numérico o una clasificación, según sea la definición de la arquitectura de la red.

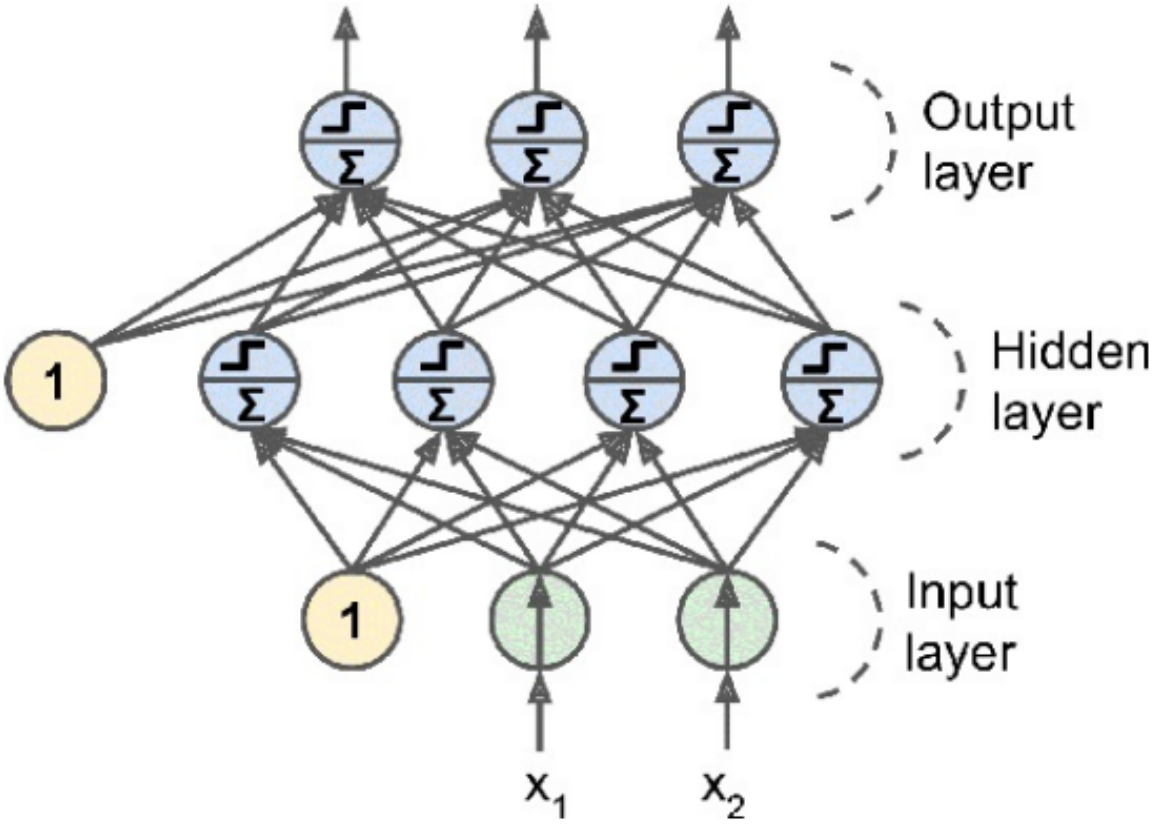


Figura 4: Representación gráfica de una red neuronal feedforward. Fuente: [Géron, 2019]

En la Figura 4 se aprecia la representación gráfica de una red neuronal de dos dimensiones, es decir, un vector $X = (x_1, x_2)$ y la cual posee tres salidas como se aprecia en la capa de salida. Cada neurona realiza una operación dependiendo de su función de activación σ y de los datos que van entrando. En notación:

$$\sigma(WX - b) \tag{11}$$

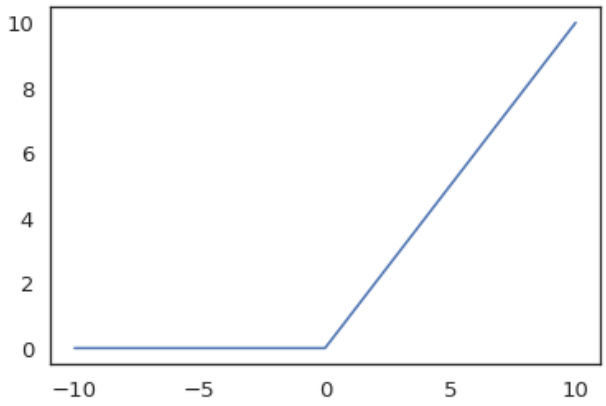
Donde W es la matriz de pesos correspondiente a la neurona, X es una segunda matriz de datos x_i que ingresan a la neurona. Este resultado es restado por un bias b, que es un valor único por capa y que no es influenciado por lo valores de la capa anterior.

Gracias a la flexibilidad que permite la construcción de una red neuronal pueden la diversas capas poseer distintas funciones de activación, las cuales dependerán de la finalidad del entrenamiento. Las funciones de activación más conocidas y utilizadas son:

- ReLU: Función de activación que representa la parte positiva de su argumento. Utilizada principalmente por su simplicidad de cómputo y su buen desempeño durante el

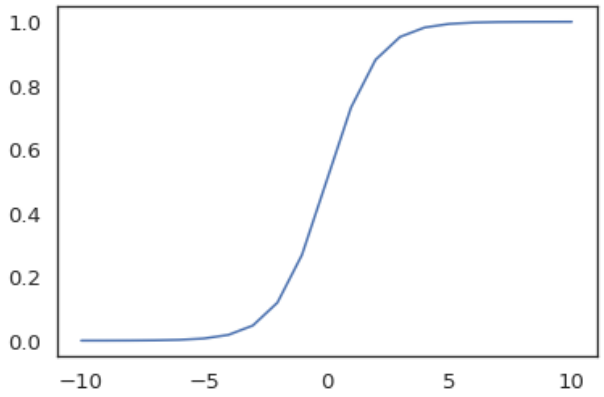
back-propagation.

$$f(x) = x^+ = max(0, x) \tag{12}$$



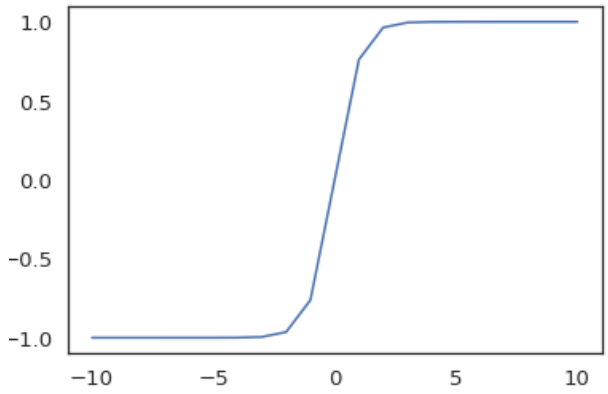
- Sigmoid: Función que representa una forma de S. Su salida está acotada entre 0 y 1.

$$\sigma(x) = \frac{e^x}{1 + e^x} \tag{13}$$



- Tanh: Función de tangente hiperbólica. Se comporta de una forma similar a la sigmoid, pero su salida está acotada entre (-1, 1), por lo que acepta valores negativos en su salida.

$$tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}} \tag{14}$$



2.4.2. REDES NEURONALES RECURRENTES

Las redes neuronales recurrentes [Rumelhart *et al.*, 1986] son un tipo de redes neuronales para procesar información secuencial. A diferencia de la red neuronal feedforward que procesa la información en una sola dirección (desde la capa de entrada hasta la salida) la red neuronal recurrente incluye conexiones a las neuronas que apuntan a si mismas. Gracias a lo anterior, las neuronas adoptan el concepto de *estado*, el cual les permite generar la secuencia y poseer una memoria que se hereda en cada iteración.

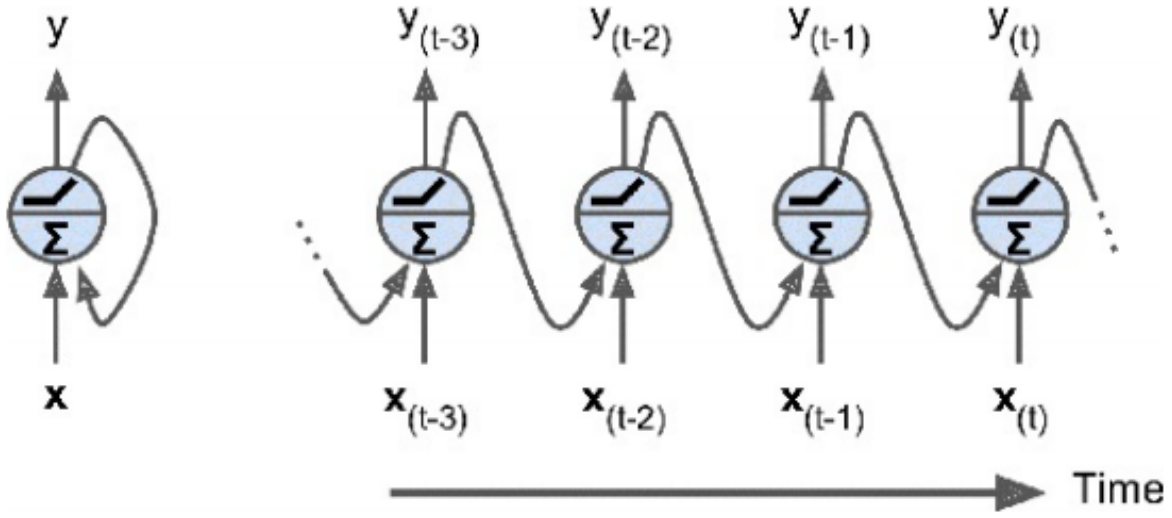


Figura 5: Representación gráfica de una red neuronal recurrente. Fuente: [Géron, 2019]

En la Figura 5, se puede apreciar la ya conocida orientación lineal desde la entrada y la salida, que en la imagen va desde abajo hacia arriba. Es posible además visualizar la conexión hacia sí mismo de la neurona, la cual podemos desenvolver para apreciar los diversos estados que va adquiriendo.

Formalizando, se considera la forma clásica de un sistema dinámico:

$$s^{(t)} = f(s^{(t-1)}, x^{(t)}; \theta) \tag{15}$$

En donde $s^{(t)}$ es el estado del sistema en el momento *t* y $x(t)$ es una función de señal externa. Se define recurrente ya que el estado *t* está siendo definido con el resultado en el estado anterior, es decir, *t-1*. El proceso de desenvolver consiste en ir mostrando los estados que van siendo traspasados a iteraciones posteriores. Si consideramos un sistema dinámico de tres estados, se puede definir como:

$$s^{(3)} = f(s^{(2)}, x^{(3)}; \theta) = f(f(s^{(1)}, x^{(2)}; \theta), x^{(3)}; \theta) \tag{16}$$

A este sistema dinámico se le puede agregar también una función de señal externa. Al momento de definir la red neuronal recurrente se representa la señal externa como los datos de entrada en el modelo. Considerando esto, se define la arquitectura formal como:

$$h^{(t)} = \sigma(W_h x^{(t)} + U_h h^{(t-1)} + b_h) \tag{17}$$

$$y^{(t)} = \sigma(W_y h^{(t)} + b_y) \tag{18}$$

En donde $h^{(t)}$ es la definición de las capas ocultas e $y^{(t)}$ la definición de la capa de salida. Además:

- $x^{(t)}$ es el vector de datos de entrada.
- $h^{(t)}$ es el vector de salida de una capa oculta.
- b_h y b_y son los bias de las capas.
- W y U son la matriz de pesos, donde W representa los pesos de los datos de entrada y U representa a los datos que son dirigidos a sí mismo en las capas.

Revisando la formalización anterior, se detectan dos flujos, uno siendo el convencional que es ajustado con los pesos W y otro recurrente que almacena los estados de cada neurona, que son ajustados por U.

2.4.3. LONG-SHORT TERM MEMORY

En [Hochreiter y Schmidhuber, 1997] definen una nueva arquitectura para la neurona de la red neuronal recurrente, renombrándola *memoria*. La búsqueda de esta nueva arquitectura nace por el complejo proceso de BPTT que tendía a gradientes explosivos y desvaneciente debido a que el error de este proceso dependía exponencialmente de la magnitud de los

pesos del modelo. El beneficio clave de esta nueva arquitectura no es solo que solventa este problema con el BPTT, si no que también permite que los modelos puedan considerar una gran cantidad de estados sin desestabilizar el aprendizaje. Posteriormente [Gers *et al.*, 2000] se define que los pesos de la conexión hacia sí mismo no sean fijos, si no condicionados según el contexto del estado.

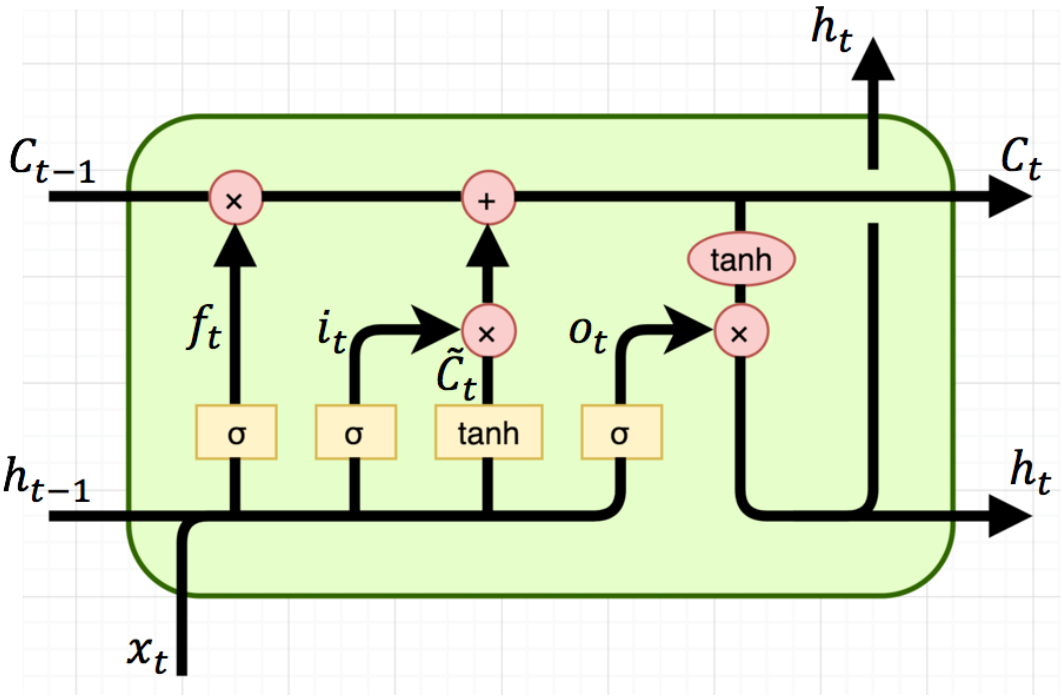


Figura 6: Radiografía de una LSTM. Fuente: [https://medium.com/@saurabh.rathor092/simple-rnn-vs-gru-vs-lstm-difference-lies-in-more-flexible-control-5f33e07b1e57]

En la Figura 6 se puede apreciar la memoria LSTM que posee los mismos parámetros que una red neuronales recurrente pero son traspasados por diversos componentes, los cuales son:

- Forget Gate: Es una compuerta que se encarga de revisar qué información será desechada del estado de la memoria. Esta compuerta es la primera sigmoide de la izquierda, y su comportamiento se puede formalizar como sigue:

$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f) \tag{19}$$

- Input Gate Layer: Es una compuerta que va a ingresar la información a la celda en el nuevo estado. Se define por dos funciones de activación, una sigmoidal y otra tangente hiperbólica. Formalizando, obtenemos:

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) \tag{20}$$

$$\tilde{C}_t = tanh(W_C \cdot [h_{t-1}, x_t] + b_C) \tag{21}$$

Luego, cuando cada capa realiza su operación, se combinan para que sea agregado al estado de la memoria.

- State layer: Tras obtener el resultado de las capas anteriores, el estado se calcula tomando el estado anterior C_{t-1} multiplicándolo por la forget layer f_t y luego sumándolo con la input gate i_t.

$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t \tag{22}$$

- Output layer: El valor de salida de la memoria será el mismo estado generado en la memoria pero filtrado con funciones de activación. Primero se procesa el output tomando los parámetros de datos de entrada y el output del estado anterior y usándolo en una sigmoidal y luego usando una tangente hiperbólica entre lo anterior y el C_t. Así, se hace un balance y se define qué partes del estado anterior continuarán y cuántas se mantendrán.

$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o) \tag{23}$$

$$h_t = o_t * tanh(C_t) \tag{24}$$

2.4.4. GATED RECURRENT UNIT

En esta arquitectura propuesta por [Cho *et al.*, 2014], se define una nueva unidad muy parecida a la LSTM pero con ciertas variaciones, conocida como GRU. La razón de esta nueva arquitectura es para lograr resolver el problema del gradiente desvaneciente y, al ser un poco más simple que la LSTM, ofrece tiempos de cómputo menores obteniendo igualmente buenos resultados.

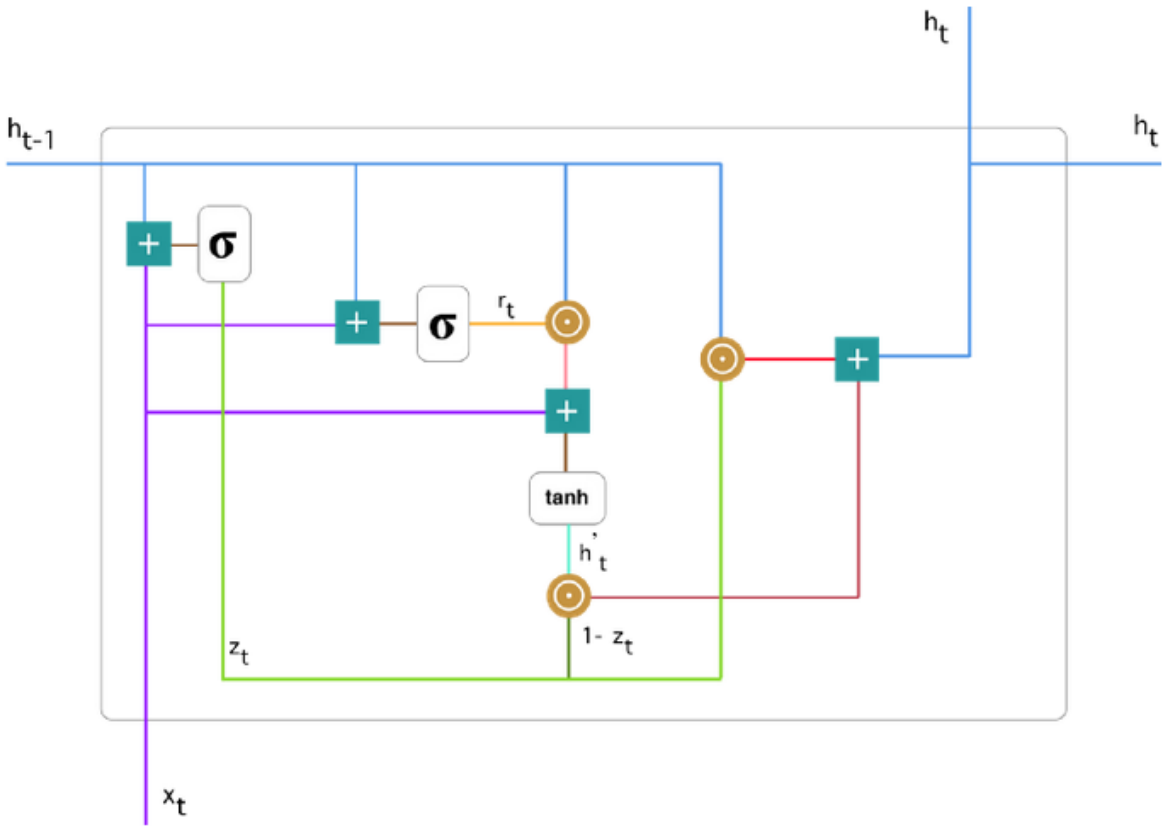


Figura 7: Radiografía de una GRU. Fuente: [https://towardsdatascience.com/understanding-gru-networks-2ef37df6c9be]

En la Figura 7 se aprecia la estructura de una unidad GRU, la cual se compone de las siguientes partes:

- Update gate: Es la compuerta de actualización y se encarga de decidir cuánta información del tiempo anterior debe mantenerse en el actual. El resultado se obtiene con:

$$z_t = \sigma(W^{(z)}x_t + U^{(z)}h_{t-1}) \tag{25}$$

- Forget gate: Es la compuerta que se encarga de olvidar la información del tiempo anterior según estime conveniente. La fórmula es la siguiente:

$$r_t = \sigma(W^{(r)}x_t + U^{(r)}h_{t-1}) \tag{26}$$

  A simple vista es la misma función que la compuerta de actualización, pero la diferencia recae en cómo sus resultados serán usados posteriormente.

- Current Content: Consiste en la información que representará a la unidad actual, y que es obtenida al realizar operaciones con la *forget gate*, los datos de entrada y el estado del tiempo anterior.

$$h_t^{'} = tanh(Wx_t + r_t \odot Uh_{t-1}) \tag{27}$$

  En donde el símbolo $\odot$ corresponde al producto de Hadamard.

- Final memory: Consiste en la salida de la unidad GRU, en donde opera con la compuerta de actualización, el estado del tiempo anterior y de la información de la unidad actual.

$$h_t = z_t \odot h_{t-1} + (1 - z_t) \odot h_t^{'} \tag{28}$$

CAPÍTULO 3
PROPUESTA DE SOLUCIÓN

En esta sección se detallan los pasos de análisis y de diseño de solución la cual será implementada para resolver el problema estipulado. Se detallarán las restricciones en consideración para desarrollar la solución, las tecnologías utilizadas y diseño formal de la solución. También se estipulará los modelos de aprendizaje automático que atacarán el problema de categorización de texto y análisis de sentimiento y cómo esta información será entregada para su representación. Por último, se definirán indicadores que ayudarán a mostrar de forma más intuitiva la información, de modo que los clientes puedan tomar decisiones fácilmente.

3.1. PREPROCESAMIENTO DE TEXTO

El preprocesamiento de texto es una tarea importante y compleja, ya que busca corregir errores de texto a la vez que se simplifica el vocabulario. Gracias a lo último, permite que los modelos puedan trabajar con datos de mejor calidad y ofrecer mejores resultados. Primero se deben identificar las principales causas de los errores en los textos y qué técnicas se utilizarán para solucionarlos. Cabe destacar que los pasos que se definan en el preprocesamiento de texto serán los mismos para el análisis de sentimiento y categorización de texto.

3.1.1. CORRECCIONES ORTOGRÁFICAS

Desde su base, las respuestas de las encuestas no siempre vendrán bien escritas. Esto dificulta el análisis por varias razones:

1. Palabras mal escritas conllevan a un análisis erróneo por parte de los modelos, ya que éstos consideran todas las palabras que encuentren en el conjunto.
2. Un modelo necesita un vocabulario de las palabras que existen en el conjunto de respuestas. Incluir palabras mal escritas aumenta el vocabulario, haciendo que el modelo trabaje en un contexto más complejo.
3. Como los modelos analizan el contexto de las frases, palabras mal escritas pueden quitarle información a su símil bien escrito, haciendo que el modelo esté más inseguro sobre cuál de las dos palabras impacta más en la decisión.

Considerando lo anterior, es importante identificar los siguientes tipos de errores ortográficos:

1. **Tildes:** La presencia o ausencia de tildes en palabras donde no corresponde. Generalmente el error más común es la ausencia de tildes, y es importante que las palabras la posean correctamente, ya que el proceso de *lematización* lo exije.
2. **Error ortográfico:** Producido por no saber cómo escribir bien una palabra (e.g. escribir conyeva en vez de conlleva) o por un error de tipeo en letras que están muy cerca en el teclado (escribir varco en vez de barco ya que las letras v y b están muy cerca).
3. **Palabras no espaciadas:** En ocasiones no se utiliza el carácter de espacio en blanco para separar palabras, produciendo ejemplos como *laresponsabilidad* o *enconclusión*. Esto dificulta al proceso de tokenización, ya que se utilizan los espacios en blanco en las sentencias para obtener las palabras.

Los tres casos explicados anteriormente serán considerados en el preprocesamiento de ambos algoritmos, con el fin de obtener un conjunto de datos de buena calidad.

Para solucionar el problema anterior, se definen dos agentes:

1. **Guardián:** Este agente se encarga de revisar todas las palabras que ingresen al análisis. Si la palabra no está bien escrita, la asigna a revisión con el segundo agente. En caso contrario, pasa a la siguiente etapa del análisis.
2. **Corrector:** Este agente detecta las palabras mal escritas y establece candidatos de palabras. Dentro de ellos, ve que tanto se parecen a la palabra escrita. Su criterio se basa en el contexto de la palabra (considerando la frase de donde se sacó) y en qué cantidad de letras se difiere cada candidato con la palabra mal escrita.

El agente corrector intentará asignar el candidato correcto a la palabra mal escrita. Pero en el caso donde la palabra es irreconocible y no existan candidatos potenciales, es desechada. Esto implica una nueva revisión de la frase, ya que en frases muy pequeñas, desechar la palabra mal escrita implica que la frase quede vacía o sin sentido.

Por último, se realiza una eliminación de ciertos carácteres especiales que están dentro de las sentencias como signos de exclamación, pregunta, puntos, slash, ampersand, etc.

3.1.2. STOPWORDS

Se define *stopwords* como un conjunto de palabras que son previamente eliminadas de los datos de texto. La razón de su eliminación recae en que son palabras muy frecuentes y no presentan información útil para la clasificación. Es más, la función de estas palabras es para que las sentencias sean coherentes y con correcto sintaxis. Debido a que se requiere disminuir el vocabulario para simplificar el análisis de los modelos, son eliminados artículos personales, conectores, etc.

Figura 8: Ejemplos de palabras que son consideradas stopwords.

3.1.3. LEMATIZACIÓN Y STEMMING

Ambos procesos consisten en transformar una palabra a su forma raíz, ocupando diferentes criterios para tal fin. El stemming consiste en transformar una palabra en su *stema*, que utiliza una heurística para cortar la última parte de una palabra y así obtener un *prefijo*. En cambio la lematización transforma en un *lema*, realiza este proceso considerando análisis morfológico y el vocabulario. Con lo anterior, se obtiene la forma básica, en el diccionario, de la palabra [Manning *et al.*, 2008].

En la Tabla 1, se presenta una comparación entre ambas técnicas, utilizando la misma palabra.

Criterio	Stemming	Lematización
Palabra raíz	Corta la palabra para obtener un prefijo raíz	Convierte la palabra en su raíz semántica
Valor raíz	El stema puede no tener significado en el español	El lema siempre tiene un significado en español
Conversión contextual	El stema considera una palabra individual y la acorta	Un lema es obtenido según su rol en la sentencia.
Output con palabra "$trabajando$"	Trabaj	Trabajar

Tabla 1: Tabla comparativa entre ambas técnicas de obtención de palabras raíz.

El stem es una solución más sencilla, fácil y que promueve un vocabulario más corto en comparación con el lema. La desventaja es que la mayoría de palabras raíces pierden significado en el lenguaje (como se aprecia en la tabla 1 al stemizar la palabra trabajando).

Por otra parte, el lema es una herramienta más compleja de implementar pero tiene como ventaja que el lema siempre tendrá un significado en el español, además que su raíz será analizada según su contexto. Por ejemplo, en las sentencias ”el trabajo está aumentando mucho esta semana” y ”yo trabajo en una empresa costera”, obtendríamos el lema *trabajo* y *trabajar* respectivamente. Lo anterior es debido al rol de la palabra *trabajo* en cada frase; en el primero actúa como sustantivo y en el segundo como un verbo.

Se utilizarán ambas técnicas en los procedimiento según sea conveniente. Nos importa que todas las palabras posean significados, pero también queremos obtener un vocabulario minimalista para facilitar el entrenamiento y la búsqueda para los algoritmos.

3.2. ANÁLISIS DE SENTIMIENTO

El análisis de sentimiento consiste en realizar análisis de texto de un conjunto de respuestas y extraer sentimientos de ellas. Las etiquetas que generalmente se usan en el análisis de sentimiento son positivas, neutras, mixtas y negativas. En este trabajo se utilizarán tres de ellas: positivas, negativas y mixtas. Las neutras fueron descartadas ya que no representan información útil en el contexto de encuestas de calidad de servicio, ya que no se enfocan en un problema o beneficio en particular. Se utilizará un modelo de red neuronal recurrente para definir el sentimiento de los comentarios.

Para definir de forma correcta la solución, se debe definir cómo obtener las etiquetas de los datos iniciales para el entrenamiento, las respuestas de encuestas que se usarán, la arquitectura de red neuronal recurrente que se utilizará para validar y, finalmente, el proceso que se llevará a cabo para lograr cumplir con los criterios de aceptación.

3.2.1. NPS COMO ETIQUETA DE DATOS

Los modelos de redes neuronales son supervisados, es decir, necesitan que los datos vengan previamente etiquetados según el sentimiento que poseen. Los clientes en QServus no han realizado encuestas en las cuales se pueda relacionar una respuesta de texto con sentimiento, tampoco poseen alguna otra fuente de datos similar. Una solución sería establecer reglas del lenguaje para definir qué adjetivos son buenos o malos, pero implicaría almacenar un diccionario que crecerá con el tiempo y que dependerá del rubro de la empresa a analizar.

A pesar de lo anterior, existe una solución donde es posible categorizar comentarios de texto de diversas encuestas si cumplen con cierta estructura. Muchas encuestas de QServus creadas por las empresas, establecen un diseño en donde se solicita al encuestado poner una nota en general o para cierto aspecto específico sobre el producto y/o servicio. Tras eso, se presenta otra pregunta de texto abierto para justificar la elección de dicha nota.

Las preguntas de escala en donde se consulta la satisfacción general o específica sobre un

producto o servicio se denomina NPS. Generalmente, las notas van en escala de 0 a 10, pero es posible utilizar las otras escalas de notas que QServus ofrece. Gracias a este diseño de encuestas, podemos tomar una respuesta de texto y transformar su nota a una etiqueta NPS.

Etiqueta NPS	Rango de nota	Etiqueta procesada
Promotor	[9, 10]	Positivo
Pasivo	[7, 8]	Mixto
Detractor	[0, 6]	Negativo

Tabla 2: Transformación utilizada para obtener la etiqueta deseada.

Debido a esta misma división, es la preferencia del uso de *mixto* sobre *neutro*, ya que las notas 7 u 8 representan un buen servicio con ciertas falencias. El nombre de Neutro no sería apropiado, ya que siguen teniendo una orientación de preferencia.

Para asegurar que la fuente de datos sea confiable y así la transformación tenga sentido, debemos elegir encuestas que posean una cantidad de respuestas que permitan un error muestral inferior al 5 %, asegurando así obtener información correcta. Se usará lo anterior para definir los datos a utilizar y transformarlos a una etiqueta apta para el entrenamiento.

3.2.2. DESCRIPCIÓN DE FUENTE DE DATOS

Los datos que serán utilizados para el entrenamiento corresponden a respuestas de cinco empresas diferentes, pero del mismo rubro. Debido a que necesitamos encuestas que posean una gran representatividad, debemos verificar que el error muestral en ellas sea menor al 5 %. En la Tabla 3 se observa, por cada encuesta, la cantidad de respuestas, destinatarios, y el error muestral correspondiente.

N° de pauta	Cantidad respuestas	Promotores	Detractores	Error muestral
1	1.437	438	602	0.59 %
2	1.000	0	516	3.17 %
3	21.572	9.818	4.894	0.3 %
4	3.938	3.196	234	2.33 %
5	689	481	111	3.97 %

Tabla 3: Tabla de resumen de los datos a entrenar.

En la Tabla 4 se presenta información detallada sobre los comentarios. Cabe destacar que esta información es tras realizar la transformación de las etiquetas y aplicar todas las técnicas de pre-procesamiento de texto. Tras esto, se eliminaron 3.707 respuestas que no contenían información correcta o sin una orientación de preferencia.

Cantidad de respuestas	Largo minimo	Largo máximo	Largo promedio	Tamaño vocabulario
24938	1	643	9	6656

Tabla 4: Tabla de resumen de los datos a entrenar.

En la Figura 9 se entrega un gráfico de columna donde se pueden visualizar las primeras doce palabras que con mayor frecuencia aparecen en el conjunto de datos.

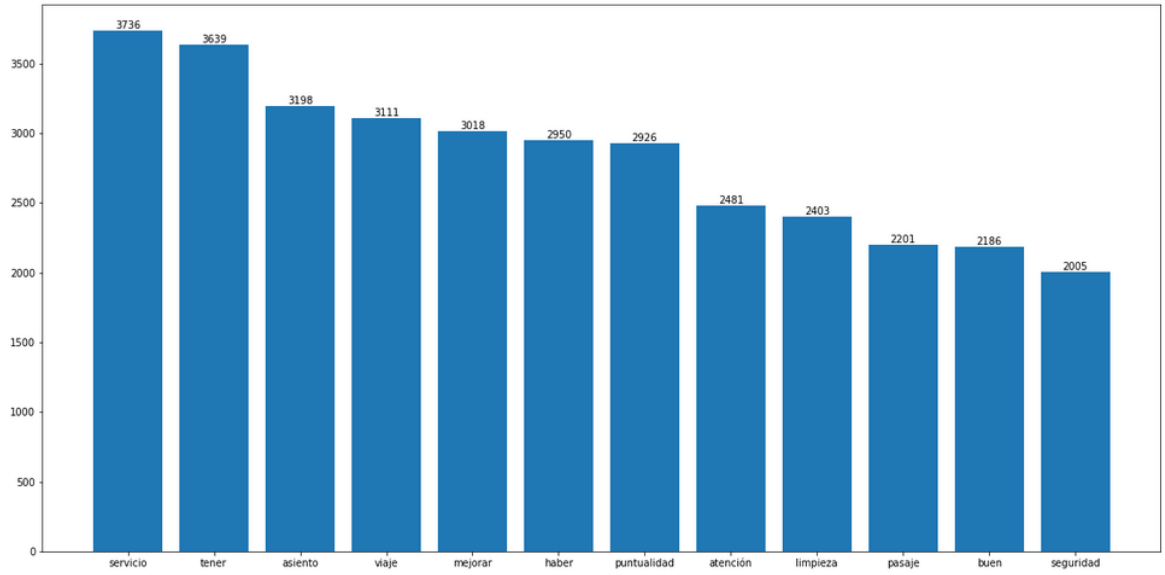


Figura 9: Palabras más frecuentes de nuestro conjunto de datos.

3.2.3. ARQUITECTURA DE REDES NEURONALES

Gracias a la transformación de las etiquetas de los datos, es posible comenzar a utilizar redes neuronales para el entrenamiento. Para esto, se proponen dos modelos de red neuronal recurrente.

LONG SHORT-TERM MEMORY

La red neuronal recurrente estará compuesta primordialmente por unidades LSTM. Son unidades más complejas que las neuronas de la red neuronal recurrente convencional y permiten un mejor manejo del contexto de las secuencias de texto. En adición, al implementar la bidireccionalidad, se obtiene contexto hacia ambos sentidos, potenciando el correcto entrenamiento de la red. La arquitectura de este modelo es la siguiente:

1. Capa de *embedding* que se encarga de transformar las palabras a su correspondiente

vector. Se utilizarán pesos pre-entrenados para mejorar la representación de los datos. Además, el largo máximo de la secuencia de texto aceptado será de 18 palabras.

2. Primera capa bidireccional de 50 celdas, ocupando un dropout de 0.3 y un dropout recurrente del 0.4.
3. Segunda capa bidireccional de 50 celdas, ocupando un dropout de 0.2 y un dropout recurrente del 0.3.
4. Una capa densa de salida de tres neuronas con una función de activación *softmax*, cada una indicando el score del sentimiento.

GATED RECURRENT UNIT

Esta red neuronal recurrente estará compuestas por unidades GRU. En comparación con las LSTM, éstas presentan un diseño menos complejo, lo que permite un tiempo de cómputo menor sin disminuir de forma considerable el desempeño. La estructura que se utilizará será la siguiente:

1. Capa de *embedding* que se encarga de transformar las palabras a su correspondiente vector. Se utilizarán pesos pre-entrenados para mejorar la representación de los datos. Además, el largo máximo de la secuencia de texto aceptado será de 18 palabras.
2. Primera capa GRU de 20 celdas, ocupando un dropout de 0.2 y un dropout recurrente del 0.3.
3. Segunda capa GRU de 50 celdas, ocupando un dropout de 0.2.
4. Una capa densa de salida de tres neuronas con una función de activación *softmax*, cada una indicando el score del sentimiento.

Ambos modelos serán entrenados con *crossentropy categórico*, ya que poseemos más de dos clases de salida. Además, se utilizará el optimizador *Adam*.

Respecto a parámetros de entrenamiento, será entrenado por 100 épocas, ocupando un *batch size* de 256 datos.

Se evaluará el modelo en los datos anteriormente definidos. Se estimará un porcentaje en el conjunto de pruebas superior a un 60 %, el cual será el mínimo aceptable para la tarea.

3.3. CATEGORIZACIÓN DE TEXTO

El problema de categorización de texto se define como la tarea de ordenar de forma automática un conjunto de documentos en categorías de un conjunto predefinido [Sebastiani, 2003].

Esta definición presenta tres aristas a tomar en cuenta: la automatización de la tarea, una definición previa de categorías y, por último, el ordenamiento de los documentos. Cada una de estas aristas presentan un desafío y siempre deben considerarse las opciones según diversos atributos de los datos (documentos) a ordenar.

A pesar de lo anterior, la categorización de texto puede encontrarse con una gran cantidad de problemas al implementarse.

3.3.1. PROCESO DE CATEGORIZACIÓN DE TEXTO

Este proceso debe considerar dos situaciones:

- Un cliente de QServus ofrece un diseño de categorización que utiliza de forma interna y la labor será automatizar el proceso.
- El cliente no posee un modelo de categorización previo, por lo que éste será creado a partir de las respuestas que estos clientes obtengan en sus encuestas, tomando un modelo que se adapta según su rubro.

Sin importar cual situación sea, de ambas se obtendrá un modelo de categorización. Este debe poseer inicialmente categorías, siendo las palabras claves ofrecidas por el cliente o encontrados mediante el algoritmo frecuentista.

El proceso consiste en preprocesar los textos para reducir el vocabulario, como también las correcciones ortográficas. Luego, se hacen comparativas entre las repuestas y las palabras claves de cada categoría para asignarles un puntaje. Este puntaje es asignado usando varios criterios, pero los más determinantes son:

- **Comparación exacta:** Consiste en una coincidencia entre las palabras de una respuesta y las palabras claves de una categoría. También se hace una comparación obteniendo los lemas de ambas partes, aunque implica un puntaje menor a la coincidencia exacta.
- **Similitud coseno:** Se posee una gran base de datos con transformaciones de palabras a vectores, entrenados en una red neuronal. Gracias a esta fuente, se puede calcular la similitud coseno entre vectores, la cual se define como:

$$\text{sim-cos}(\theta) = \frac{\boldsymbol{A} \cdot \boldsymbol{B}}{||\boldsymbol{A}|| \cdot ||\boldsymbol{B}||} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2}\sqrt{\sum_{i=1}^{n} B_i^2}} \tag{29}$$

  Este criterio se vuelve más determinante cuando, tras obtener los puntajes iniciales, retorna más de una categoría con puntaje máximo.

En el proceso, además, se considerarán los siguientes puntos:

- Los comentarios tipo ”nada que agregar” o ”lo mismo que la pregunta anterior” serán siempre excluidos del análisis, ya que no vale la pena asignarles una etiqueta.
- Los comentarios demasiado cortos (refiérase a una o dos palabras) serán difícilmente categorizados, ya que a pesar que se refieran a un tema, no presentarán información importante para el cliente.

Otro punto importante, es que ocupando este mismo modelo de categorización, se hará un análisis respecto al enunciado de la pregunta. La razón es que las preguntas presentan una intención o contexto, haciendo que la gente responda detalles específicos. Por lo mismo, en preguntas del tipo *¿Cómo la empresa ha ayudado respecto a la movilización?* es esperable que la mayoría de las respuestas estén bajo una categoría de ”Transporte” o ”Movilización”.

Un comentario siempre será etiquetado con una sola categoría. Es esperable que respuestas más extensas puedan implicar diversos temas, pero la intención es obtener el contexto principal al cual se refiere, ya que una implementación multi-categórica implicaría una inconsistencia en la información presentada en los indicadores.

3.3.2. PROCESO DE AGRUPACIÓN DE TEXTOS CATEGORIZADOS

La necesidad de agrupar los comentarios surge dada la similitud que existe entre muchos de ellos, lo que al momento de desplegar algún indicador al respecto en pantalla, ocuparía demasiado espacio. La intención es mostrar la mayor cantidad de información relevante en la pantalla sin tener datos que confundan o sean redundantes.

Para lograr este objetivo, se realizarán comparaciones cruzadas entre las respuestas que, gracias a sus parecidos y también a la distancia entre los vectores, podemos detectar si son parecidos, y juntarlos dentro de un *grupo*. Debido a que los grupos pueden poseer muchos comentarios, es importante destacar cuántas respuestas contienen, para así destacar que muchas personas opinan de la misma forma.

3.4. INDICADORES

Los indicadores son las diferentes gráficas que QServus ofrece para la visualización de datos de distinto tipo, con el fin de esclarecer información al usuario para ayudar a su toma de decisiones. Es el núcleo del segundo paso del esquema de QServus: *Analizar* (Ver Figura 1).

QServus, en la construcción de sus indicadores, se cerciora que éstos posean la información de forma muy clara y de fácil comprensión para el cliente. Además, se preocupa de definir

estándares de diseño que describen formas, tamaños y colores según el contexto. Esto último es esencial, ya que presenta dos beneficios primordiales: el primero consiste en que los tamaños y formas hacen alusión a conceptos generalizados en la vida diaria, potenciando la comprensión del cliente al gráfico. El segundo punto, involucra que el cliente tenga un esquema de cómo QServus representa sus indicadores, haciendo que la adaptación de un cliente a nuevos tipos de indicadores sea inmediata.

Debido a lo anterior, al momento de crear nuevos indicadores que permitan mostrar la información que será analizada por la API, estos deben seguir los patrones de diseño para que los usuarios de QServus se adapten rápidamente y sepan leer estos indicadores.

Existen en QServus un conjunto de gráficos que serán la base para generar los nuevos indicadores que mostrarán la información proveniente de nuestra API. Entre ellos tenemos:

1. Gráfico de barras: Consiste en un gráfico de barras horizontales, en donde cada barra representa un atributo y el tamaño de éste una cantidad. Además, si cada atributo toma diversos valores, se pueden utilizar barras acopladas con diferentes colores y así potenciar la capacidad de este gráfico.

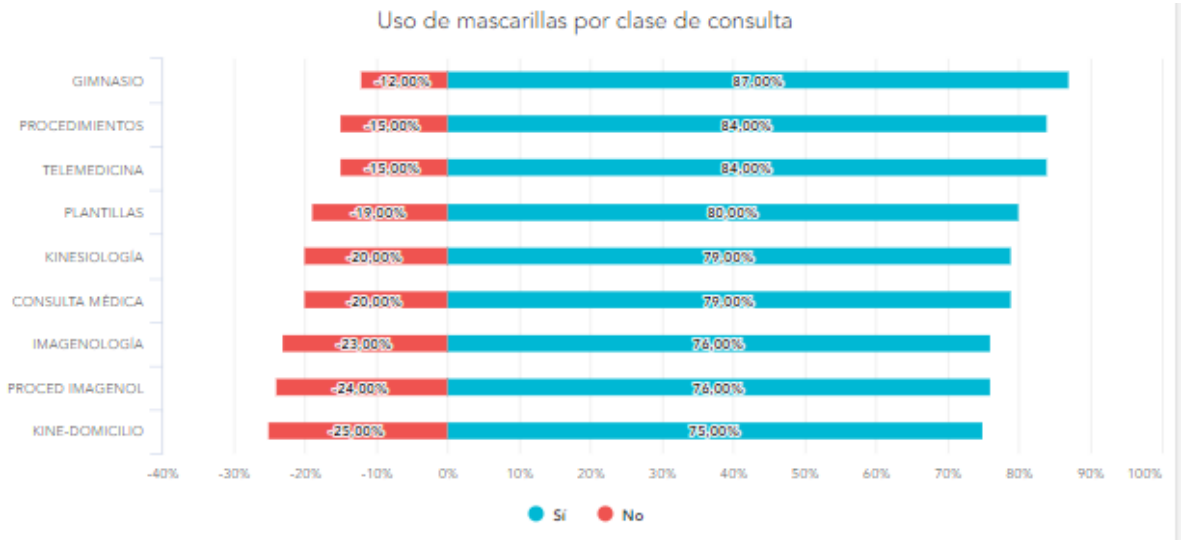


Figura 10: Gráfico de barras implementado en QServus, utilizando una pregunta de tipo Sí/No.

2. Gráfico de torta: Este gráfico consiste en una circunferencia particionada según las frecuencias de ciertas opciones. Es utilizada principalmente para ver el porcentaje que un atributo contiene respecto al total, ayudando a la comparación de sus cantidades.

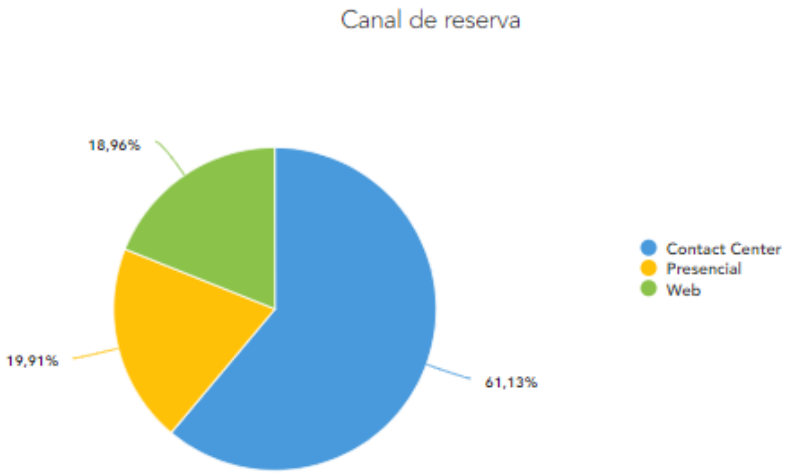


Figura 11: Gráfico de torta implementado en QServus, indicando el porcentaje de personas que utilizan un medio u otro.

3. Nube de palabras: Es un gráfico que presenta un conjunto de palabras claves. No poseen un orden definido (solo no deben solaparse) y el tamaño de una palabra será más grande mientras más veces aparezca en las respuestas de esa pregunta. Sirve esencialmente para detectar de forma fácil los conceptos que están usando los encuestados para responder.

Figura 12: Gráfico de nube de palabras implementado en QServus. Corresponde a una pregunta de justificación general sobre la satisfacción del servicio.

Los gráficos anteriormente expuestos, servirán de base para armar los indicadores que la API dejará a disposición para las aplicaciones de QServus. Al analizar los textos de diversas

empresas, la API ofrecerá nuevos atributos a las respuestas de texto que son: una categorización, un sentimiento y una agrupación de categoría. Con lo último en mente, se han diseñado 6 nuevos tipos de indicadores los cuales serán definidos a continuación.

3.4.1. GRÁFICO DE BARRAS: RÁNKING POR CATEGORÍA

Datos a utilizar: Los atributos del gráfico serán las categorías detectadas. El tamaño de las barras representará la frecuencia de los comentarios categorizados en esa pregunta.

Función del indicador: Mostrar al usuario cuáles son las categorías más mencionadas en las respuestas y detectar las tendencias.

Al tomar el concepto de *ranking*, las barras de categorías van a estar ordenadas de mayor a menor. Será un indicador principal, que dará a grandes rasgos, cuales son las categorías con mayor número de menciones.

Criterios de aceptación

- Presenta datos de ejemplo y reales de forma correcta, sin errores.
- Las categorías pertenecen al estudio.
- Las barras de categorías están ordenadas de mayor a menor.
- El indicador posee una carga menor de 3[s].

3.4.2. GRÁFICO DE BARRAS: CATEGORÍA VS SENTIMIENTO

Datos a utilizar: Los atributos del gráfico serán las categorías detectadas. El tamaño de las barras representará la frecuencia de los comentarios categorizados en esa pregunta, incluyendo un nivel extra que representa el sentimiento.

Función del indicador: Mostrar al usuario cuáles son las categorías más mencionadas en las respuestas y cómo se presenta el sentimiento en los diversos comentarios, para que así el cliente no solo sepa cuáles categorías tienen una mayor mención, si no también si esas menciones son positivas o negativas.

Este gráfico estará presente cuando las respuestas hayan sido procesadas por análisis de sentimiento y categorización de texto. Para esto, se modificará el gráfico de barras: categoría y sentimiento para que las barras sean particionadas. Cada partición poseerá un color que indicará el sentimiento con un tamaño relativo al total. Cada barra tendrá un único tooltip, donde se mostrará la frecuencia para los tres sentimientos. Al igual que el anterior, seguirá siendo ordenado de mayor a menor respecto a la frecuencia total.

Criterios de aceptación

- Presenta datos de ejemplo y reales de forma correcta, sin errores.
- Las categorías pertenecen al estudio.
- La frecuencia de los sentimientos es correcta para cada categoría.
- El orden de la barra particionada es, de izquierda a derecha, positivos, mixtos y negativos.
- Las barras de categorías están ordenadas de mayor a menor.
- El indicador posee una carga menor de 3[s].

3.4.3. NUBE DE PALABRAS MEJORADA

Datos a utilizar: Los datos de la nube serán palabras claves y además bigramas, ambas provenientes de las respuestas de texto de una misma pregunta.

Función del indicador: Mostrar al usuario cuáles son las palabras más mencionadas y también los bigramas más utilizados.

La denominación de **mejorada** a este tipo de gráfico recae en la inclusión de los bigramas. Un bigrama o un 2-grama es un par de palabras que aparecen de forma consecutiva en un corpus. Gracias a estos bigramas podemos añadir contexto a los términos que aparecían en la nube de palabras corriente.

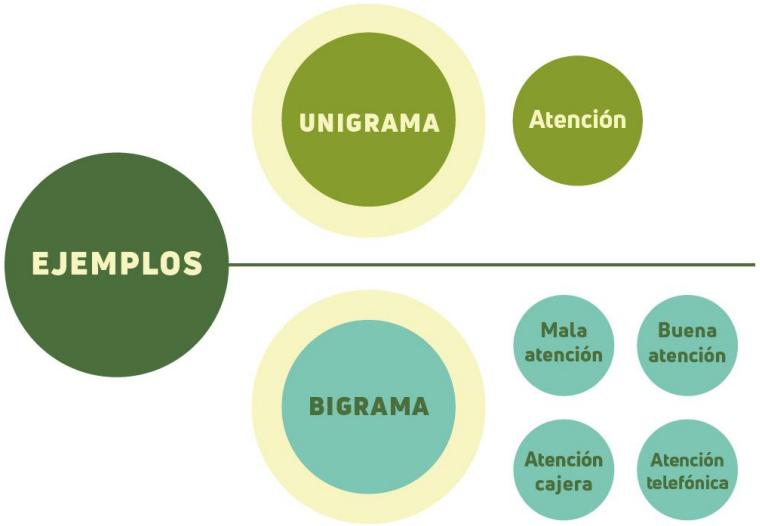


Figura 13: Comparativa entre unigramas y bigramas, y cómo estas últimas nos entregan más información.

La inclusión de un bigrama parece ser sencilla, pero existen dos dificultades en su implementación:

- Un bigrama implica que ambas palabras deben poseer el mismo tamaño y también que el conjunto será en la mayoría de las veces más largo que un 1-grama. Esto implica que un bigrama ocupará más espacio en el canvas que un unigrama, como mínimo en el eje horizontal.
- Un bigrama tiene mucha más importancia semántica que un unigrama, ya que ofrece más contexto y es más útil observar que dos palabras aparecieron muchas veces de forma consecutiva. Debido a esto, se debe incluir esa importancia en el gráfico para que los bigramas destaquen.

Para esto se considerará una escala distinta para unigrama y bigrama, de tal forma que su tamaño no dependa de la frecuencia (como cualquier gráfico de nube de palabras se basa para definirlo). Por lo tanto, un bigrama puede ser más grande en el canvas que un unigrama de mayor frecuencia.

Respecto al tooltip para cada palabra, se mostrará la frecuencia de ese unigrama o bigrama dentro del estudio.

Criterios de aceptación

- Presenta datos de ejemplo y reales de forma correcta, sin errores.
- Los bigramas destacan a pesar de su poca frecuencia.
- Las palabras tienen buena ortografía.
- El indicador posee una carga menor de 3[s].

3.4.4. TREEMAP

Este indicador presenta un gráfico *treemap*, que son cuadriláteros que representan las categorías encontradas en el análisis. Cada figura poseerá un subnivel en donde se muestran las palabras claves más frecuentes. Para acceder a este subnivel, se debe hacer click en el cuadrilátero deseado y se mostrará el detalle.

Datos a utilizar: Las categorías de las respuestas de texto. Además, para cada categoría se usarán las palabras claves más frecuentes. Como métricas, se usarán su frecuencia y porcentaje respecto al total.

Función del indicador: Mostrar de una forma más interactiva el porcentaje de las categorías en las respuestas de una pregunta y las palabras claves más frecuentes de una categoría.

El treemap tiene como intención producir un indicador más interactivo, mostrando las categorías y palabras claves que las representen. Como se explicó anteriormente, el nivel superior consiste en categorías y el nivel inferior en palabras claves. El indicador debe cumplir con ciertos comportamientos y criterios para cumplir la definición anterior:

- En el nivel superior de categorías, cada cuadrilátero debe poseer un color distinto para poder diferenciar mejor la pertenencia de una categoría.
- Al hacer click en un cuadrilátero, éste mostrará sus sub-divisiones, expandiéndose hasta ocupar todo el canvas para mostrar el siguiente nivel. El color de este sub-nivel es el mismo que la categoría a la cual pertenece.
- Para volver al nivel superior, se puede hacer click en cualquier cuadrilátero de palabra clave o en un botón en la esquina superior derecha. Al hacer esta acción, se comprimirán los cuadriláteros, volviendo a la vista del nivel superior.
- El tooltip para cada cuadrilátero indicará la frecuencia y el porcentaje respecto al total.
- En el sub-nivel inferior, se mostrarán solamente palabras claves que posean frecuencia mayor a tres.

Criterios de aceptación

- Presenta datos de ejemplo y reales de forma correcta, sin errores.
- Las categorías pertenecen al estudio.
- La frecuencia y porcentajes de cada categoría y palabras claves son correctas.
- Al hacer click en una figura en el primer nivel, lleva al detalle de las palabras claves de la categoría respectiva.
- Al hacer click en una figura en el segundo nivel, debe volver a mostrar el nivel superior.
- El indicador posee una carga menor de 3[s].

3.4.5. GRÁFICO DE SATISFACCIÓN/SENTIMIENTO VS CATEGORÍA

Datos a utilizar: De todas las respuestas de una pregunta se utilizarán sus categorías, sentimientos y frecuencias respectivas. También, en caso de que una pregunta de texto esté relacionada con otra pregunta de escala, puede reemplazarse el eje de sentimiento por satisfacción neta.

Función del indicador: Mostrar de forma concisa toda la información que la API es capaz de ofrecer respecto a la analítica de texto en un solo gráfico, presentando al cliente todas las aristas en torno a las respuestas de texto abierto.

Este gráfico representa una complejidad alta, ya que implica la creación de múltiples gráficos de torta que entregan mucha información de forma individual y deben estar alineados respecto a un gráfico de tipo barra vacío. Para aclarar el diseño de este gráfico se explicará en pasos de construcción:

- **Círculo interior:** El círculo interior define a la categoría y su frecuencia, siendo esta última representada por el tamaño del círculo.
    - **Contenido:** Poseerá dentro de éste el nombre de la categoría que representa. Además, el porcentaje de satisfacción que, dependiendo su valor, puede tomar color verde, amarillo o rojo.
    - **Color:** Un color gris muy claro. La intención que no destaque tanto como el círculo exterior, pero tampoco que se camufle con el fondo blanco del canvas.
    - **Tooltip:** Poseerá el porcentaje y el texto *recomendación neta* para detallar el significado del porcentaje.
- **Círculo exterior:** Consiste en un anillo que encierra al círculo interior. Este puede ser particionado hasta tres piezas donde a pesar que se refieren a cosas distintas, cumplen el mismo formato. Cada partición sigue:
    - **Color:** Poseerá un color verde, amarillo o rojo, haciendo alusión a los sobre estándar, estándar y bajo estándar, respectivamente.
    - **Tooltip:** Contiene el porcentaje respecto al total de promotores, pasivos y detractores, seguido de un texto que lo identifica entre las tres opciones. Además, el tooltip posee un fino borde del mismo color para asociar mejor el concepto.

  Por lo tanto, el anillo exterior detallará el porcentaje de promotores, pasivos y detractores para hacer la relación con el porcentaje de satisfacción total.
- **Eje Y:** Define la escala de la recomendación general en el gráfico. Igual que los anillos exteriores, estará particionado en tres partes con los estados y colores anteriormente mencionados. El rango de la recomendación general va desde -100 a 100. Gracias a este eje, podemos posicionar los círculos en el gráfico.
- **Posicionamiento circulos acoplados:** El posicionamiento de los círculos dependerá de la definición de los eje x e y. Los criterios para posicionar son:
    - Eje Dependiente: Como este eje indica la recomendación general, cada círculo será posicionado de tal forma que el centro esté alineado en el eje dependiente a la misma altura donde se encuentra su valor de recomendación general.

- Eje Independiente: Este eje no participa para el análisis del indicador, pero si ayuda a situar los círculos después de asignarle una posición en el eje dependiente, ayudando a que no se solapen al momento de dibujarlos.

En resumen, la cantidad de información que contendrá este indicador es:

- Cantidad de comentarios dentro de una categoría.
- Sentimiento/recomendación de respuestas.
- Porcentajes de ambos puntos anteriores respecto al total.
- Sentimiento/recomendación neta.

Estos puntos dentro del indicador ayudarán a hacer la comparativa de categorías mucho más fácil, además de identificar inmediatamente cuáles son las categorías más comentadas y más críticas, que ayudarán a tomar decisiones oportunas.

Criterios de aceptación

- Presenta datos de ejemplo y reales de forma correcta, sin errores.
- Las categorías pertenecen al estudio.
- Los porcentajes para la recomendación neta o sentimiento son los correctos para cada categoría.
- Los tamaños de los círculos corresponden a la cantidad total de respuestas por categoría.
- Ninguno de los círculos se solapan en el canvas.
- El centro del círculo está a la misma altura que el valor de la recomendación neta del círculo en el eje Y.
- El indicador posee una carga menor de 3[s].

3.4.6. TABLA DE COMENTARIOS

Datos a utilizar: Se utilizan las categorías y las agrupaciones de textos con sus respectivas frecuencias.

Función del indicador: Permite mostrar en pestañas principales las categorías encontradas y en cada una de ellas, un despliegue que detalla de forma directa los comentarios que posee esa etiqueta.

Este indicador ayudará al cliente cuando quiera revisar de forma directa los comentarios y a cuál categoría corresponden. Se incluirán los siguientes criterios en la construcción de la tabla:

1. Las pestañas de categorías serán ordenadas desde la mayor frecuencia a la menor.
2. Los comentarios aparecerán tal cual como fueron respondidos en la encuesta. A pesar de utilizar procesos que corrigen las palabras, debemos mostrar las respuestas originales.
3. Se implementará en esta tabla el concepto de grupos de categorías. Por lo anterior, los comentarios en las pestañas desplegables también poseerán frecuencia, indicando cuántos comentarios engloba ese grupo.

Criterios de aceptación

- Presenta datos de ejemplo y reales de forma correcta, sin errores.
- Las categorías pertenecen al estudio.
- La frecuencia de los grupos de categoría son correctos.
- Las pestañas de categorías están ordenadas de mayor a menor.
- El indicador posee una carga menor de 3[s].

Los indicadores serán generados por la API en un formato base, suficiente para que las aplicaciones de QServus puedan realizar modificaciones según necesiten y dependiendo del contexto de sus dashboard.

3.5. QSERVUS AI

QServus AI es el nombre con el cual se define la API de inteligencia artificial de QServus y que proveerá todas las herramientas necesarias para realizar análisis de datos descriptivo y predictivo según se requiera. Dentro de este diseño de REST API, se buscará implementar las mejores prácticas de diseño y de QServus para que la comunicación sea efectiva, eficiente y de fácil comprensión.

En esta sección se explicará porqué se decidió por una REST API como arquitectura y diseño de software para la aplicación. Se mostrará el modelo de datos base para realizar las diferentes tareas. También, la definición de los endpoints y los formatos para que las peticiones sean correctas y, por último, cómo serán implementadas en la API las tres soluciones descritas anteriormente.

3.5.1. COMUNICACIÓN ENTRE APLICACIONES

QServus posee muchas aplicaciones para distintos propósitos. Dentro de su conjunto, se encuentran las siguientes:

1. QServus Pro: Es la aplicación principal de QServus, que reúne todas las funcionalidades que puede ofrecer la plataforma. Está orientada a empresas más grandes donde puedan personalizar sus encuestas e indicadores, además de definir tareas automáticas, crear bitácoras para registrar interacción con clientes, etc.
2. QServus Lite: Es la versión más liviana de QServus Pro, pensada principalmente para plataformas móviles, pero igualmente accesible mediante otros dispositivos. Está enfocada a empresas más pequeñas que necesitan información general de sus clientes. Debido a lo anterior, es más restrictiva respecto a los diseños e indicadores presentes.
3. Cliente Interno: Versión de QServus Pro enfocada principalmente en encuestas para evaluación interna de empresas. Presenta diseños de encuesta adicionales como también indicadores personalizados para ese contexto.
4. SAC Lite: Es una aplicación centrada en el servicio de atención a clientes, permitiendo el registro de tickets sobre las atenciones de diversas solicitudes.
5. QSend It: Una aplicación que se encarga en el envío y registro de tareas automatizadas, principalmente de anuncios o mensajes por correo.
6. Aprendus: Aplicación de aprendizaje on-line de diversos temas.

Como se puede apreciar, son muchas aplicaciones. Crear para cada una un módulo que se encargue de realizar análisis de inteligencia artificial traería los siguientes prejuicios:

- Los módulos no serían escalables, ya que funcionalidades que beneficien a muchas aplicaciones implica más tiempo de desarrollo debido a que se encuetran separados.
- Difícil mantención de versiones de los módulos de inteligencia artificial, teniendo aplicaciones con versiones más antiguas del módulo que otras.
- La carga de este nuevo módulo para los servidores de cada aplicación se acumularía, haciendo que las funcionalidades principales de cada aplicación se vean mermadas.
- Cualquier error que pueda presentar este módulo, al estar construido dentro de cada aplicación, puede producir caídas o demoras en el servicio, influyendo directamente en los requerimientos no-funcionales del up-time.

Muchos argumentos rechazan la idea de una implementación directa en cada aplicación, por lo que se debe optar por un software único que esté a la escucha de diversas peticiones y que, dependiendo de qué aplicación la realiza, se ejecute siguiendo el contexto requerido. Debido a esto, se decide la construcción de una API, ya que en contraste con las razones anteriores:

- La API es escalable, pues con una buena definición del modelo de datos y la obtención de los recursos, puede manejar una gran cantidad de datos y de muchas aplicaciones.
- Al ser única, el control de versiones es más sencillo. Además, la API puede ser dividida en módulos, lo que hace que ciertas funcionalidades no topen con otras y así no afecten a su comportamiento.
- La API al ser independiente de las aplicaciones, es ejecutada en un servidor aparte por lo que la carga se concentra en un solo servidor.
- No afecta en el up-time de las demás aplicaciones, ya que si la API llegase a fallar, las aplicaciones recibirán códigos HTTP con el resultado de las peticiones, permitiéndoles manejar las excepciones y mantener de forma constante el porcentaje de up-time establecido.

A continuación, se mostrará de forma concisa el modelo de datos inicial que se utilizó para ordenar los datos entrantes y los resultantes en los diferentes procesos y algoritmos que serán ejecutados.

3.5.2. MODELO DE DATOS

QServus AI requiere un modelo de datos que represente todos los recursos necesarios para los análisis como así también los resultados generados por éstos.

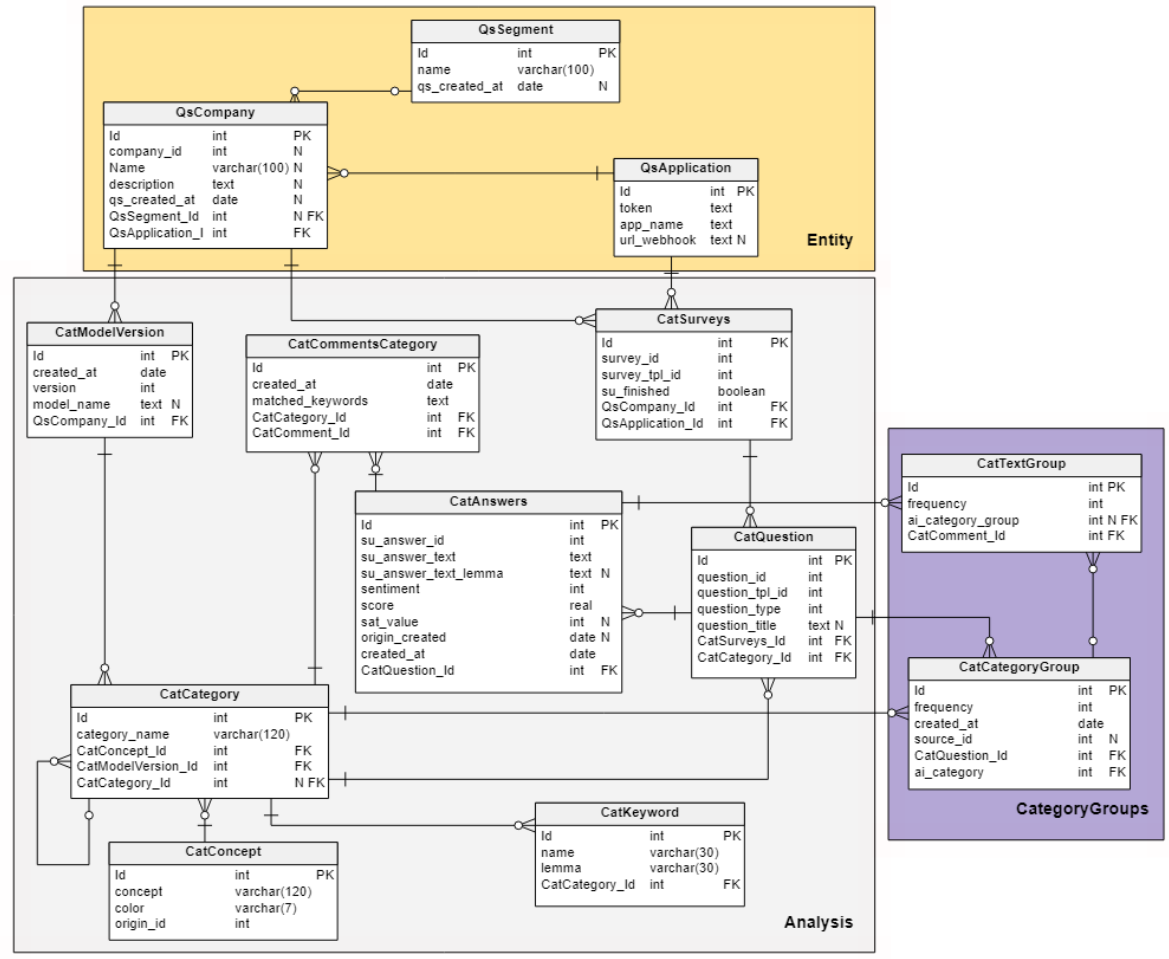


Figura 14: Modelo de datos con que la API trabajará para el análisis CRUD de sus recursos.

En la Figura se aprecian las tablas diseñadas para QServus AI. Se dividen en tres grupos:

- **Entity:** Se centra en las tablas necesarias para QServus AI para identificar las aplicaciones que realizan las peticiones y filtrar información.
- **Analysis:** Concentran las tablas que almacenarán la información crucial para identificar los recursos, como también almacenar resultados de análisis.
- **CategoryGroups:** Consiste en tablas que almacenarán las categorías de grupos.

Estos grupos en conjunto y con la definición de los endpoints, permitirán a QServus AI a realizar las tareas que fueron diseñadas en las secciones anteriores.

3.5.3. DEFINICIÓN DE ENDPOINTS

QServus AI debe definir endpoints en los cuales las diferentes aplicaciones podrán acceder, guardar y/o eliminar datos de la API como también ejecutar los diferentes análisis que están a su disposición. El detalle de cada endpoint que se definió se encuentra en la sección de *Anexos*, debido a su gran extensión.

Todos los endpoints presentes en la API, aunque realicen distintas operaciones en los datos o análisis, presentan ciertas características en común, que serán definidas a continuación.

- Cada endpoint debe ser solicitado con un token de autenticación que indique qué aplicación es la que solicita el recurso. Claramente, es parte de una capa de seguridad para que solo las aplicaciones registradas en QServus AI puedan realizar acciones en él.
- Gracias al token de autenticación anterior, también es posible definir qué datos están disponibles para cada aplicación y restringir los accesos entre datos de las aplicaciones si es necesario.
- Incumplir con la regla del token de autenticación siempre será respondida con un error de código HTTP 401 UNAUTHORIZED. Considera los casos cuando no se envía token o se envía pero no está registrado en QServus AI.
- Cuando una aplicación requiera acceder a los datos de otra, siempre se devolverá un error de código HTTP 403 FORBIDDEN, indicando que no posee los permisos.
- Todas las peticiones de tipo POST poseen un formato estructurado y definido para la petición. Hay funcionalidades que revisan el correcto comportamiento y, en caso contrario, devolverán un error de código HTTP 400 BAD REQUEST, indicando qué parte de la estructura está errónea o pérdida.
- Los endpoint de análisis no entregarán de forma inmediata los resultados debido a que éstos pueden demorarse. Estos serán ejecutados de forma asíncrona, y notificarán a la aplicación cuándo el procesamiento esté finalizado, adjuntando un identificador del análisis para que puedan ser solicitados.

CAPÍTULO 4
VALIDACIÓN DE LA SOLUCIÓN

En esta sección se detallará la puesta en marcha de las soluciones anteriormente definidas. Se presentarán métricas de evaluación como también resultados de categorización. Por otra parte, se mostrarán los indicadores de forma gráfica como también el desempeño de la API con los distintos endpoints que se crearon.

4.1. ANÁLISIS DE SENTIMIENTO

Para poder hacer las comparaciones entre ambos modelos y los diferentes valores en sus hiperparámetros, se utilizarán las siguientes métricas:

1. *Accuracy* (acc): Es el porcentaje, en un aspecto global, de la cantidad de clasificaciones correctas respecto al total de observaciones.
2. *Precision* (pr): Es el porcentaje de las clasificaciones de una clase que son correctas. Se calcula con la siguiente fórmula:
$$C_{pr} = \frac{TP}{TP + FP} \tag{30}$$
3. *Recall* (rc): Consiste en la cantidad de clasificaciones correctas de una clase respecto a la cantidad real de esa clase en los datos de entrenamiento. Este se calcula con lo siguiente:
$$C_{rc} = \frac{TP}{TP + FN} \tag{31}$$
4. *f1-score*: Es la media armónica entre las dos métricas anteriores, con el fin de obtener una evaluación que considere ambos aspectos. Se calcula con la siguiente fórmula:
$$F_1 = 2 \cdot \frac{pr \cdot rc}{pr + rc} \tag{32}$$

Cada una de estas métricas, a excepción del *accuracy*, serán diferentes para cada clase. El *accuracy* cubre un aspecto más global a cómo el modelo predice de forma correcta la categoría de cada dato. En cambio los tres restantes están más ligado a cómo el modelo se desempeña al predecir datos de una clase u otra.

Esta evaluación se dividirá en dos fases: la primera consiste en el entrenamiento de ambos modelos y comparar sus resultados. La segunda se vuelven a entrenar los datos pero utilizando *cross-validation*.

4.1.1. FASE 1: EVALUACIÓN DE MODELOS

Ambos modelos son entrenados con los mismos hiperparámetros, los cuales son:

- Épocas: 50
- Tamaño del batch: 512
- Tamaño de embedding: 30
- Tamaño del vocabulario: 6659

Primero, revisaremos los gráficos que definen el *accuracy* del modelo y, por otra parte, la fluctuación de la función de pérdida.

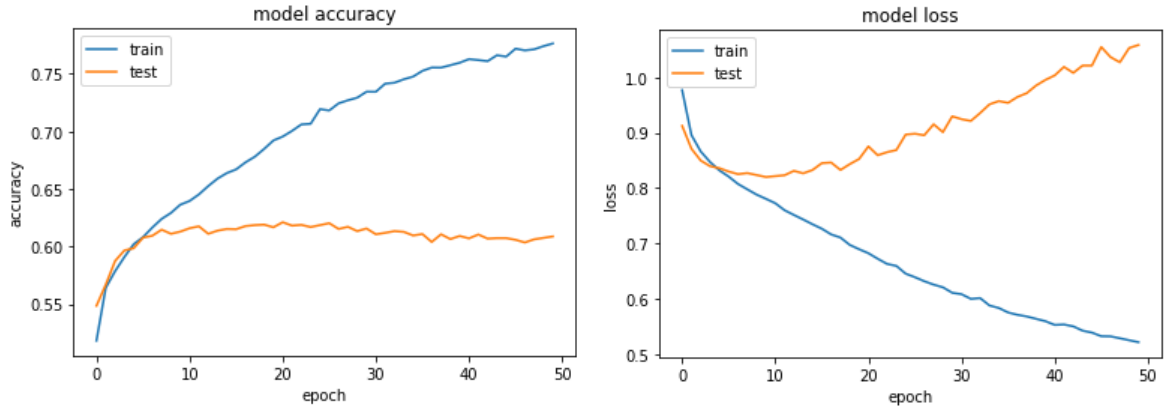


Figura 15: Función de *accuracy* a la izquierda y la función de pérdida a la derecha, para el modelo utilizando LSTM.

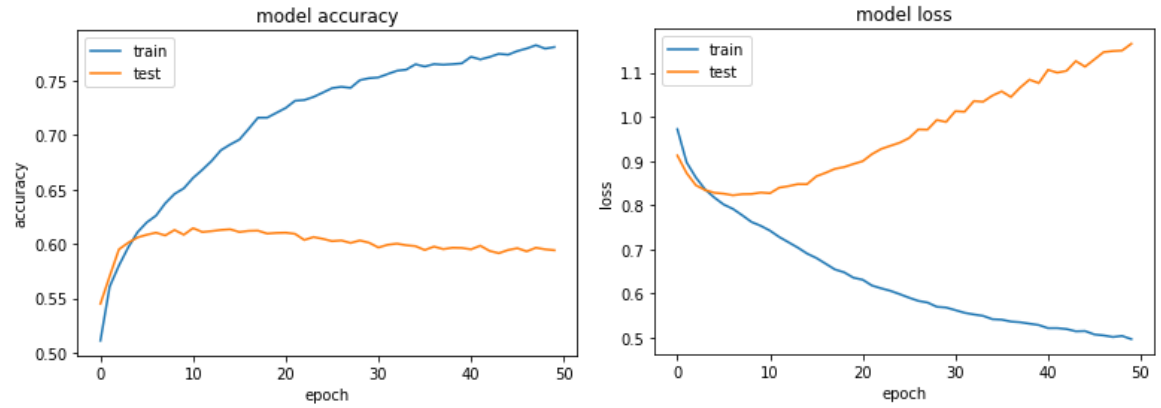


Figura 16: Función de *accuracy* a la izquierda y la función de pérdida a la derecha, para el modelo utilizando GRU.

A simple vista, podemos ver la presencia de sobreajuste, ya que el *accuracy* del conjunto de entrenamiento mejora constantemente, mientras que en el conjunto de pruebas mejora las primeras épocas, para después mantener su fluctuación cercano al 60 %. Esto quiere decir que el modelo está aprendiendo de buena forma con los datos de entrenamiento, pero se ve incapaz de demostrar esa pericia en datos que no ha visto.

Comparando ambos modelos, apreciamos que el modelo con GRU es menos propenso a fluctuaciones y a mejorar de forma constante a través de las épocas, mientras que el modelo con LSTM posee ciertas recaídas durante las épocas.

Para revisar esto a más detalle, la matriz de confusión muestra los falsos positivos y negativos para cada clase (Figura 17):

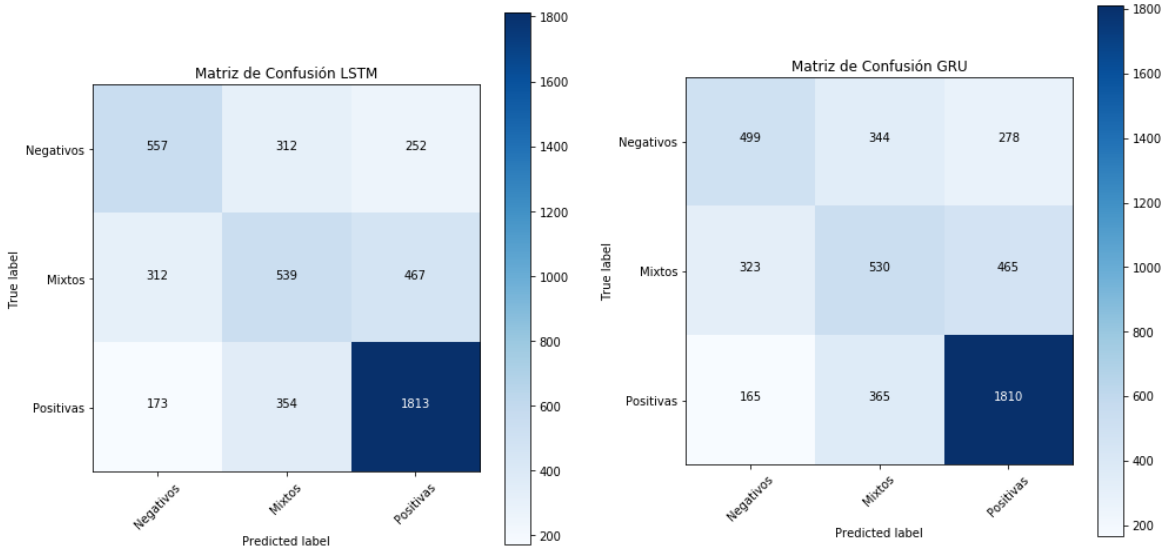


Figura 17: Matrices de confusión para el modelo con LSTM (izquierda) y para el modelo usando GRU (derecha).

Ambos modelos se desempeñan mejor etiquetando respuestas positivas, aunque LSTM es un mejor clasificador de negativos mientras que GRU es de respuestas mixtas. Aún así, apreciamos una gran cantidad de falsos para cada clase. Respecto a las métricas de evaluación, revisar Tabla 5 y 6:

Clase	Precision	Recall	f1-score
Negativos	53.5 %	49.7 %	51.5 %
Mixtos	44.7 %	40.9 %	42.7 %
Positivos	71.5 %	77.5 %	74.4 %

Tabla 5: Métricas de evaluación para el modelo con unidades LSTM.

Clase	Precision	Recall	f1-score
Negativos	50.6 %	44.5 %	47.3 %
Mixtos	42.8 %	40.2 %	41.5 %
Positivos	70.9 %	77.4 %	74 %

Tabla 6: Métricas de evaluación para el modelo con unidades GRU.

Con esta comparación, es más evidente que el modelo con unidades LSTM tiene un mejor desempeño que el de unidades GRU. A pesar que no se diferencian mucho respecto a cuántas predicciones de una clase son realmente correctas (recall), el modelo LSTM es menos propenso a confundirse de clase al predecir.

En líneas generales, ambos modelos presentan sobreajuste debido a la complejidad de los textos y sus etiquetas, las cuales son provenientes del NPS. Debido a esta misma razón, se realiza la segunda fase implementando *cross-validation* para poder enfrentar esta problemática.

4.1.2. FASE 2: CROSS-VALIDATION

Cross-validation consiste en particionar los datos a entrenar asignando diferentes porciones al conjunto de entrenamiento y de pruebas. La razón es evitar un sesgo respecto a qué datos se encontraban en cada conjunto. Por ejemplo, al hacer una partición puede que los datos con sentimiento positivo estén más concentrados en el conjunto de entrenamiento que en el de validación, lo que puede explicar el bajo desempeño de la red cuando etiqueta respuestas mixtas o negativas.

Para esta evaluación se utilizan ambos modelos y se realizan 5 ordenamientos diferentes para los datos.

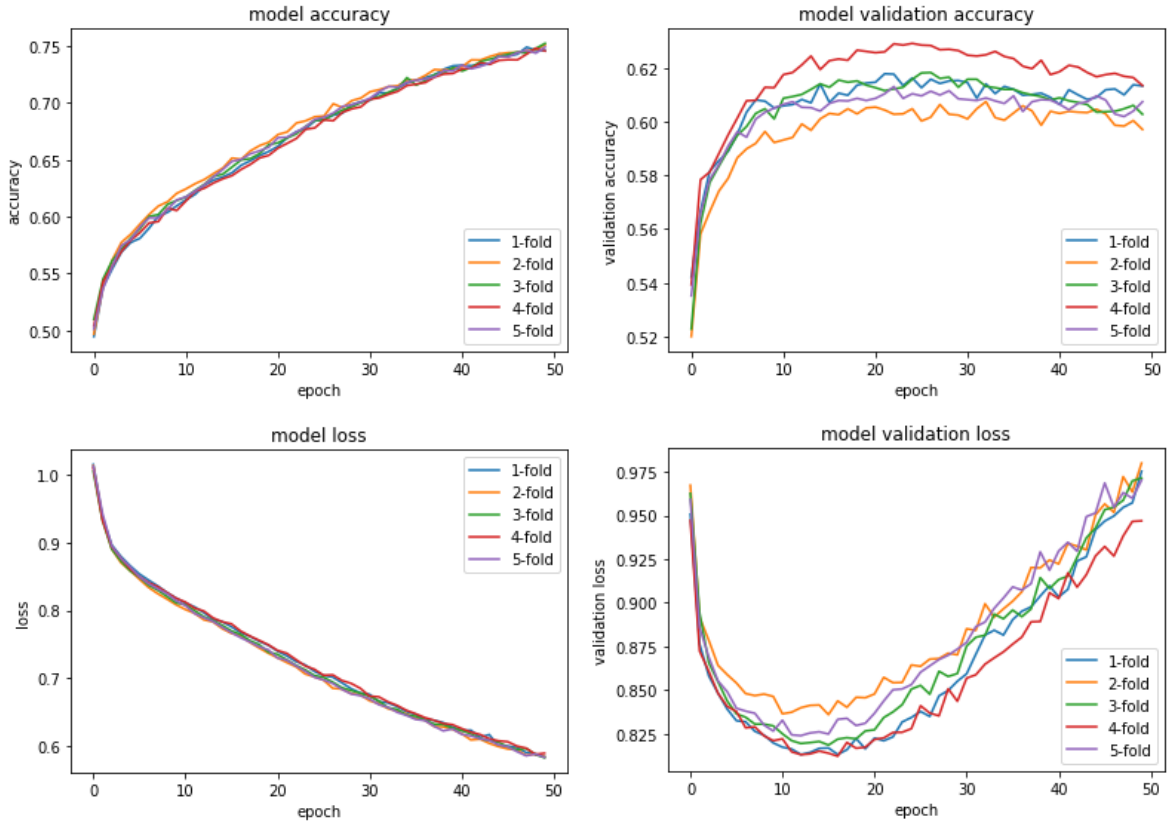


Figura 18: Resultados para el modelo LSTM. En la columna izquierda los resultados de entrenamiento y la columna derecha los de pruebas. La primera fila consiste en la métrica del *accuracy* y le fila inferior a la función de pérdida.

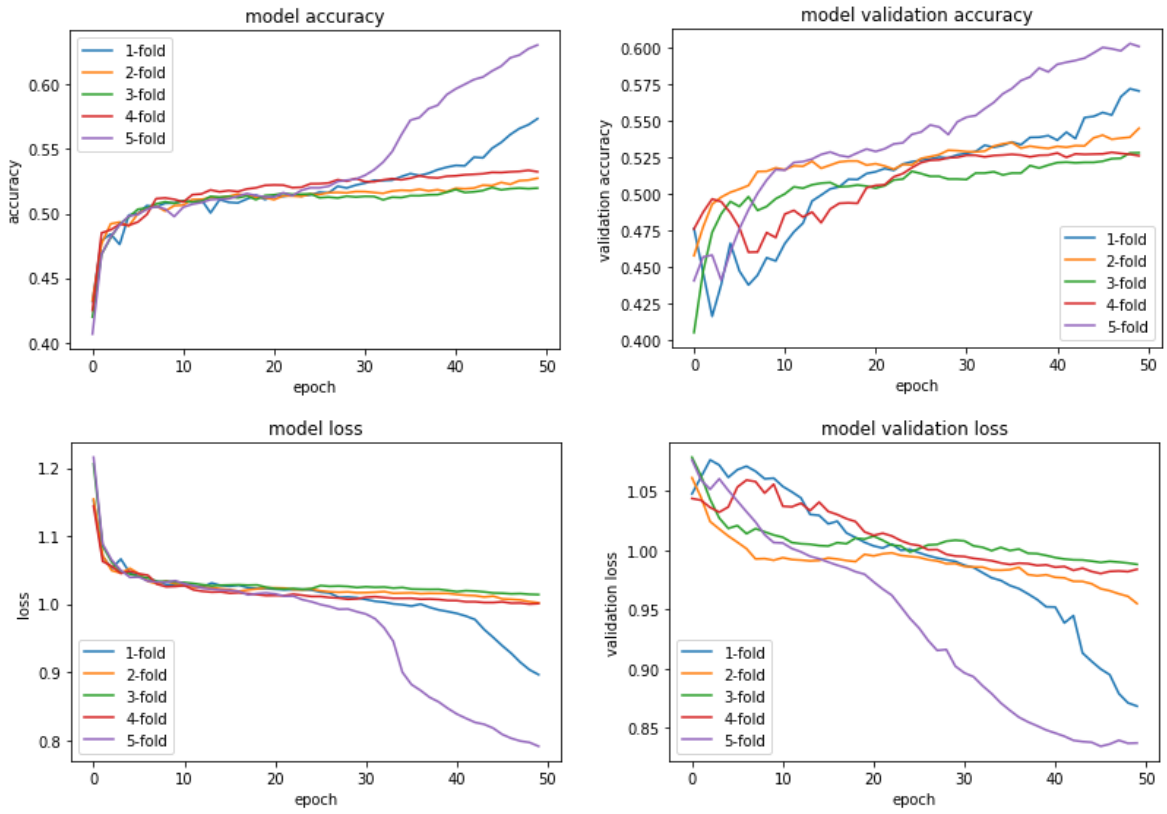


Figura 19: Resultados para el modelo GRU. En la columna izquierda los resultados de entrenamiento y la columna derecha los de pruebas. La primera fila consiste en la métrica del *accuracy* y le fila inferior a la función de pérdida.

Podemos apreciar que la diferencia entre los modelos se hace mucho más evidente. El modelo GRU es muy sensible a la distribución de los conjuntos de datos, influenciándose fácilmente, en cambio el modelo LSTM es mucho más estable ante diferentes conjuntos de datos. Esa estabilidad nos hace escoger, a primera vista, el modelo LSTM.

Revisando los resultados del modelo LSTM, escogemos el modelo que fue entrenado con el cuarto ordenamiento, ya que se obtuvo una mejor precisión en el conjunto de pruebas.

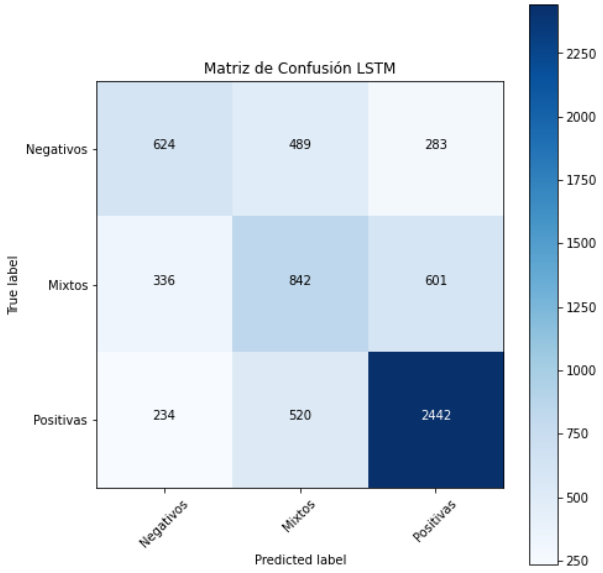


Figura 20: Matriz de confusión del modelo LSTM con el cuarto ordenamiento del conjunto de datos.

Clase	Precision	Recall	f1-score
Negativos	52.3 %	44.7 %	48.2 %
Mixtos	45.5 %	47.3 %	46.4 %
Positivos	73.4 %	76.4 %	74.9 %

Tabla 7: Métricas de evaluación para el modelo con unidades GRU.

Tras este análisis, con el modelo entrenado con el cuarto ordenamiento, se obtuvo el mejor resultado de la prueba. Aún así, los porcentajes en el conjunto de validación no parecen ser muy convincentes. Para salir de dudas, comprobamos el desempeño de la red con ciertos ejemplos, los cuales están detallados en la Tabla 8.

Respuesta	Etiqueta del modelo
en general se a mejorado mucho, falta tener mas tecnología o sistemas mas completos	Mixto
Hay muchos procedimientos en donde se deben considerar a las áreas comerciales, para mejorar el workflow de este, y para que haya seguimiento de tal.	Negativo
implementar sistema de seguimiento de solicitudes para todas las áreas	Mixto
mayor compromiso y proactividad de las personas que integran cada una de las áreas	Negativo
responder el teléfono, y no dar a entender que el problema es del ejecutivo sino un problema banco	Negativo
Encuentro que en ocasiones falta comunicación , sin embargo dada la situación actual la comunicación via Teams encuentro que ha sido favorable	Mixto
Que todos manejen la misma información, remar todos para el mismo lado	Mixto
Excelente disposición y rapidez en la entrega de requerimientos	Positivo
Con menos paradas, se aprecia la rapidez del tren y mejora mucho la experiencia del viaje	Positivo
Buenas tardes,me encantó viajar y así será desde este viaje en adelante,muchas gracias	Positivo

Tabla 8: Desempeño de la red neuronal recurrente con respuestas de diversas encuestas.

Se aprecia que el modelo tiene una predisposición a categorizar los comentarios en Mixto, dejando en claro que al modelo le cuesta definir la dualidad positiva o negativa.

4.2. CATEGORIZACIÓN DE TEXTO

4.2.1. FUENTE DE DATOS

QServus, como se ha mencionado reiteradamente, posee muchas encuestas de diversos clientes con preguntas de texto abierto, por lo que la la cantidad de datos disponibles es abundante. Esto si bien es una ventaja para ejecutar algoritmos de IA, trae también un problema importante que se produce al tener muchas respuestas de clientes que pertenecen a diferentes rubros, ya que un modelo de categorización general puede no identificar las reales intenciones de los encuestados, como también puede escaparse información importante de una categoría a otra, sobre todo si las *keywords* van aumentando o disminuyendo su importancia según el contexto. Tras lo anterior, se considerará un contexto específico, en el cual se tomarán sus datos y se creará un modelo de categorización de texto que pueda responder a las principales problemáticas del rubro seleccionado, el cual servirá de puntapié para ir mejorando los modelos por empresa.

Se considerará para este análisis las respuestas de 5 encuestas, las cuales engloban una cantidad de 40 preguntas de texto abierto, sumando en total unas 3747 respuestas. El rubro escogido para este análisis será el de finanzas, ya que es uno de los rubros que más encuestas y respuestas posee. De manera detallada, en la Tabla 9 se presenta información cuantitativa de la fuente de datos a utilizar.

N° de encuesta	N° de preguntas de texto	N° de respuestas en encuesta
1	8	169
2	8	729
3	8	1646
4	8	340
5	8	863

Tabla 9: Descripción cuantitativa de las encuestas donde se realizará la categorización de texto con el modelo de categorización de teletrabajo.

4.2.2. DISEÑO MODELO DE CATEGORÍAS

El diseño que se ocupará en este trabajo fue construido para un contexto donde se evalúa la experiencia con el teletrabajo dentro de una empresa, considerando el contexto de pandemia mundial producido por COVID-19. Para esto, se consideraron las siguientes categorías:

- **Tiempos de respuesta:** Se refiere al tiempo de respuesta que perciben los trabajadores en diversos procesos de su empresa.
- **Infraestructura:** Explica todas las herramientas físicas que necesitan los trabajadores para poder realizar su labor en casa. Estos pueden ser una computadora adecuada, impresoras, etc.
- **Internet:** Indica cómo es la experiencia con el internet de los hogares y si cumple los estándares necesarios para un buen desempeño.
- **Sistemas:** Posee los comentarios que se refieren a cómo los sistemas de la empresa han funcionado en el contexto de pandemia.
- **Operatividad de procesos:** Cómo los procesos se han adaptado (o no) al teletrabajo.
- **Horario:** Se refiere a cómo los tiempos y horarios de la rutina del trabajo en oficina han cambiado al momento de trabajar desde casa.
- **Compatibilidad familiar:** Explica cómo se ha compatibilizado el trabajo en casa con las necesidades hogareñas y familiares.
- **Apoyo:** Indica cómo la empresa ha ayudado a sus trabajadores (o no) para poder superar las nuevas dificultades que puede presentar el teletrabajo.

- **Comunicación:** Posee los comentarios que tratan sobre la comunicación entre trabajadores y cómo se establecen las condiciones e informaciones.
- **Otros:** Son comentarios que no fueron categorizados en ninguna de las anteriores.

Con lo anterior definido, se realiza la categorización de las respuestas con el modelo definido. En la tabla 10 y 11 se especifica una porción de las respuestas con su categoría asignada.

Comentario	Categoría
agradezco la posibilidad de poder estar de manera remota sin la preocupación de contagiarme	Apoyo
Es vital contar con plataformas robustas y con un muy buen soporte, que permitan lograr continuidad laboral.	Sistemas
Yo estoy con teletrabajo por mi embarazo, y por el tipo de trabajo que realizaba en la oficina,se me dificultó al principio el hacer labores porque trabajamos con documentos físicos, pero con el tiempo me dio la oportunidad de apoyar en actividades nuevas y a otras áreas por lo que terminó siendo mejor	Operatividad - Procesos
cuesta el horario de cierre, por que igual se cierra mas tarde	Horario
Me a parecido una muy buena experiencia en lo laboral, y creo que contamos con grupos de trabajos comprometidos con el mismo fin, generar una atención de calidad y cercanía con el cliente, las herramientas son dinámicas y efectivas en el desarrollo de las labores, solo me queda agradecer el compromiso y voto de confianza de la Compañía. Gracias !!	Apoyo
Es una muy buena oportunidad considerando los tiempos de traslado, el riesgo de salud teniendo a un hijo asmático y no me complica ya que cumplo con mis horarios de conexión y hasta hoy sin dificultades técnicas	Compatibilidad familiar
Ha sido una oportunidad para poder conciliar el trabajo con la familia y aprovecharla al máximo	Compatibilidad familiar
la experiencia en casa a sido excelente para mi, lo único malo son las reuniones fuera de horario, las cuales son han citado pasada las 18:20 horas y se han extendido casi hasta las 20:00 horas, en lo cual ocupan nuestros tiempos de estar en familia con reuniones innecesarias.-	Horario
Igual se extraña el contacto con las personas.....	Comunicación
Nunca pensé que mi tipo de labor se podría hacer de manera remota, ha sido maravilloso, todo lo malo también trae lago bueno y agradezco de haber tenido esta oportunidad, que mis niños me vean a diario acá trabajando y compartiendo con ellos es impagable, por eso estoy muy agradecida.	Apoyo
se echa de menos el contacto con los compañeros de trabajo	Comunicación
me hubiese pertinente si me hubiesen facilitado un computador ya que lo comparto con mi hijo, facilitaría un poco mas las cosas. si los niños pudiesen asistir al colegio el trabajo remoto seria aun mas productivo, podría combinarse trabajo remoto con presencial	Infraestructura
buena experiencia trabajar en forma remota y nos podemos dar cuenta que si se puede ,debería implementarse este sistema en un futuro no muy lejano , así poder tener mas tiempo con la familia y compatibilizar vida personal con el trabajo de una mejor manera .	Compatibilidad familiar
De repente he tenido problemas de internet, pero ha sido muy buena experiencia.	Internet

Tabla 10: Detalle cuantitativo sobre las respuestas y sus etiquetas.

Agradecer a la compañía, a nuestros jefes y Gerentes por la preocupación por nuestro salud, bienestar físico, Mental durante todo este período.	Apoyo
creo que a nuestra empresa nos falta potenciar herramientas de trabajo colaborativo, whiteboard, one drive, office 365	Sistemas
Cumplir mis labores desde el hogar, me ha permitido ahorrarme un tiempo considerable en transporte (En mi caso particular me ahorro alrededor de 2 horas y media), lo que me permite tener una mejor calidad de vida.	Horario
en el contexto de esta crisis sanitaria, convivo con adultos mayores y niños que requieren atención, es por ello que me gustaría que se flexibilizara el horario de colación, para poder atenderlos. obvio que sin crisis sanitaria eso ya estaría resuelto.	Compatibilidad familiar
Al no estar fisicamente junto con el grupo de trabajo, se hace más lenta la comunicación para solucionar casos con problemas de los clientes, seguimiento de las operaciones	Tiempos de respuesta
solo gracias por dejar trabajar remoto, porque ha sido lejos la experiencia familiar que siempre anhele.	Compatibilidad familiar
Creo es mas cómodo ya que uno pierde mucho tiempo en los traslados y es cansador manejar mas de una hora de ida y de vuelta	Horario
Para mí lo ideal sería un modelo mixto entre presencial y remoto.	Operatividad - Procesos
Mejorar los tiempos de espera cuando debemos comunicarnos con Soporte de la Cía. para solucionar los problemas con nuestros equipos.	Tiempos de respuesta
Es importante tener los Recursos básicos para trabajar en casa, una comoda silla, uno pasa muchas horas sentado, la almohadilla de ACHS para proteger la muñeca, a mi se me ha resentido.	Infraestructura
Recomiendo que puedan facilitar las sillas , audifonos, monitor para realizar un mejor trabajo	Infraestructura
Solo que estoy tremendamente agradecido de esta institución, por toda la preocupación y el apoyo prestado soy un funcionario agradecido de trabajar en esta institución...	Apoyo
Mi gran problema es la calidad de internet que puedo acceder en la zona de mi domicilio.	Internet
Teletrabajo debe ir acompañado de entrega de materiales para el trabajo, tanto funcional como de apoyo.	Apoyo

Tabla 11: Detalle cuantitativo sobre las respuestas y sus etiquetas.

4.3. INDICADORES

Se listarán los indicadores que fueron definidos con una figura representando su resultado final. También se expresará si los criterios de aceptación fueron logrados para cada uno de ellos.

4.3.1. GRÁFICO DE BARRAS: RÁNKING POR CATEGORÍA

Este gráfico de barras busca representar la cantidad de respuestas que fueron categorizadas con cierta etiqueta. Es una forma rápida de demostrar qué categorías son más mencionadas en las respuestas.

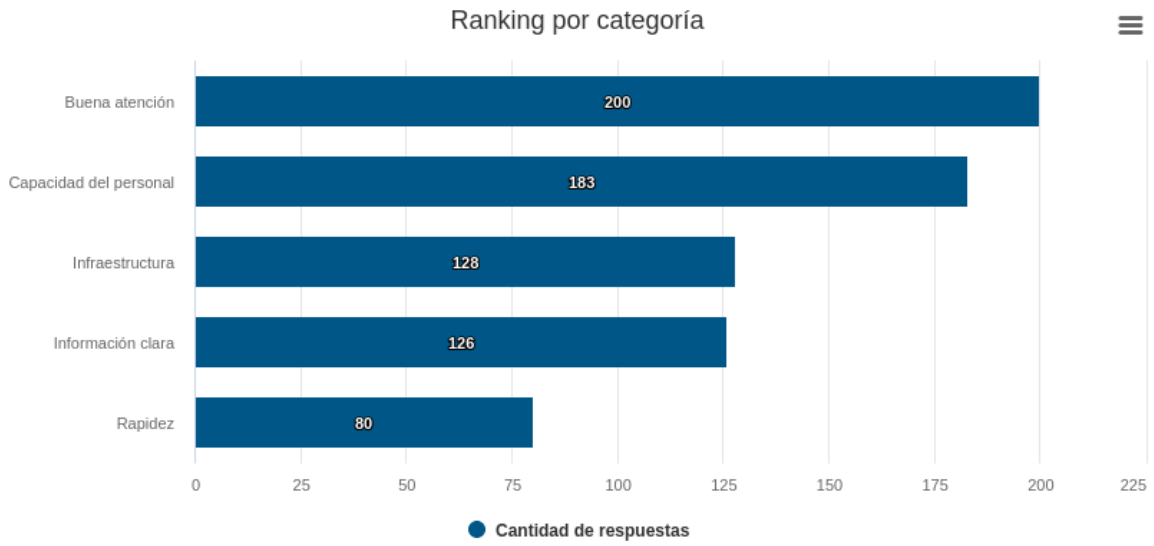


Figura 21: Ejemplo del gráfico, que muestra la frecuencia para cada categoría.

El funcionamiento de este indicador es bastante simple, y se puede apreciar en la Figura 21 que las categorías están ordenadas de mayor a menor, indicando su frecuencia. Con esto, sabemos cuántos comentarios están con cada etiqueta.

Listando los criterios de aceptación, tenemos:

1. Presenta los datos de forma correcta, sin errores: Cumple con el criterio, mostrando la frecuencia para cada categoría en la muestra.
2. Las categorías pertenecen al estudio: Pertenecen al estudio de clientes del área financiera.
3. Las barras de categorías están ordenadas de mayor a menor: Se cumple, tal cual como se observa en la figura.
4. El indicador posee una carga menor de 3[s]: Implementado este indicador en el servidor de pruebas, la carga del indicador fue de 1,8[s], siendo luego de 0,9[s] tras utilizar el caché del cliente.

4.3.2. GRÁFICO DE BARRAS: CATEGORÍA VS SENTIMIENTO

Este gráfico busca representar cómo son los sentimientos de los comentarios agrupados bajo una misma categoría.

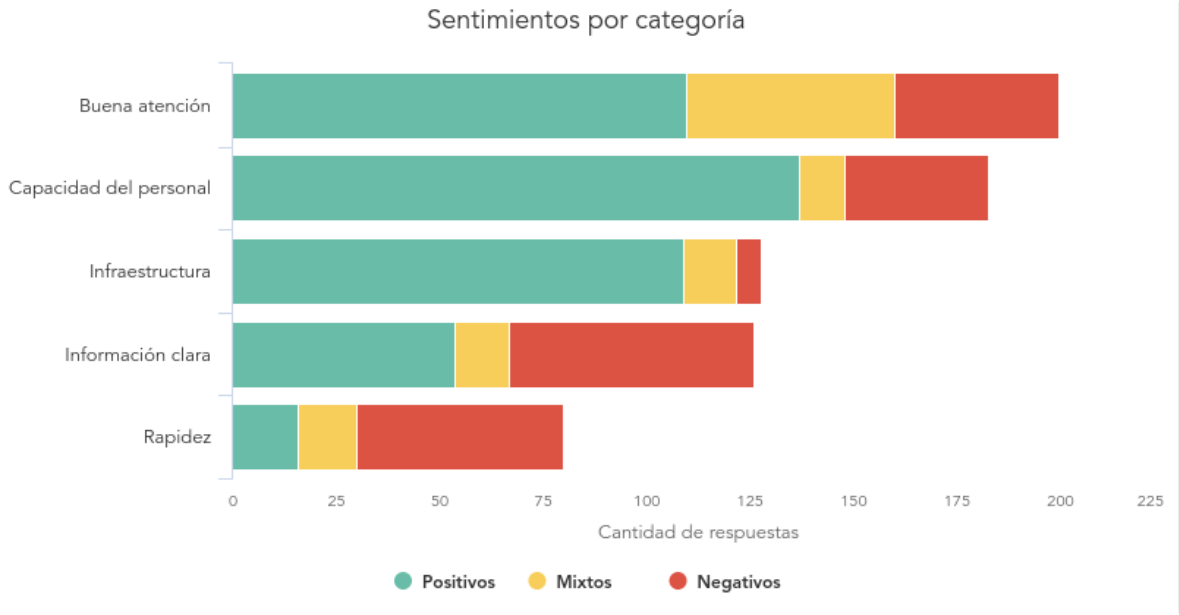


Figura 22: Ejemplo de gráfico, que presenta los sentimientos de cinco categorías.

Podemos apreciar que el gráfico de la Figura 22 posee un eje X que se adapta a la cantidad de datos presentes. Además, cada barra se encuentra correctamente stackeada con un color específico, en donde cada uno está descrito en la leyenda. Listando los criterios de aceptación, tenemos:

1. Presenta los datos de forma correcta, sin errores: Cumple este criterio, siendo la suma de cada sentimiento igual al total presentado en el eje X con el largo de cada barra.
2. La frecuencia de los sentimientos es correcta para cada categoría: Cada sentimiento presenta frecuencias correctas.
3. El orden de la barra particionada es correcta: Como se aprecia la base corresponde a los positivos, luego neutros y finalmente negativos.
4. Las barras de categorías están ordenadas de mayor a menor: Están ordenadas de mayor a menor, desde arriba a abajo, considerando la frecuencia total de respuestas por categoría.
5. El indicador posee una carga menor de 3[s]: Implementado este indicador en el servidor de pruebas, la carga del indicador fue de 1,8[s], siendo luego de 1[s] tras utilizar el caché en el cliente.

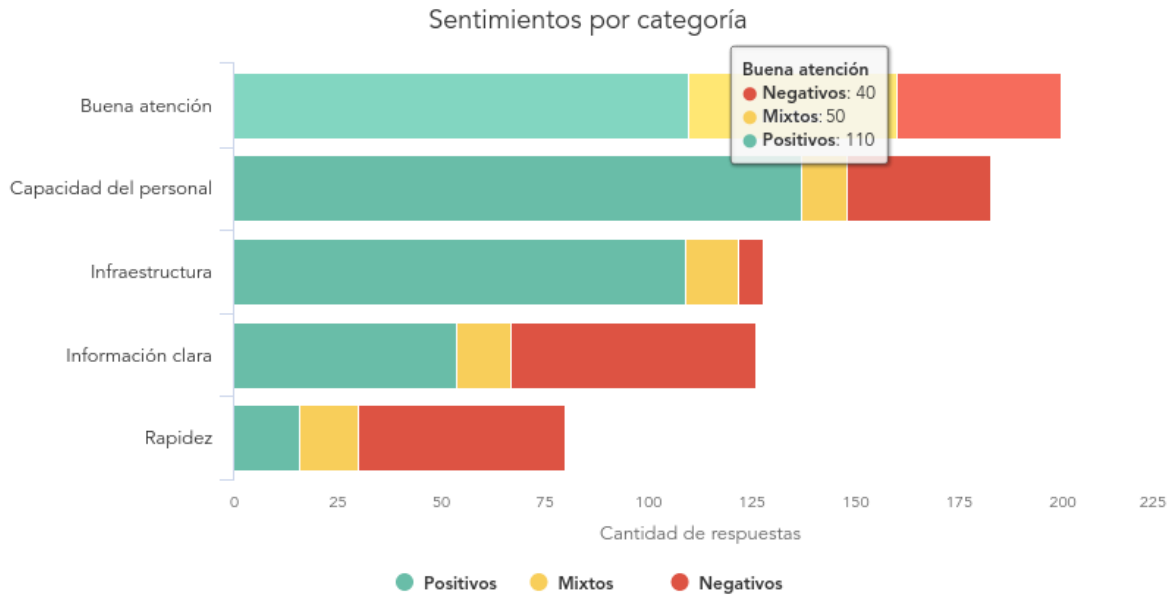


Figura 23: Formato del tooltip para cada barra, indicando las cantidades para cada sentimiento.

4.3.3. NUBE DE PALABRAS MEJORADA

Este gráfico busca mostrar información importante respecto a tendencias en lo que la gente responde, mostrando la frecuencia de unigramas y bigramas.

Figura 24: Ejemplo de gráfico que muestra los unigramas y bigramas más frecuentes.

Los bigramas poseen un tamaño más grande, permitiendo que el usuario se percate fácilmente de ellos. En la Figura 24, podemos apreciar que el unigrama *atención* es frecuente, pero gracias a los bigramas podemos ver que las opiniones son diversas y de contextos específicos.

Listando los criterios de aceptación, tenemos:

1. Presenta los datos de ejemplo y reales de forma correcta, sin errores: Cumple este criterio, mostrando la frecuencia de cada palabra en el tooltip, correspondiente a las respuesta de salud que fueron utilizadas.
2. Los bigramas destacan a pesar de su poca frecuencia: Los bigramas, debido a su importancia semántica, se muestran con un tamaño considerable en comparación con los unigramas que poseen frecuencia parecida.
3. Las palabras tienen buena ortografía: Las palabras poseen buena ortografía y correcto uso de tildes.
4. El indicador posee una carga menor de 3[s]: Implementado este indicador en el servidor de pruebas, la carga de este fue de 1,1[s], siendo luego de 0,3[s] tras utilizar el caché en el cliente.

4.3.4. TREEMAP

Este gráfico busca representar de forma más didáctica e intuitiva la categorización de comentarios y sus palabras claves más frecuentes.

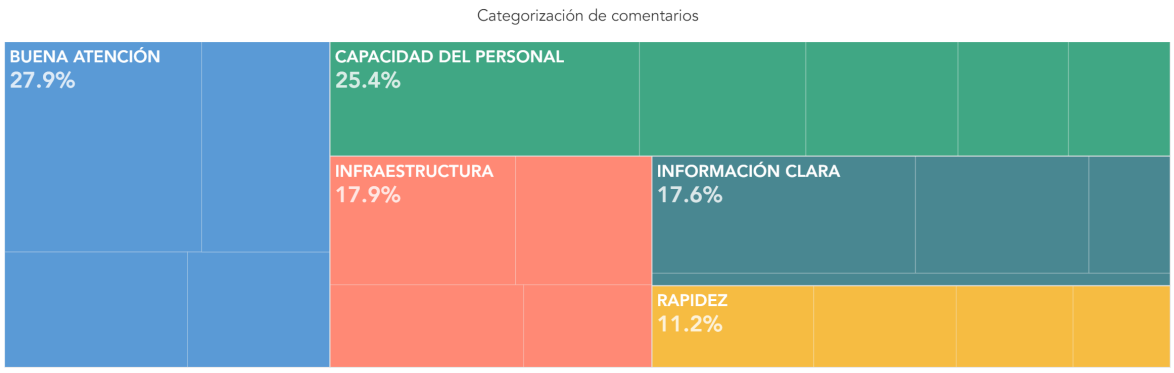


Figura 25: Ejemplo de gráfico de treemap representando las categorías, su color y porcentaje respecto al total de respuestas.

Se aprecia la forma de cuadrilátero que presenta el gráfico de la Figura 25. Además, cada uno está segmentado en pequeños cuadriláteros, haciendo entender que estos se pueden desglosar para encontrar más información. Cada categoría posee su color propio para diferenciarse, acompañados de un porcentaje que implica cuánto poseen respecto al total de comentarios. Podemos apreciar que al hacer click en uno de los cuadriláteros, es posible observar las palabras claves más frecuentes en cada una.

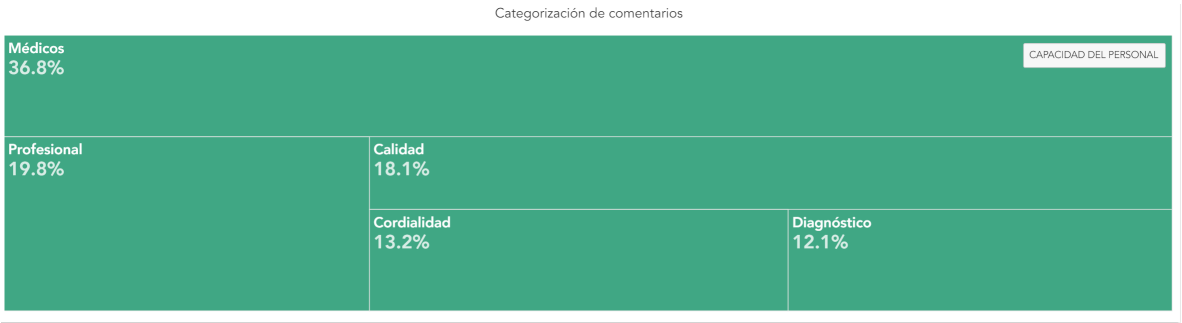


Figura 26: Ejemplo de gráfico de treemap cuando hacemos click en la categoría de calidad del personal.

Listando los criterios de aceptación, tenemos:

1. Presenta los datos de forma correcta, sin errores: Cumple este criterio, mostrando de forma transparente los porcentajes y frecuencias para cada categoría y palabra clave.

2. Al hacer click en una figura en el primer nivel, lleva al detalle de sus palabras claves: Cumple el criterio, donde al hacer click podemos apreciar las palabras claves. Además, mantienen el mismo color que su categoría.

3. Al hacer click en una figura en el segundo nivel, debe volver al nivel superior: Se replica este comportamiento. Además, en caso de no ser intuitivo, en la esquina superior derecha se presenta un botón con el nombre de la categoría, que también lleva al nivel superior.

4. El indicador posee una carga menor de 3[s]: El indicador es cargado en 1,3[s], para luego con uso de caché tener un tiempo de carga de 0.2[s].

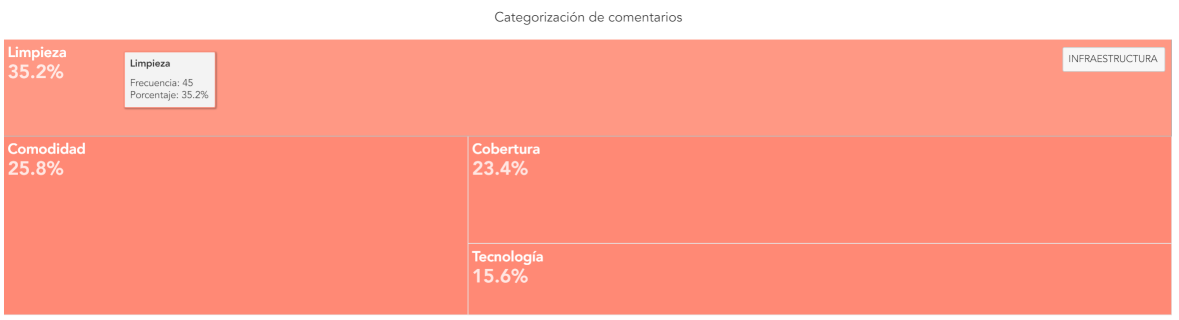


Figura 27: Formato del tooltip que, además del porcentaje, indica la frecuencia para la categoría o keyword, en este caso, una palabra clave para la categoría Infraestructura.

4.3.5. GRÁFICO DE RECOMENDACIÓN/SENTIMIENTO VS CATEGORÍA

Este gráfico busca representar en un indicador las categorías y sus sentimientos o recomendaciones, indicando otras métricas como frecuencia total, relativa, satisfacción neta, etc.

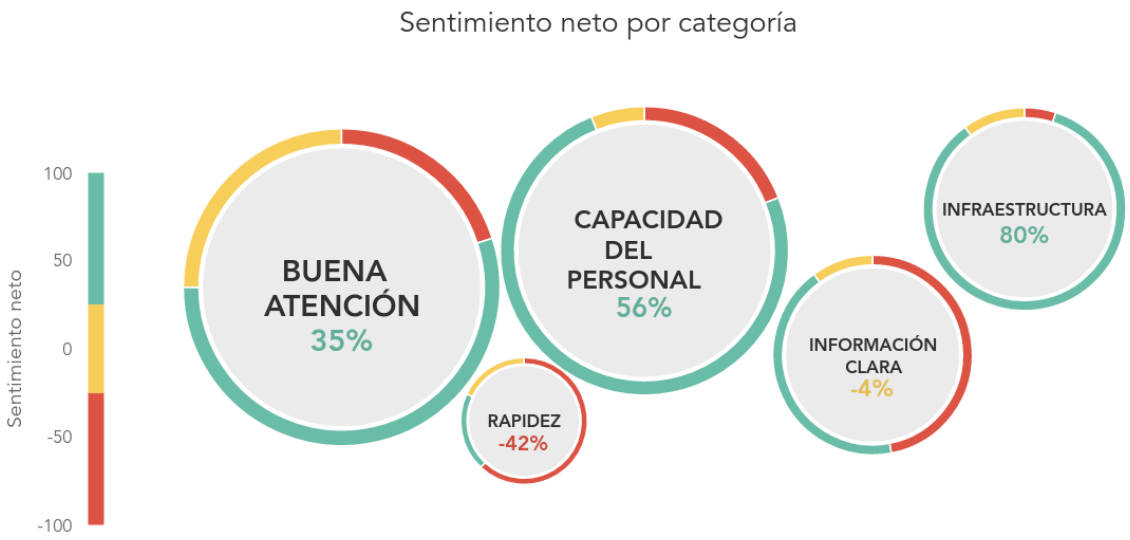


Figura 28: Ejemplo de gráfico de sentimiento vs categoría, donde cada burbuja representa la información completa de cada categoría.

Podemos apreciar que el gráfico de la Figura 28 posee una gran cantidad de información en cuanto a:

- El tamaño de la burbuja indica la cantidad de comentarios que fueron etiquetados para cada categoría.
- El borde de cada círculo posee tres colores, donde la porción de cada perímetro corresponde a la cantidad de comentarios con un sentimiento específico o según la representación de recomendación. Se mantienen los mismos colores que el gráfico de categoría por sentimiento para la consistencia.
- El porcentaje corresponde al sentimiento/recomendación neta y su color tiene relación con su valor y el eje dependiente.
- El eje Y posee tres porciones de colores y va de un rango de -100 a 100. El centro del círculo se posiciona respecto al eje Y según su valor de satisfacción neta.

Considerando los criterios de aceptación para este indicador, se tiene:

- Presenta los datos de forma correcta, sin errores: Cumple con este criterio, siendo proporcional los porcentajes netos y específicos de la satisfacción, como también la relación de los tamaños de los círculos. Además, los círculos están correctamente alineados según su satisfacción neta con el eje Y.
- Los porcentajes para la satisfacción neta o sentimiento son los correctos: Cumple que los porcentajes específicos y netos tienen relación entre sí.

- Los tamaños de los círculos corresponden a la cantidad total de respuestas: Se cumple este criterio, siendo muy notoria la diferencia de tamaños de los círculos indicando qué categorías poseen más o menos comentarios que otra.
- Ninguno de los círculos se solapan en el canvas: Como se muestra en el gráfico, ningún círculo se solapa entre sí.
- El centro del círculo está a la misma altura que su valor de satisfacción neta y el eje dependiente: Correcto, se aprecia que los círculos están centrados en el eje dependiente.
- El indicador posee una carga menor de 3[s]: Cumple, ya que el indicador se demora 1,6[s] en cargar, mientras que al estar en caché se demora 0,2[s] en su carga.

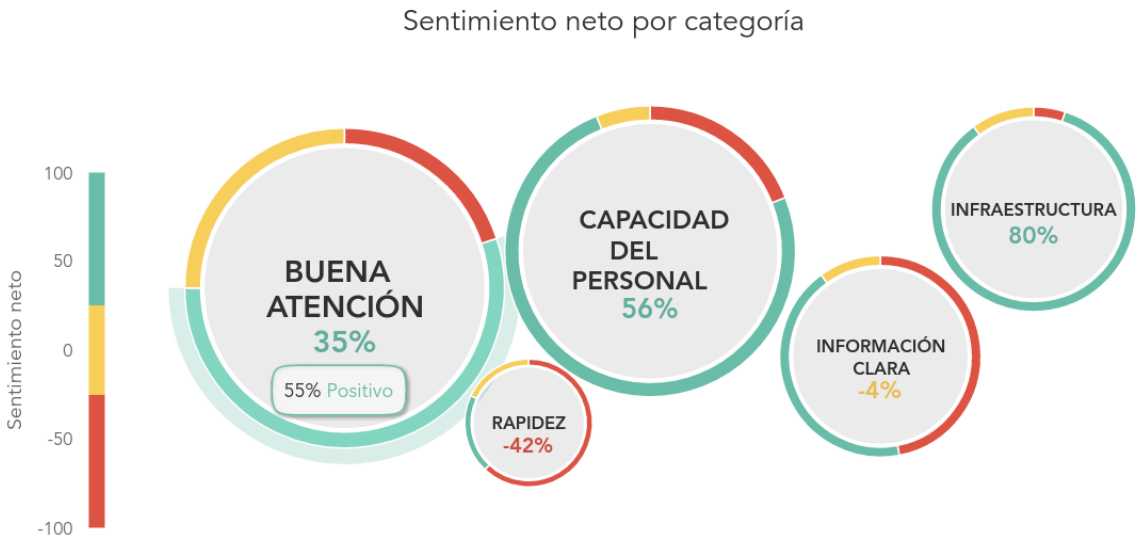


Figura 29: Formato del tooltip al colocar el mouse encima de una porción de borde, indicando el porcentaje de comentarios positivos.

4.3.6. TABLA DE COMENTARIOS

Este gráfico busca mostrar directamente los comentarios indicando la categoría a la que pertenecen. Además, comentarios que sean muy parecidos entre sí serán agrupados. Con esto, la tabla debe poseer una columna con la *cuenta*, indicando la cantidad de comentarios que fueron agrupados en esa fila.

Tabla de comentarios y literales

Servicio	Cuenta	
Sistemas	**10**	–
No cuento con PC Corporativo, por lo cual tengo que realizar las reuniones por Teams fuera de conexión remota, por lo que los archivos y presentacion de informacion debo coordinarme en enviarlo ami correo personal y poder acceder a ellos desde allí.	1	
No tener un equipo personal a mi disposición que me permita recibir llamados, derivarlos, etc., la conexión a VPN se interrumpe y largas esperas para hablar con Soporte.	1	
En un par de ocasiones era difícil conectarme al escritorio remoto debido a la conexión o problemas en la oficina	1	
Durante las mañanas, por lo general, se queda pegadísimo VPN, en las tardes funciona mejor	1	
Problemas con la VPN. Recepción de demasiados mails en horarios extra laborales.	1	
solo que se pierde muy a menudo la conexion con escritorio remoto	1	
Dificultad de Soporte técnico sobre todo al inicio	1	
solo algunos problemas con la caída de sistemas	1	
La inestabilidad de la red VPN al comienzo	1	
el servicio de cisco es intermitente	1	
Infraestructura	**9**	+
Compatibilidad familiar	**8**	+
Comunicación	**7**	+
Internet	**4**	–
El teléfono pierde la señal, los clientes me escuchan mal y muchas veces no me entienden claramente	1	
No siempre, pero hay dificultades tecnicas, por ejemplo, el internet es lento.	1	
tuve problemas de internet al principio solamente	1	
internet deficiente solucionado	1	
Operatividad - Procesos	**2**	–
he tenido dificultades con los documentos físicos que envían los clientes.	1	
mi equipo es muy lento para relizar mi travajo	1	
Horario	**1**	+

Figura 30: Tabla de comentarios, mostrando las respuestas, su categoría y frecuencia.

Se especifica correctamente el enunciado de la pregunta como las categorías y también sus comentarios. Se utilizan tonalidades de grises, un color blanco y uso de negrita para diferenciar los niveles de la tabla.

Listando los criterios de aceptación, tenemos:

- Presenta los datos de ejemplo y reales de forma correcta, sin errores: Las presenta de forma correcta, con comentarios acorde a su categoría, además de la frecuencia.
- Las categorías pertenecen al estudio: Pertenecen, y se comprueban con los datos mostrados en los otros indicadores.
- La frecuencia de los grupos de categoría son correctos: Son los correctos, además que se contrasta con la cantidad de comentarios iniciales con los finales en la tabla.

- Las pestañas de categorías están ordenadas de mayor a menor: Se aprecia que va desde la categoría más frecuente hasta la que menos posee.
- El indicador posee una carga menor de 3[s]: El indicador de demora 1.2[s] en cargar, mientras que al estar en caché se demora 0,3[s] en su carga.

4.4. QSERVUS AI

Para validar el funcionamiento de la API se realiza una prueba del procedimiento completo para la obtención de todos los indicadores implementados. Se detallan los diversos formatos, cómo se realizan las llamadas y las respuestas recibidas por el servidor. Para organizar el proceso, lo haremos paso a paso tal como se define en el siguiente listado:

1. *api/analysis/analyse_text*: Consiste en el proceso de etiquetado tanto para categoría como sentimiento.
2. *api/analysis/group_text*: Consiste en combinar respuestas similares, para poder mostrar de forma ordenada el indicador de tabla de comentarios.
3. *api/charts/category_ranking*: Se obtiene el gráfico de ranking de categorías.
4. *api/charts/wordcloud*: Se obtiene el gráfico de nube mejorada.
5. *api/charts/sentiment_bar*: Se obtiene el gráfico de sentimiento por categoría.
6. *api/charts/treemap*: Se obtiene el gráfico de caja de resonancia, o treemap.
7. *api/charts/comment_table*: Se obtiene el gráfico de tabla de comentarios.
8. *api/charts/bubble_chart*: Se obtiene el gráfico de burbujas para ambos casos.

Se utilizarán las respuestas de una pregunta con el enunciado ”¿Cuál es tu evaluación del teletrabajo y cómo podría mejorarse?”. En total son 450 respuestas, y serán categorizadas con el modelo de categorización de teletrabajo definido en la validación de la categorización de texto.

Por último, cierta información en la entrega o respuesta de los endpoints será reemplazada por tres guiones. La razón es que la información corresponde a respuestas reales de clientes de QServus, y se busca proteger la identidad de la empresa como de las personas que responden.

4.4.1. CATEGORIZACIÓN DE DATOS

Se preparan las respuestas para el formato de entrada que pide el endpoint de *analyze_text*. En el siguiente cuadro se presenta la información que fue enviada.

```
POST /api/analysis/analyse_text/

{
    "metadata": {
        "su_survey_id": 85,
        "su_survey_tpl": 52,
        "company_id": 7,
        "model_version": 2,
        "model_name": "teletrabajo_---",
    },
    "answers": [{
        "question": {
            "su_question_id": 2265,
            "su_question_tpl_id": 2884,
            "su_question_title": ,
            "su_question_type": 32,
        },
        "answer": {
            "su_answer_id": 111544,
            "su_answer_scale": 1,
            "su_answer_text": "que el horario remoto sea el mismo
                               que en oficina",
            "su_answer_datetime": "2020-11-26 21:49:07.404232+00",
        }
    },...]
}
```

Al ser 450 preguntas, no se muestra el contenido completo de los datos de entrada, pero se muestra el ejemplo con la primera respuesta. Los tres puntos suspensivos al final del cuadro indican que existen más respuestas.

La respuesta obtenida por el endpoint al ingresar, fue la siguiente:

```
{
    "status": "Success",
    "message": "Everything is correct, the answers are processing",
    "answers_count": 450,
    "questions_count": 1
}
```

La información que retorna el endpoint consiste en una variable que indica que el proceso fue realizado correctamente, con un mensaje adjunto. Además, señala la cantidad de respuestas y preguntas involucradas.

Debido a la gran cantidad de respuestas que puedan ser enviadas en una sola llamada, este endpoint revisa si el formato de entrada es correcto y de forma asíncrona activa la tarea de categorización. Así, el cliente recibe de forma rápida una respuesta correcta o incorrecta. Gracias al uso de webhooks, la API notificará al cliente cuándo el procesamiento esté terminado, y pueda solicitar la información mediante los indicadores.

4.4.2. AGRUPACIÓN DE TEXTOS

Posterior a la categorización de las respuestas es posible realizar la agrupación de los comentarios. Como las respuestas y toda la información necesaria ya fue almacenada en la API, se debe enviar solamente el identificador de la pregunta que queremos que sus respuestas, en caso de parecerse, se agrupen.

```
POST /api/analysis/group_text/

{
    "qid": 2265,
}
```

Este endpoint responde apenas termine el procesamiento, ya que el proceso es más rápido que la categorización de texto y el *payload* es mucho más liviano. Tras eso, responde con un mensaje indicando que fue procesado correctamente.

```
POST /api/analysis/group_text/

{
    "status": "Success",
    "message": "All the answers were grouped successfully"
}
```

4.4.3. GRÁFICO DE RANKING DE CATEGORÍAS

Tras el procesamiento de las respuestas, ya es posible obtener los indicadores. Como primer indicador, se obtiene el de ranking de categorías, el cual se logra gracias al endpoint correspondiente. Debemos, además, entregar el identificador de la pregunta cuyas respuestas queremos que se reflejen en el gráfico.

```
GET /api/charts/category_ranking/

{
    "qid": 2265,
}
```

Tras finalizar, se recibe un JSON con el formato específico, que contiene toda la información para que las plataformas de QServus puedan mostrar perfectamente los indicadores. Todos los indicadores devolverán ese tipo de formato.

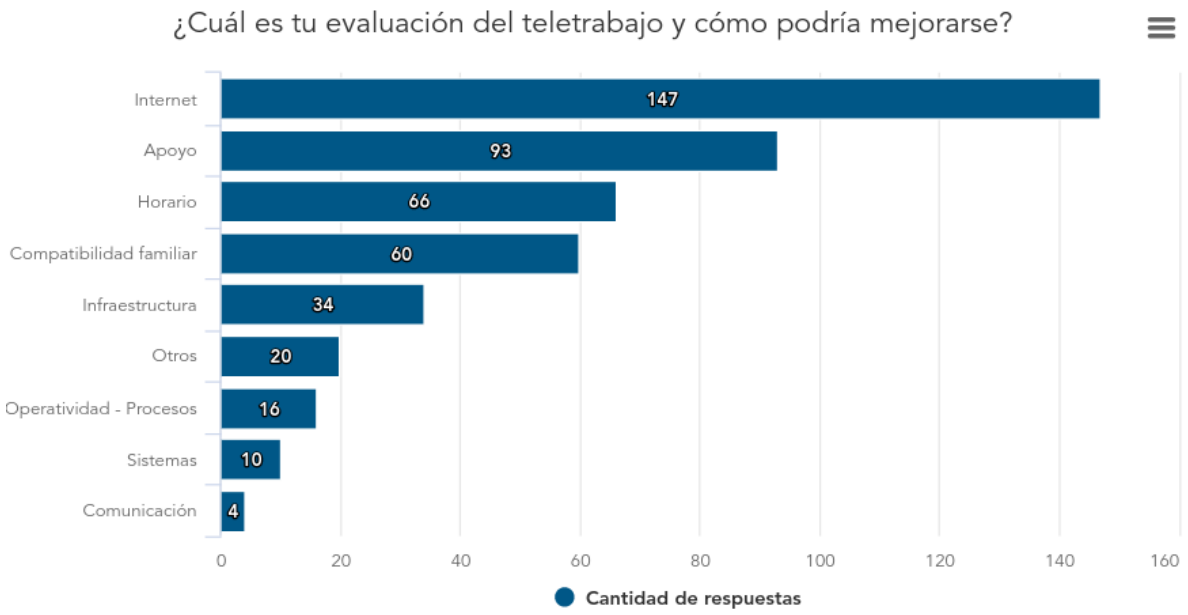


Figura 31: Gráfico de ranking de categorías.

Podemos apreciar que la categoría que más se menciona es la de Internet, seguido de Apoyo, Horarios y Compatibilidad Familiar, siendo los más importantes. Finalmente, los sistemas de la empresa y la comunicación entre trabajadores no parece ser un tema determinante en la evaluación.

4.4.4. GRÁFICO DE NUBE MEJORADA

Ahora, veremos cuáles son los unigramas y bigramas que más destacan en el conjunto global de respuestas. Para eso se envía el identificador de la pregunta al endpoint correspondiente.

```
GET /api/charts/wordcloud/

{
    "qid": 2265,
}
```

Tras realizar la petición al endpoint, devuelve el JSON que al renderizar, genera el gráfico de la Figura 32.

Figura 32: Gráfico de nube mejorada.

Al ver el indicador, podemos apreciar que los temas más frecuentes son la nueva modalidad de trabajo y el contexto que éste representa en los horarios laborales.

4.4.5. GRÁFICO DE BARRA: SENTIMIENTO POR CATEGORÍA

Para este indicador solo se necesita el identificador de la pregunta, apuntando al endpoint que le corresponde.

```
GET /api/charts/sentiment_bar/

{
    "qid": 2265,
}
```

Tras enviar la petición, QServus AI retorna el JSON con la información del indicador, el cual al ser renderizado en QServus genera el gráfico de la Figura 33:

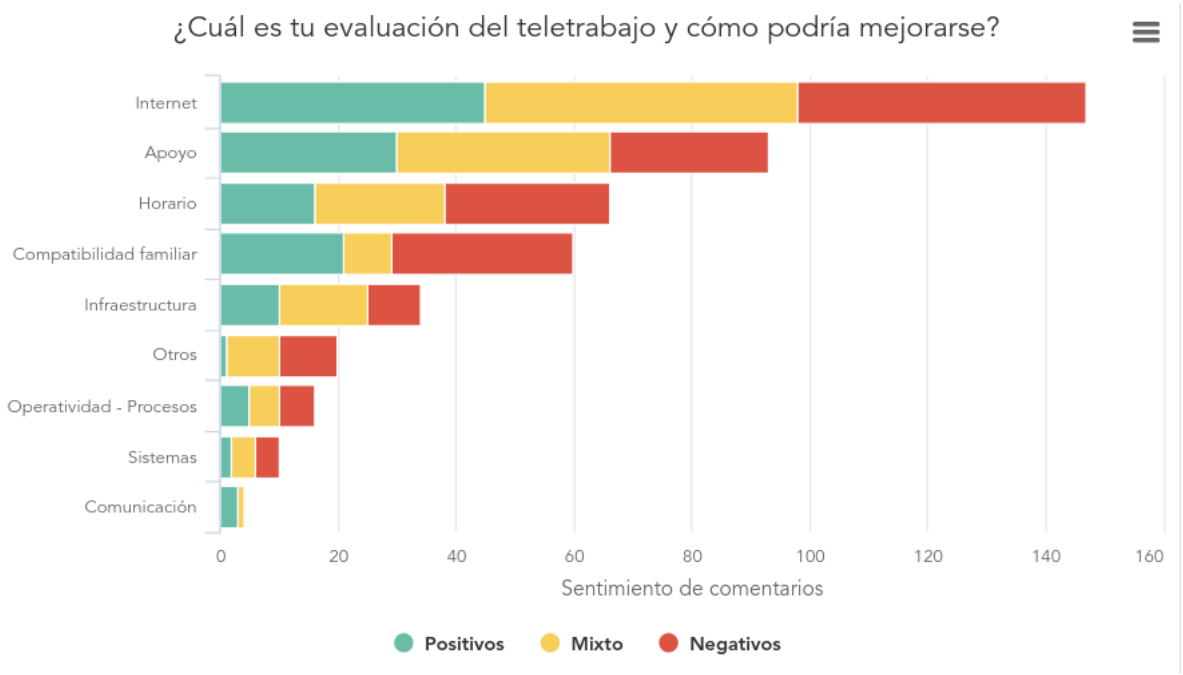


Figura 33: Gráfico de barras: sentimiento por categoría.

Al observar el gráfico de la Figura 33, es evidente que la cantidad de comentarios mixtos y negativos es superior a los positivos para la mayoría de las categorías. Un mayor detalle de las cantidades de respuestas por cada tipo de sentimiento, se obtiene a través de Tabla 12.

Categoría	Negativos	Mixtos	Positivos	Total
Internet	49	53	45	147
Apoyo	27	36	30	93
Horario	28	22	16	66
Compatibilidad Familiar	31	8	21	60
Infraestructura	9	15	10	34
Otros	10	9	1	20
Operatividad - Procesos	6	5	5	16
Sistemas	4	4	2	10
Comunicación	0	1	3	4
Total	164	153	133	450

Tabla 12: Detalle cuantitativo sobre las respuestas y sus etiquetas.

4.4.6. GRÁFICO DE TREEMAP

Para la obtención de este indicador, se debe enviar al endpoint correspondiente el identificador de la pregunta que se desea graficar.

```
GET /api/charts/treemap/

{
    "qid": 2265,
}
```

Tras obtener la respuesta, se renderiza el contenido y obtenemos el gráfico de la Figura 33.

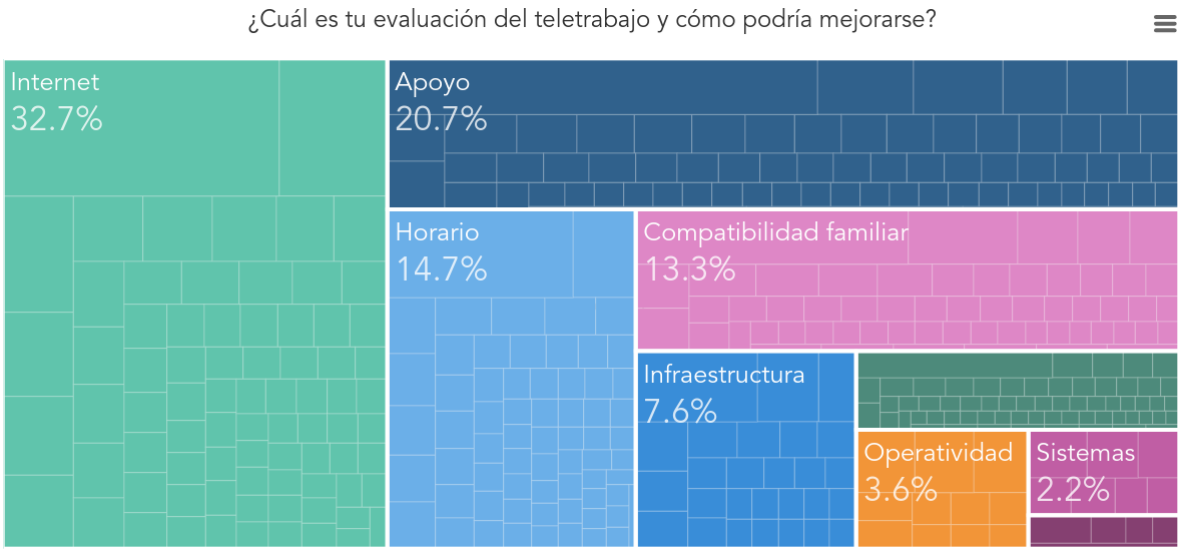


Figura 34: Gráfico treemap de las respuestas.

En gráfico de Figura 34, se aprecia la cantidad de palabras claves que posee cada categoría, como también su proporción respecto al total. Es posible realizar desgloses para apreciar las palabras claves de cada categoría (ver Figura 35).

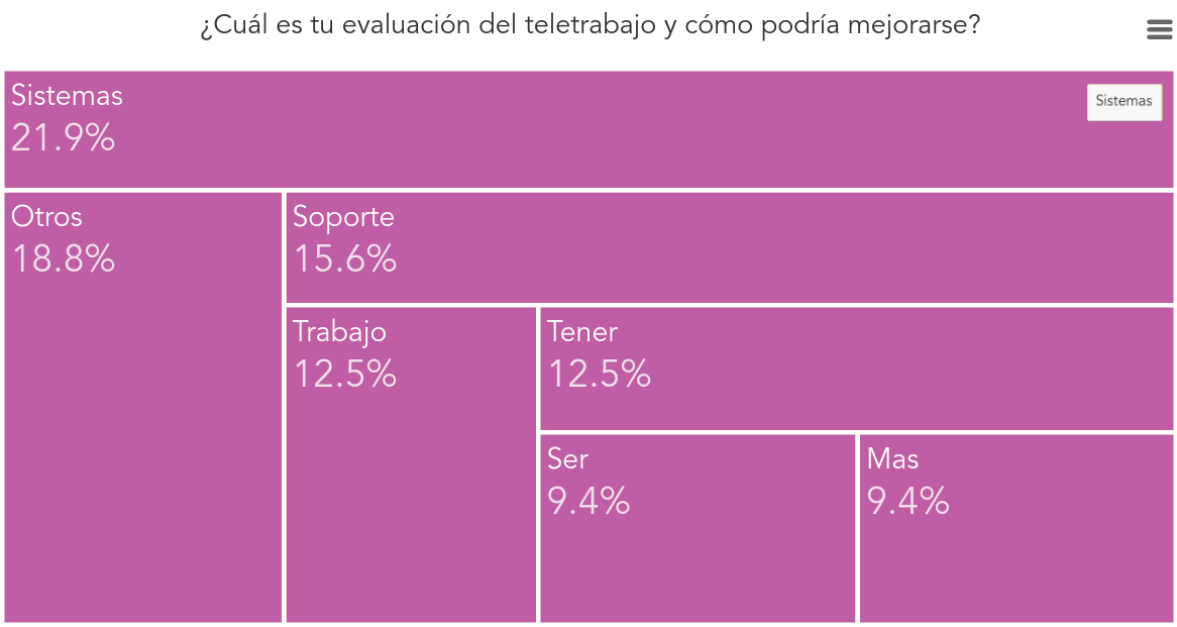


Figura 35: Desglose de la categoría sistemas.

4.4.7. GRÁFICO DE TABLA DE COMENTARIOS

Este gráfico permitirá ver de mejor forma los comentarios que fueron etiquetados tras el procesamiento. Se debe llamar a su endpoint correspondiente, dando también el identificador de la pregunta.

```
GET /api/charts/comment_table/

{
    "qid": 2265,
}
```

Esto devuelve un JSON con la información de cada respuesta y su categoría. Además, si hay mensajes que se repitieron dentro del conjunto, serán agrupados, indicado la frecuencia de cuantas respuestas representan.

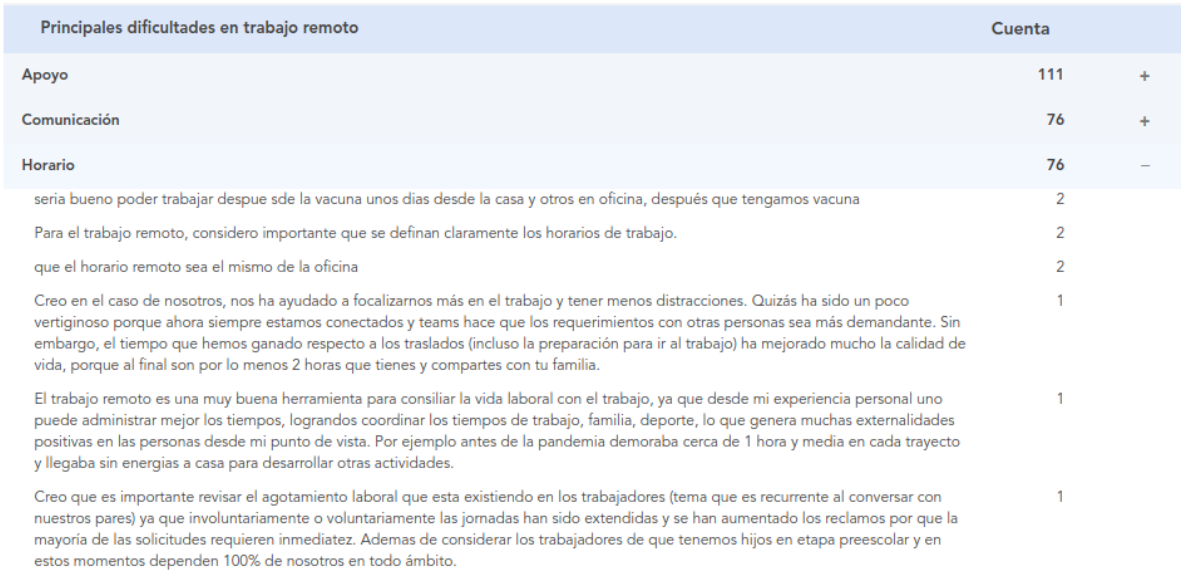
Principales dificultades en trabajo remoto	Cuenta	
Apoyo	111	+
Comunicación	76	+
Horario	76	+
Compatibilidad familiar	65	+
Otros	32	+
Infraestructura	31	+
Operatividad - Procesos	25	+
Sistemas	18	+
Internet	16	+

Figura 36: Vista sin desglosar de la tabla con las categorías encontradas.

En la Figura 36, cada categoría posee una cuenta, donde vemos la cantidad de respuestas que fueron categorizadas. Al hacer click en los signos +, podemos ver el detalle dentro de una etiqueta.

Principales dificultades en trabajo remoto	Cuenta	
Apoyo	111	–
Agradecer por los esfuerzos de todos.	5	
Solo, que estoy agradecida por el constante apoyo y preocupacion de la empresa , en la nueva modalidad de trabajo remoto, en mi caso personal me acomoda mucho .	2	
Agradecer la posibilidad que me entregó [redacted] de trabajar a distancia sin exponerme ni exponer a mi familia.	2	
Agradecido enormemente por todo el apoyo brindado por el [redacted] en el bienestar integral de cada colaborador.	2	
Se agradece el tremendo esfuerzo que han hecho como [redacted] para cuidar la salud de todos sus trabajadores.	2	
solamente agradecimiento por todo el apoyo del trabajo en estos tiempos que han sido difíciles.	2	
Mi Jefatura me ha dado todas las herramientas para trabajar con tranquilidad desde mi casa.	2	
No, creo que esta todo claro.	2	
el apoyo recibido de las jefaturas en este periodo, personalmente ha sido excelente. Ojala pudieran hacer notar eso. Por otro lado ,comparando con mis cercanos que están en esta modalidad, nosotros somos privilegiados, tanto en la preocupación del grupo como de las jefaturas directa. Lo único malo para mi ,es que no cuento con las instalaciones físicas y de conectividad para poder realizar mejor mi trabajo.	1	
Si. Esta evolución de trabajo en remoto necesita de un cambio cultural que permita el empoderamiento, el autoaprendizaje, el conciliar el propósito de la empresa con el propósito personal, el que nuestros lideres nos inspiren. Creo que es una paso excepcional el que hemos dado y me siento orgullosa de trabajar en una gran compañía. Lo que comento va en la línea de que nos apoyemos en ser aun mejores. :)	1	

Figura 37: Desglose de la categoría apoyo.



Principales dificultades en trabajo remoto	Cuenta	
Apoyo	111	+
Comunicación	76	+
Horario	76	–
seria bueno poder trabajar despue sde la vacuna unos dias desde la casa y otros en oficina, después que tengamos vacuna	2	
Para el trabajo remoto, considero importante que se definan claramente los horarios de trabajo.	2	
que el horario remoto sea el mismo de la oficina	2	
Creo en el caso de nosotros, nos ha ayudado a focalizarnos más en el trabajo y tener menos distracciones. Quizás ha sido un poco vertiginoso porque ahora siempre estamos conectados y teams hace que los requerimientos con otras personas sea más demandante. Sin embargo, el tiempo que hemos ganado respecto a los traslados (incluso la preparación para ir al trabajo) ha mejorado mucho la calidad de vida, porque al final son por lo menos 2 horas que tienes y compartes con tu familia.	1	
El trabajo remoto es una muy buena herramienta para consiliar la vida laboral con el trabajo, ya que desde mi experiencia personal uno puede administrar mejor los tiempos, logrando coordinar los tiempos de trabajo, familia, deporte, lo que genera muchas externalidades positivas en las personas desde mi punto de vista. Por ejemplo antes de la pandemia demoraba cerca de 1 hora y media en cada trayecto y llegaba sin energias a casa para desarrollar otras actividades.	1	
Creo que es importante revisar el agotamiento laboral que esta existiendo en los trabajadores (tema que es recurrente al conversar con nuestros pares) ya que involuntariamente o voluntariamente las jornadas han sido extendidas y se han aumentado los reclamos por que la mayoría de las solicitudes requieren inmediatez. Ademas de considerar los trabajadores de que tenemos hijos en etapa preescolar y en estos momentos dependen 100% de nosotros en todo ámbito.	1	

Figura 38: Desglose de la categoría horario.

4.4.8. GRÁFICO DE BURBUJA

Tras el procesamiento de las respuestas, ya es posible obtener los indicadores. Como prime-rindicador, obtendremos el de burbuja, el cual se obtiene entregando en el endpoint corres-pondiente el identificador de la pregunta cuyas respuestas se desea queden reflejadas en el gráfico..

```
GET /api/charts/bubble_chart/

{
    "qid": 2265,
}
```

El endpoint devuelve un JSON con un formato específico, que contiene toda la información para que las plataformas de QServus puedan mostrar perfectamente los indicadores.

Figura 39: Gráfico de burbujas de las respuestas.

En Figura 39 se aprecia que la mayor parte de los encuestados opina que Internet es lo que podría mejorarse, además que hay opiniones muy divididas ya que las respuestas pertenecen a varias categorías, donde todas, a excepción de Comunicación, poseen respuestas negativas.

CAPÍTULO 5
CONCLUSIONES

5.1. APRENDIZAJE DEL TRABAJO REALIZADO

Se realiza la construcción de un software (API) considerando el contexto de desarrollo de una empresa específica, colaborando a que pueda comunicarse y ofrecer los diversos servicios de forma estructurada, simple y rápida. Además, dentro de las funcionalidades del software, se implementan conocimientos generales de *machine learning*, por una parte con el análisis de sentimiento aplicando un modelo supervisado como las redes neuronales recurrentes y, por la otra, la categorización de texto utilizando un modelo no supervisado con asignación de puntajes y distancias entre vectores con palabras claves. Además, se aprende conceptos tanto de programación y diseño para la construcción de indicadores, los cuales permiten mostrar de forma correcta la información obtenida tras el análisis. Se documenta de forma clara el funcionamiento de los endpoints del software y su uso, como también las pruebas donde se específica el flujo completo para obtener todas las funcionalidades que QServus AI puede ofrecer. Por último, este trabajo permitió implementar los conocimientos adquiridos en la universidad en un contexto real, teniendo que considerar no sólo el desarrollo de la solución, si no adaptarse a requerimientos no funcionales y también de clientes.

5.2. CUMPLIMIENTO DE OBJETIVOS

En síntesis, en este trabajo se construyó una API que logra analizar las respuestas de texto abierto asignándoles un sentimiento y una categoría. Además, presenta diversos indicadores que muestra la información de forma clara y concisa.

Respecto al primer objetivo, se logra desarrollar un modelo de red neuronal para la predicción de sentimientos en respuestas de texto abierto. A pesar de la limitante de no poseer datos con una etiqueta directa con sentimiento, se logra encontrar una alternativa que es la métrica de recomendación, que facilita el entrenamiento de la red neuronal. Tras ese procedimiento, el modelo logró una precisión en conjunto de validación de un 60 %, ofreciendo un desempeño aceptable. Cabe destacar que la limitante del modelo son los casos particulares, donde la nota asignada no corresponde a la justificación en sí, es decir, notas muy buenas con justificación que critica al servicio.

Con el segundo objetivo, se logra implementar un modelo no supervisado para la categorización de texto. A pesar que se utilizó un modelo de categorías que puede parecer general, debido a que está focalizado en un contexto de pandemia que ha afectado a todos, aún así sigue siendo un modelo que se nutrió con respuestas que no cubre la totalidad de empresas con las que QServus trabaja. Además, no siempre las respuestas de texto son tan generales,

por lo que la importancia del contexto de la pregunta ayuda a que la categorización con un modelo sea lo más enfocada posible. Mientras más respuestas se tengan de un mismo contexto, se logra encontrar encontrar mejores palabras claves, y las distancias entre las palabras serán más justificadas, potenciando el desempeño.

Se cumple a cabalidad el tercer objetivo, creando diversos indicadores que representan de forma correcta los resultados. Se crearon seis indicadores, los cuales consideran de forma individual o conjunta las métricas de categoría y sentimiento. Dentro de los que consideran solo categoría están el gráfico de ranking, tabla de comentarios y treemaps, los cuales mostraron en la validación ser de gran utilidad, mostrando en forma simple los datos, ayudando a identificar la presecia y frecuencia de las categorías en las respuestas. Los gráficos que combinan ambas métricas, como el gráfico de categoría vs sentimiento y el gráfico de burbujas, presentan las categorías con un contexto más completo, mostrando la intención de cada uno de ellos, siendo un indicador muy útil para saber en qué sectores del servicio o producto es más importante enfocarse. Por último, la nube de palabras mejorada entrega la presencia de bigramas, que da un contexto más completo sobre las palabras de mayor frecuencia en las respuestas. Es muy útil como indicador de entrada, ya que da una introducción, sin considerar categorías, de cómo la gente se está expresando.

Finalmente, con respecto al cuarto objetivo específico para el cual era necesario haber cumplido los tres anteriores, se pudo comprobar que la API funciona de manera adecuada, incluyendo una capa de seguridad que permite que no solo las aplicaciones de QServus sean las únicas que puedan realizar las tareas, si no también identificarlas para dar un servicio más personalizado. A pesar de la poca variedad actual de los modelos de categorización, que hacen que solo empresas de un rubro o contexto específico puedan beneficiarse de esas funcionalidades, la comunicación y definición de los endpoints permiten que a futuro se puedan mantener y proveer estas capacidades a empresas de diversos rubros, logrando crecer tanto en peticiones como funcionalidades.

5.3. TRABAJO A FUTURO

QServus AI tiene un gran potencial dentro de la empresa consultora QServus, no solo en escalabilidad y en cómo podrá realizar todas sus tareas cuando la cantidad de clientes y respuestas por encuestas aumenten considerablemente, si no que también está la oportunidad de proyectar sus servicios ofreciendo análisis más precisos y avanzados.

Respecto al análisis de sentimiento, se concluye que es posible a futuro aumentar el nivel de comprensión, utilizando emociones en reemplazo de sentimientos. A pesar que los sentimientos dan un buen indicio sobre la intención, las emociones son mucho más fuertes en las opiniones de los clientes y presentan un nivel de análisis más específico. Para lo anterior se sugiere agregar emociones como alegría, enojo, tristeza, resignación, miedo, etc. Además, se puede definir una escala de emociones creciente desde emoción más negativa a la más positiva, y dar un puntaje que anime a las empresas a aumentarlas para mejorar la satisfacción

de sus clientes.

En el ámbito de categorización de texto es posible trabajar a futuro en varias aristas, como implementación de multi-categoría que permite dar un análisis más completo a respuestas más extensas. Por otro lado, la inclusión de sub-niveles de categorías, permitiendo que categorías generales puedan seguir siendo diversificadas para buscar conjuntos más claros. El gran beneficiado con la abstracción de niveles de categorías sería el gráfico de treemap, ya que su comportamiento encaja perfectamente con este nuevo requerimiento.

Se sugiere además la implementación de un modelo supervisado que tome los comentarios categorizados mediante el modelo de reglas y pueda mejorar la categorización de modelos de diversos rubros. Además, estos modelos supervisados tras su entrenamiento sirvirían para poder, con un análisis posterior, lograr la especificación de análisis para distintas empresas pero del mismo rubro.

Por último, la creación de nuevos indicadores con diferentes gráficos. Esto ofrece a los usuarios una gama de representaciones de los mismos datos, que se adaptarán a sus necesidades o costumbres. Ofrecer más opciones siempre será bienvenido para los usuarios.

5.4. REFLEXIONES PERSONALES

Este trabajo de título implicó una gran cantidad de conocimientos en diversas áreas las cuales, en su conjunto, permitieron la construcción de un software capaz de realizar lo anteriormente detallado.

Un aspecto fundamental fue el aprendizaje automático, tanto supervisado como no-supervisado. Esto ayudó a tener una gran gama de algoritmos y herramientas disponibles para analizar y escoger las más apropiadas para el problema. Gracias a la presencia en la carrera de las asignaturas electivas de Aprendizaje Automático e Introducción a las Redes Neuronales Artificiales, esta etapa del trabajo resultó fluida y se pudieron obtener los resultados buscados.

Para la obtención de los datos que nutren a los algoritmos de aprendizaje automático, es importante el conocimiento de procesamiento de lenguaje natural *NLP*. Es esencial para entender el contexto y los datos que se deben analizar: el rol de las palabras en una frase, cómo transformarlas para que la computadora las pueda comprender, la corrección de texto, *part of speech* de una palabra en una frase, etc. Gracias a esto, pueden anticipar errores en las respuestas e identificar de mejor forma la solución a ellos.

El *NLP* es esencial para comprender las respuestas, pero es necesario el conocimiento del negocio y así potenciar la semántica y el léxico de las respuestas. Esto implica conocer procesos y diversas siglas de los clientes, complejizando aún más el significado de diversas respuestas. Además, la conversación con los clientes fue esencial para tomar los requerimientos. Por lo

tanto, las habilidades blandas también fueron importantes para la negociación de requerimientos y el correcto entendimiento entre ambas partes.

La información obtenida de los algoritmos no era suficiente para los usuarios, porque seguía siendo información compleja y sin un contexto. Para eso, el conocimiento de la analítica de datos y cómo representarlos de forma gráfica fue esencial, ya que permite una correcta elección de los tipos de gráficos a usar. Esto incluye qué forma poseerán los gráficos, cómo los colores influyen en las apreciaciones de los usuarios, la formas de las figuras, títulos y cómo los usuarios leen los gráficos. Todo lo anterior debe ser considerado con la plataforma de QServus, por lo tanto, la inclusión de los nuevos gráficos en QServus debe ser orgánica y no representar una ruptura respecto a los gráficos ya existentes.

Cabe concluir también, la importancia que los pasos antes mencionados deben ser trabajados en conjunto. El conocimiento de desarrollo de software es esencial para la construcción de la API que alberga todas las funcionalidades y es capaz de recibir y entregar la información en tiempos acotados y con una estructura estricta y ordenada. Esto implica la conexión de diversas tecnologías, la creación de una base de datos estable, sencilla y sin redundancia. También se necesita conocimiento de *backend* y *frontend* para el correcto procesamiento y renderizado de información. La línea de desarrollo de software, que va desde Base de Datos hasta Diseño de Interfaces Usuarias fueron esenciales para la elección y desarrollo de la aplicación.

Con el software finalizado, se necesitaron conocimientos de servidores para el correcto levantamiento de la aplicación. Se realizó la incorporación de la aplicación en el servidor, migraciones de bases de datos, actualizaciones e instalación de requerimientos, asignación de sub-dominio para acceder a los recursos y configuraciones de puertos para la comunicación tanto interna como externa.

Finalmente, se necesitaron conocimientos de la plataforma de QServus en sí. Comprender cómo el ecosistema de sus diversas aplicaciones funciona, sus vistas y lógicas en cada aspecto. Lo anterior como un proceso de la elección del software, su arquitectura y para evitar replicar código o procesos ya existentes. También implica conocer los estándares de desarrollo de la empresa y replicarlos para que la implementación pueda ser comprendida por todos.

En resumen, el trabajo de título ofreció mucho aprendizaje en diversas áreas y también una actitud auto-didacta para encontrar soluciones, realizar pruebas y entregar resultados.

REFERENCIAS BIBLIOGRÁFICAS

[Cho *et al.*, 2014] Cho, K., van Merrienboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., y Bengio, Y. (2014). Learning phrase representations using rnn encoder-decoder for statistical machine translation.

[Fielding, 2000] Fielding, R. (2000). Architectural styles and the design of network-based software architectures.

[Gers *et al.*, 2000] Gers, F. A., Schmidhuber, J. A., y Cummins, F. A. (2000). Learning to forget: Continual prediction with lstm. *Neural Comput.*, 12(10):2451–2471.

[Goodfellow *et al.*, 2016] Goodfellow, I., Bengio, Y., y Courville, A. (2016). Deep learning.

[Géron, 2019] Géron, A. (2019). Hands-on machine learning with scikit-learn, keras tensorflow.

[Hochreiter y Schmidhuber, 1997] Hochreiter, S. y Schmidhuber, J. (1997). Long short-term memory. *Neural Comput.*, 9(8):1735–1780.

[Jurafsky y Martin, 2019] Jurafsky, D. y Martin, J. (2019). An introduction to natural language processing, computational linguistics, and speech recognition.

[Liddy, 2001] Liddy, E. D. (2001). Natural language processing.

[Manning *et al.*, 2008] Manning, C., Raghavan, P., y Schütze, H. (2008). An introduction to information retrieval.

[Reichheld, 2003] Reichheld, F. (2003). The one number you need to grow.

[Rumelhart *et al.*, 1986] Rumelhart, D., Hinton, G., y Williams, R. (1986). Learning representations by back-propagating errors. *Nature*, (323):533–536.

[Sebastiani, 2003] Sebastiani, F. (2003). Text categorization.

[Sparck, 1972] Sparck, J. K. (1972). A statistical interpretation of term specificity and its application in retrieval. *Journal of Documentation*, 28(1):11–21.

ANEXOS

DEFINICIÓN DE ENDPOINTS

6.1. ENCUESTAS

```
DELETE /api/analysis/survey/{survey_id}/
```

Comportamiento esperado: El endpoint busca la encuesta almacenada y sus dependencias, es decir, las preguntas y respuestas vinculadas. Luego de encontrarlas, las elimina del sistema.

Parámetros de llamada

- **survey_id:** Es el identificador de la encuesta en la aplicación que lo solicita.

Respuestas

- **Código 200:** Cuando el token enviado sea correcto y se encuentre un identificador de encuesta que le corresponda, se elimina el recurso y se devuelve un mensaje notificando.


```
{
    "message": "The survey was successfully eliminated",
    "extra_data": {
        "survey_id": 32,
        "count_survey_deleted": 1,
        "count_question_deleted": 8,
        "count_answers_deleted": 305,
    }
}
```


- **Código 404:** Si no existe el identificador para la empresa que lo está solicitando, devuelve un JSON explicando el error.

```
{
    "error": "That survey does not exists"
}
```

6.2. PREGUNTAS

```
DELETE /api/analysis/question/{question_id}/
```

Comportamiento esperado: El endpoint busca la pregunta almacenada y también las respuestas vinculadas. Tras encontrarlas, las elimina del sistema.

Parámetros de llamada

- **question_id:** Es el identificador de la pregunta en la aplicación que lo solicita.

Respuestas

- **Código 200:** Cuando el token enviado sea correcto y se encuentre un identificador de pregunta que le corresponda, se elimina el recurso y se devuelve un mensaje notificando.

```
{
    "message": "The question was successfully eliminated",
    "extra_data": {
        "question_id": 1222,
        "count_question_eliminated": 1,
        "count_answers_eliminated": 105
    }
}
```

- **Código 404:** Si no existe el identificador para la empresa que lo está solicitando, devuelve un JSON explicando el error.

```
{
    "error": "That question does not exists"
}
```

6.3. RESPUESTAS

```
DELETE /api/analysis/answer/{answer_id}/
```

Comportamiento esperado: El endpoint busca la respuesta almacenada y la elimina del sistema.

Parámetros de llamada

- **question_id:** Es el identificador de la respuesta en la aplicación que lo solicita.

Respuestas

- **Código 200:** Cuando el token enviado sea correcto y se encuentre un identificador de respuesta que le corresponda, se elimina el recurso y se devuelve un mensaje notificando.

```
{
    "message": "The question was successfully eliminated",
    "extra_data": {
        "answer_id": 3222,
        "count_answers_eliminated": 1
    }
}
```

- **Código 404:** Si no existe el identificador para la empresa que lo está solicitando, devuelve un JSON explicando el error.

```
{
    "error": "That answer does not exists"
}
```

6.4. MODELO DE CATEGORIZACIÓN

6.4.1. LEER KEYWORD

```
GET /api/analysis/keyword/{keyword_id}/
```

Comportamiento esperado: El endpoint busca la keyword almacenada y devuelve su nombre, además de la categoría a la cual pertenece.

Parámetros de llamada

- **keyword_id:** Es el identificador de la keyword en la aplicación.

Respuestas

- **Código 200:** Cuando el token enviado sea correcto y se encuentre un identificador de keyword que le corresponda, se devuelve un mensaje en formato JSON indicando el nombre de la keyword y a la categoría que pertenece.

```
{
    "keyword": "Polvo",
    "category": "Limpieza",
    "category_id": 25
}
```

- **Código 404:** Si no existe el identificador para la empresa que lo está solicitando, devuelve un JSON explicando el error.

```
{
    "error": "There is not a keyword with that identifier"
}
```

6.4.2. CREAR KEYWORDS

`POST /api/analysis/keyword/`

Comportamiento esperado: Almacena la keyword considerando los datos de entrada.

Parámetros de llamada

- **keywords:** Una lista de *strings* que representa la cantidad de keywords a almacenar.
- **category:** El identificador de la categoría a la cual pertenecerá/n la/s keyword/s.

- **model_id:** El identificador del modelo de categorización en el cual será almacenada/s la/s keywords.

Respuestas

- **Código 200:** Cuando el token enviado sea correcto y se encuentre un identificador de keyword que le corresponda, se devuelve un mensaje en formato JSON indicando el nombre de la keyword y a la categoría que pertenece.

```
{
    "keyword": "Polvo",
    "category": "Limpieza",
    "category_id": 25
}
```

- **Código 404:** Si no existe el identificador para la empresa que lo está solicitando, devuelve un JSON explicando el error.

```
{
    "error": "There is not a keyword with that identifier"
}
```

6.5. ANÁLISIS DE TEXTO

`POST /api/analysis/analyse_text/`

Comportamiento esperado: Categoriza y extrae sentimiento de las respuestas incluidas en el payload.

Parámetros de llamada

- **metadata:** Es un objeto que posee información contextual de las respuestas.
- **answers:** Una lista de objetos con las respuestas que serán analizadas.

Respuestas

- **Código 200:** Cuando el token enviado sea correcto y toda la información del payload cumpla con el formato establecido. El endpoint responde con un mensaje indicando que las respuestas se estarán procesando.

```
{
    "status": "Success",
    "message": "Everything is correct, the answers are processing",
    "answers_count": 234,
    "questions_count": 1
}
```

- **Código 404:** Este mensaje será retornado si no se cumple con la estructura correcta del payload. Dependiendo de qué sección es la errónea, será el mensaje resultante.

```
{
    "error": "There is not a categorization model with that
              identifier for that company"
}
```

6.6. AGRUPACIÓN DE TEXTOS

`POST /api/analysis/group_text/`

Comportamiento esperado: Agrupa respuestas parecidas, para ser mostradas en el indicador de tabla de comentarios.

Parámetros de llamada

- **qid:** Es el identificador de la pregunta, de la cual sus respuestas serán agrupadas si es que existen comentarios similares.

Respuestas

- **Código 200:** Cuando el token enviado sea correcto y y existen respuestas para el identificador de pregunta otorgado.

```
{
    "status": "Success",
    "message": "All the answers were grouped successfully",
}
```

- **Código 404:** Este mensaje será retornado si el identificador otorgado no existe, o no corresponde a la entidad que lo solicita.

```
{
    "error": "There is not a question with that identifier"
}
```

Desde esta parte, las respuestas de los siguientes endpoints no serán detallados en este documento, ya que muestra información sensible de la empresa, tanto de los mensajes que se usan como de la forma explícita en que se generan los gráficos.

6.7. GRÁFICO RANKING POR CATEGORÍA

`POST /api/chart/category_ranking/`

Comportamiento esperado: Retorna un objeto con toda la información e instrucciones para construir el gráfico de ranking por categoría.

Parámetros de llamada

- **qid:** Es el identificador de la pregunta, de la cual sus respuestas serán utilizadas para generar el indicador.

Respuestas

- **Código 200:** Cuando el token enviado sea correcto y y existen respuestas para el identificador de pregunta otorgado.
- **Código 404:** Este mensaje será retornado si el identificador otorgado no existe, o no corresponde a la entidad que lo solicita.

```
{
    "error": "There is not a question with that identifier"
}
```

6.8. GRÁFICO CATEGORÍA VS SENTIMIENTO

`POST /api/chart/sentiment_bar/`

Comportamiento esperado: Retorna un objeto con toda la información e instrucciones para construir el gráfico de categoría vs sentimiento.

Parámetros de llamada

- **qid:** Es el identificador de la pregunta, de la cual sus respuestas serán utilizadas para generar el indicador.

Respuestas

- **Código 200:** Cuando el token enviado sea correcto y y existen respuestas para el identificador de pregunta otorgado.
- **Código 404:** Este mensaje será retornado si el identificador otorgado no existe, o no corresponde a la entidad que lo solicita.

```
{
    "error": "There is not a question with that identifier"
}
```

6.9. NUBE DE PALABRAS MEJORADA

`POST /api/chart/wordcloud/`

Comportamiento esperado: Retorna un objeto con toda la información e instrucciones para construir el gráfico de nube de palabras mejorada.

Parámetros de llamada

- **qid:** Es el identificador de la pregunta, de la cual sus respuestas serán utilizadas para generar el indicador.

Respuestas

- **Código 200:** Cuando el token enviado sea correcto y y existen respuestas para el identificador de pregunta otorgado.
- **Código 404:** Este mensaje será retornado si el identificador otorgado no existe, o no corresponde a la entidad que lo solicita.

```
{
    "error": "There is not a question with that identifier"
}
```

6.10. TREEMAP

```
POST /api/chart/treemap/
```

Comportamiento esperado: Retorna un objeto con toda la información e instrucciones para construir el gráfico de treemap.

Parámetros de llamada

- **qid:** Es el identificador de la pregunta, de la cual sus respuestas serán utilizadas para generar el indicador.

Respuestas

- **Código 200:** Cuando el token enviado sea correcto y y existen respuestas para el identificador de pregunta otorgado.
- **Código 404:** Este mensaje será retornado si el identificador otorgado no existe, o no corresponde a la entidad que lo solicita.

```
{
    "error": "There is not a question with that identifier"
}
```

6.11. GRÁFICO SATISFACCIÓN VS CATEGORÍA

```
POST /api/chart/bubble_chart/
```

Comportamiento esperado: Retorna un objeto con toda la información e instrucciones para construir el gráfico de burbujas.

Parámetros de llamada

- **qid:** Es el identificador de la pregunta, de la cual sus respuestas serán utilizadas para generar el indicador.

Respuestas

- **Código 200:** Cuando el token enviado sea correcto y y existen respuestas para el identificador de pregunta otorgado.
- **Código 404:** Este mensaje será retornado si el identificador otorgado no existe, o no corresponde a la entidad que lo solicita.

```
{
    "error": "There is not a question with that identifier"
}
```

6.12. TABLA DE COMENTARIOS

```
POST /api/chart/comment_table/
```

Comportamiento esperado: Retorna un objeto con toda la información e instrucciones para construir el gráfico de tabla de comentarios.

Parámetros de llamada

- **qid:** Es el identificador de la pregunta, de la cual sus respuestas serán utilizadas para generar el indicador.

Respuestas

- **Código 200:** Cuando el token enviado sea correcto y y existen respuestas para el identificador de pregunta otorgado.
- **Código 404:** Este mensaje será retornado si el identificador otorgado no existe, o no corresponde a la entidad que lo solicita.

```
{
    "error": "There is not a question with that identifier"
}
```