
Señas Chapinas: Traductor de LENSEGUA

Procesamiento de Lenguaje Natural

Stefano Alberto Aragoni Maldonado



UNIVERSIDAD DEL VALLE DE GUATEMALA
Facultad de Ingeniería



Señas Chapinas: Traductor de LENSEGUA
Procesamiento de Lenguaje Natural

Trabajo de graduación en modalidad de Megaproyecto Tecnológico
presentado por Stefano Alberto Aragoni Maldonado
Para optar al grado académico de Licenciado en Ingeniería en Ciencia
de la Computación y Tecnologías de la Información

Guatemala, noviembre del 2024

UNIVERSIDAD DEL VALLE DE GUATEMALA
Facultad de Ingeniería



Señas Chapinas: Traductor de LENSEGUA
Procesamiento de Lenguaje Natural

Trabajo de graduación en modalidad de Megaproyecto Tecnológico
presentado por Stefano Alberto Aragoni Maldonado
Para optar al grado académico de Licenciado en Ingeniería en Ciencia
de la Computación y Tecnologías de la Información

Guatemala, noviembre del 2024

Vo.Bo.:

(f) _____
MSc, MBA, Ing. Luis Alberto Suriano Saravia

Tribunal Examinador:

(f) _____
MSc, MBA, Ing. Luis Alberto Suriano Saravia

(f) _____

(f) _____

Fecha de aprobación: Guatemala, ____ de diciembre de 2024.

Prefacio

La motivación detrás de este proyecto surge del deseo de querer apoyar a la comunidad sorda en Guatemala mediante el desarrollo de soluciones tecnológicas que faciliten su inclusión y mejoren su capacidad de comunicación. El objetivo es empoderar a las personas sordas, permitiéndoles interactuar en una sociedad que, con frecuencia, no se adapta a sus necesidades. Es importante mencionar que el sistema fue diseñado usando la gramática y vocabulario de la lengua de señas de la Ciudad de Guatemala, lo que podría limitar su alcance y aplicabilidad en otras regiones del país.

Por último, me gustaría expresar mi sincero agradecimiento al Ing. Alberto Suriano por su valiosa orientación y apoyo a lo largo de este proyecto. Asimismo, agradezco a la Asociación Educativa para el Sordo (ASEDES) y a En-Señas, cuya colaboración en la elaboración del *dataset* y la evaluación del sistema fue fundamental para el éxito del proyecto.

Índice

| | |
|--|-------------|
| Prefacio | v |
| Lista de Figuras | x |
| Lista de Cuadros | xi |
| Resumen | xiii |
| 1. Introducción | 1 |
| 2. Justificación | 3 |
| 3. Objetivos | 5 |
| 3.1. Objetivo General | 5 |
| 3.2. Objetivos Específicos | 5 |
| 4. Alcance | 7 |
| 5. Marco Teórico | 9 |
| 5.1. Discapacidad Auditiva | 9 |
| 5.1.1. Causas de la sordera | 9 |
| 5.1.2. Clasificación de la sordera | 10 |
| 5.1.3. Impacto de la discapacidad auditiva en las personas | 10 |
| 5.1.4. Barreras y desafíos enfrentados por la comunidad sorda en Guatemala | 11 |
| 5.2. Lengua de Señas de Guatemala (LENSEGUA) | 11 |
| 5.2.1. Historia | 11 |
| 5.2.2. Variaciones regionales | 12 |
| 5.2.3. Características | 12 |
| 5.3. Natural Language Processing (NLP) | 13 |
| 5.3.1. Técnicas y fundamentos | 13 |
| 5.3.2. Modelos principales | 14 |
| 5.4. Generative Pre-trained Transformer (GPT) | 16 |
| 5.4.1. Fundamentos de GPTs | 16 |
| 5.4.2. Fine-Tuning | 18 |
| 5.4.3. Prompt Engineering | 18 |
| 5.5. Explainable Artificial Intelligence (XAI) | 18 |
| 5.5.1. Local Interpretable Model-agnostic Explanations (LIME) | 19 |

| | |
|---|-----------|
| 6. Antecedentes | 21 |
| 7. Metodología | 23 |
| 7.1. Creación de conjunto de datos | 23 |
| 7.1.1. Recolección de datos | 23 |
| 7.1.2. Data augmentation | 25 |
| 7.1.3. Preparación del conjunto de datos | 26 |
| 7.2. Fine-tuning | 26 |
| 7.2.1. Identificación de parámetros óptimos | 26 |
| 7.2.2. Fine-tuning del modelo GPT-3.5-Turbo | 27 |
| 7.2.3. Evaluación del modelo GPT-3.5-Turbo fine-tuneado | 27 |
| 7.3. Prompt engineering | 28 |
| 7.4. Implementación y evaluación del sistema | 29 |
| 7.4.1. Desarrollo de plataforma web | 29 |
| 7.4.2. Desarrollo de encuesta | 29 |
| 7.4.3. Pruebas con usuarios | 30 |
| 8. Resultados | 31 |
| 8.1. Fine-tuning | 31 |
| 8.2. Prompt engineering | 35 |
| 8.3. Retroalimentación de la comunidad sorda | 38 |
| 9. Análisis de resultados | 43 |
| 10. Conclusiones | 47 |
| 11. Recomendaciones | 49 |
| Bibliografía | 51 |
| Anexos | 55 |

Lista de Figuras

| | | |
|-----|--|----|
| 1. | Arquitectura de red neuronal recurrente. | 14 |
| 2. | Arquitectura de transformer | 15 |
| 3. | Arquitectura de Generative Pre-trained Transformer (GPT). | 17 |
| 4. | Reunión de equipo de trabajo con ASEDES. | 24 |
| 5. | Generación recursiva de nuevas frases. | 25 |
| 6. | Reunión presencial en En-Señas donde se presentó la solución desarrollada. | 30 |
| 7. | Evolución de la pérdida durante el <i>fine-tuning</i> final del modelo GPT-3.5-Turbo. . . . | 32 |
| 8. | Resultados de LIME para la interpretación generada por el modelo GPT-3.5-Turbo estándar para la frase “antes tu policía llamar preguntá”. | 33 |
| 9. | Resultados de LIME para la interpretación generada por el modelo <i>fine-tuneado</i> para la frase “antes tu policía llamar preguntá”. | 33 |
| 10. | Resultados de LIME para la interpretación generada por el modelo GPT-3.5-Turbo estándar para la frase “futuro hospital él ir”. | 34 |
| 11. | Resultados de LIME para la interpretación generada por el modelo <i>fine-tuneado</i> para la frase “futuro hospital él ir”. | 34 |
| 12. | Resultados de LIME para la interpretación generada por el modelo GPT-3.5-Turbo estándar para la frase “pasado yo ir no”. | 35 |
| 13. | Resultados de LIME para la interpretación generada por el modelo <i>fine-tuneado</i> para la frase “pasado yo ir no”. | 35 |
| 14. | Aplicación web desarrollada para facilitar interacción con modelo <i>fine-tuneado</i> | 39 |
| 15. | Distribución de calificaciones otorgadas por los usuarios a la interpretación de la frase No. 1 generada por el modelo <i>fine-tuneado</i> con la última versión del <i>prompt</i> | 55 |
| 16. | Distribución de calificaciones otorgadas por los usuarios a la interpretación de la frase No. 1 generada por el modelo estándar con la última versión del <i>prompt</i> | 56 |
| 17. | Preferencia de los usuarios entre dos interpretaciones de la frase No. 1: Interpretación 1 (modelo <i>fine-tuneado</i>) e Interpretación 2 (modelo estándar). | 56 |
| 18. | Distribución de calificaciones para la interpretación de la frase No. 2 generada por el modelo <i>fine-tuneado</i> con la última versión del <i>prompt</i> | 57 |
| 19. | Distribución de calificaciones para la interpretación de la frase No. 2 generada por el modelo estándar con la última versión del <i>prompt</i> | 57 |
| 20. | Preferencia de los usuarios entre dos interpretaciones de la frase No. 2: Interpretación 1 (modelo <i>fine-tuneado</i>) e Interpretación 2 (modelo estándar). | 58 |
| 21. | Distribución de calificaciones para la interpretación de la frase No. 3 generada por el modelo <i>fine-tuneado</i> con la última versión del <i>prompt</i> | 58 |

| | |
|---|----|
| 22. Distribución de calificaciones para la interpretación de la frase No. 3 generada por el modelo estándar con la última versión del <i>prompt</i> | 59 |
| 23. Preferencia de los usuarios entre dos interpretaciones de la frase No. 3: Interpretación 1 (modelo <i>fine-tuneado</i>) e Interpretación 2 (modelo estándar). | 59 |
| 24. Distribución de calificaciones para la interpretación de la frase No. 4 generada por el modelo <i>fine-tuneado</i> con la última versión del <i>prompt</i> | 60 |
| 25. Distribución de calificaciones para la interpretación de la frase No. 4 generada por el modelo estándar con la última versión del <i>prompt</i> | 60 |
| 26. Preferencia de los usuarios entre dos interpretaciones de la frase No. 4: Interpretación 1 (modelo <i>fine-tuneado</i>) e Interpretación 2 (modelo estándar). | 61 |
| 27. Distribución de calificaciones para la interpretación de la frase No. 5 generada por el modelo <i>fine-tuneado</i> con la última versión del <i>prompt</i> | 61 |
| 28. Distribución de calificaciones para la interpretación de la frase No. 5 generada por el modelo estándar con la última versión del <i>prompt</i> | 62 |
| 29. Preferencia de los usuarios entre dos interpretaciones de la frase No. 5: Interpretación 1 (modelo <i>fine-tuneado</i>) e Interpretación 2 (modelo estándar). | 62 |
| 30. Distribución de calificaciones para la interpretación de la frase No. 6 generada por el modelo <i>fine-tuneado</i> con la primera versión del <i>prompt</i> | 63 |
| 31. Distribución de calificaciones para la interpretación de la frase No. 6 generada por el modelo <i>fine-tuneado</i> con la segunda versión del <i>prompt</i> | 63 |
| 32. Distribución de calificaciones para la interpretación de la frase No. 6 generada por el modelo <i>fine-tuneado</i> con la tercera versión del <i>prompt</i> | 64 |
| 33. Distribución de calificaciones para la interpretación de la frase No. 6 generada por el modelo <i>fine-tuneado</i> con la cuarta versión del <i>prompt</i> | 64 |
| 34. Distribución de calificaciones para la interpretación de la frase No. 6 generada por el modelo <i>fine-tuneado</i> con la última versión del <i>prompt</i> | 65 |
| 35. Preferencia de los usuarios entre las interpretaciones de la frase No. 6: Interpretación 1 (versión 1 del <i>prompt</i>), Interpretación 2 (versión 2 del <i>prompt</i>), Interpretación 3 (versión 3 del <i>prompt</i>), Interpretación 4 (versión 4 del <i>prompt</i>) e Interpretación 5 (versión 5 del <i>prompt</i>). | 66 |

Lista de Cuadros

| | | |
|-----|--|----|
| 1. | Estructura del conjunto de datos desarrollado. | 24 |
| 2. | Ejemplos de perturbaciones interpretadas por el modelo <i>fine-tuneado</i> junto con sus distancias de Levenshtein normalizadas correspondientes. | 28 |
| 3. | Valores de parámetros y pérdidas de validación en experimentos preliminares | 31 |
| 4. | Parámetros utilizados para el <i>fine-tuning</i> del modelo GPT-3.5-Turbo. | 32 |
| 5. | Contribución de cada palabra según los Valores de Influencia (LIME) en las interpretaciones generadas por el modelo GPT-3.5-Turbo estándar y su versión <i>fine-tuneada</i> para la frase “antes tu policía llamar pregunta”. | 33 |
| 6. | Contribución de cada palabra según los Valores de Influencia (LIME) en las interpretaciones generadas por el modelo GPT-3.5-Turbo estándar y su versión <i>fine-tuneada</i> para la frase “futuro hospital él ir”. | 34 |
| 7. | Contribución de cada palabra según los Valores de Influencia (LIME) en las interpretaciones generadas por el modelo GPT-3.5-Turbo estándar y su versión <i>fine-tuneada</i> para la frase “pasado yo ir no”. | 35 |
| 8. | Distancias de Levenshtein promedio calculadas a partir de las interpretaciones generadas por el modelo GPT-3.5-Turbo (estándar) y el modelo <i>fine-tuneado</i> , utilizando diversos <i>prompts</i> | 38 |
| 9. | Puntuación promedio otorgada por usuarios finales a las interpretaciones generadas por el modelo GPT-3.5-Turbo estándar y su versión <i>fine-tuneada</i> para la frase “ayer cine tu ir pregunta”. | 39 |
| 10. | Puntuación promedio otorgada por usuarios finales a las interpretaciones generadas por el modelo GPT-3.5-Turbo estándar y su versión <i>fine-tuneada</i> para la frase “pasado yo medicina comprar para mamá”. | 40 |
| 11. | Puntuación promedio otorgada por usuarios finales a las interpretaciones generadas por el modelo GPT-3.5-Turbo estándar y su versión <i>fine-tuneada</i> para la frase “ayer abuelo llamar tu pregunta”. | 40 |
| 12. | Puntuación promedio otorgada por usuarios finales a las interpretaciones generadas por el modelo GPT-3.5-Turbo estándar y su versión <i>fine-tuneada</i> para la frase “antes tienda tu amiga ver pregunta”. | 40 |
| 13. | Puntuación promedio otorgada por usuarios finales a las interpretaciones generadas por el modelo GPT-3.5-Turbo estándar y su versión <i>fine-tuneada</i> para la frase “tu opinión cuál pregunta”. | 41 |
| 14. | Puntuación promedio otorgada por usuarios finales a las interpretaciones generadas por el modelo GPT-3.5-Turbo <i>fine-tuneado</i> , utilizando diferentes <i>prompts</i> , para la frase “hospital yo necesitar ir ahora ojalá mucho carro no porque emergencia”. | 41 |

Resumen

La lengua de señas guatemalteca (LENSEGUA) posee una gramática propia que difiere significativamente del español tradicional. Debido a estas diferencias, la traducción directa de una secuencia de señas al español no garantiza coherencia lingüística. Con el fin de resolver esta problemática, se adaptó un modelo GPT-3.5-Turbo, mediante la técnica de *fine-tuning*, para que pudiera interpretar estos textos y generar respuestas coherentes.

El proyecto se llevó a cabo en tres fases: en primer lugar, se creó un *corpus* representativo en colaboración con intérpretes y personas sordas para capturar las particularidades gramaticales de LENSEGUA; posteriormente, se realizó el *fine-tuning* del modelo, ajustando parámetros clave para optimizar su capacidad interpretativa; y finalmente, se aplicaron técnicas de *prompt engineering* para mejorar la claridad de las instrucciones proporcionadas al modelo. Los resultados demostraron la efectividad de la metodología, ya que al comparar las interpretaciones del modelo *fine-tuneado* con las del modelo estándar, se observó una reducción del 42.23 % en la distancia promedio de Levenshtein. En otras palabras, las interpretaciones del modelo *fine-tuneado* fueron significativamente más precisas y coherentes en comparación con las del modelo estándar.

En conclusión, este trabajo presenta una herramienta que, según la comunidad sorda, facilita la comunicación entre personas sordas y oyentes. Además, el éxito del proyecto resalta el potencial de la inteligencia artificial para abordar desafíos sociales y culturales específicos, evidenciando que la tecnología puede ser un aliado importante en la búsqueda de soluciones que promuevan la equidad.

CAPÍTULO 1

Introducción

En Guatemala, aproximadamente 300,000 personas enfrentan algún grado de discapacidad auditiva [35]. En un esfuerzo por reducir la brecha lingüística existente entre la comunidad sorda y la oyente, en el 2020, el Congreso de Guatemala promulgó el Decreto 3-2020. Esta legislación reconoce oficialmente a la lengua de señas de Guatemala, también conocida como LENSEGUA, como un medio de comunicación compuesto por movimientos corporales y una gramática propia de las personas sordas [6]. A pesar de este importante avance legislativo, su impacto hasta el momento ha sido limitado, lo que subraya la urgencia de implementar soluciones innovadoras para mejorar la comunicación y la inclusión de las personas sordas en la sociedad guatemalteca.

En este contexto, a través de este megaproyecto se busca crear un sistema de reconocimiento e interpretación de LENSEGUA. Para esto, se plantea el desarrollo de una aplicación móvil que le permita a los usuarios grabar videos de personas utilizando la lengua de señas guatemalteca. Una vez grabados, se analizará automáticamente los videos para reconocer y, posteriormente, interpretar las señas al idioma español. Es importante destacar que para este proyecto únicamente se usarán ciertas palabras de uso cotidiano.

El presente módulo se centra en la adaptación de un *large language model* (LLM) utilizando la técnica de *fine-tuning*. El propósito de este enfoque es permitir que un modelo GPT detecte y aprenda la gramática de LENSEGUA mediante el análisis de frases ilustrativas. A partir de este aprendizaje, se espera que el modelo pueda recibir palabras asociadas a señas, previamente detectadas por un modelo de visión por computadora, para luego interpretarlas y generar oraciones coherentes en español. La necesidad de esta adaptación se fundamenta en la considerable diferencia gramatical entre la lengua de señas y el español, lo que hace insuficiente la simple detección de señas para una comunicación efectiva.

CAPÍTULO 2

Justificación

La Asociación de Sordos de Guatemala (ASEGUA) establece que, para asegurar una inclusión adecuada de la comunidad sorda en el país, deberían haber 10 personas oyentes capacitadas en lengua de señas por cada persona sorda. Sin embargo, la realidad del país difiere de este indicador, ya que únicamente se ha reportado la participación de 11,500 personas en cursos básicos de lengua de señas. Además, ASEGUA ha señalado que únicamente existen 95 intérpretes certificados en todo el territorio nacional [22].

Esta situación, según la 'Encuesta Sociolingüística de la Comunidad Sorda en Guatemala', marginaliza a la población sorda y la limita a relacionarse exclusivamente con otras personas sordas o con hijos de adultos sordos (HDAS). La escasez de personas oyentes capacitadas en LENSEGUА y de intérpretes certificados crea barreras significativas para la comunicación y la participación de personas sordas en la sociedad guatemalteca. Por ejemplo, debido a la falta de personal capacitado, de las 10 escuelas públicas para personas sordas, ninguna ofrece educación secundaria. Esta situación, a su vez, ocasiona que las personas sordas no puedan acceder a trabajos que les proporcionen un salario digno [27].

Con base a lo anteriormente expuesto, el proyecto 'Señas Chapinas' tiene como objetivo mitigar las dificultades que enfrentan las personas sordas en Guatemala. Más específicamente, busca mejorar la comunicación entre ambas comunidades mediante una aplicación que permite capturar gestos en lengua de señas, los cuales son interpretados y convertidos en oraciones en español. Esta herramienta espera facilitar la interacción de las personas sordas con su entorno, permitiéndoles expresarse con mayor facilidad.

En consonancia con este objetivo, el módulo de procesamiento de lenguaje tiene como meta abordar las complejidades inherentes de la lengua de señas guatemalteca. A diferencia del español, donde las oraciones siguen la estructura sujeto, verbo y objeto, en la lengua de señas este orden se invierte. Además, es importante destacar que no se utilizan artículos, preposiciones ni conjunciones [22]. Debido a estas diferencias, la traducción directa de una secuencia de señas al español no garantiza coherencia lingüística. Por tanto, la incorporación de un *large language model* busca proporcionar una interpretación más precisa y significativa de los mensajes expresados en LENSEGUА.

CAPÍTULO 3

Objetivos

3.1. Objetivo General

Adaptar un *large language model* (LLM) para que asimile la gramática de LENSEGUA, con el fin de que el modelo pueda interpretar oraciones que utilicen dicha gramática y las escriba correctamente en español.

3.2. Objetivos Específicos

- Recopilar una variedad de frases relevantes que ejemplifiquen la gramática de LENSEGUA y generar un conjunto de datos a partir de ellas.
- Aplicar la técnica de *fine-tuning* en el modelo GPT-3.5-Turbo utilizando el conjunto de datos desarrollado.
- Explorar y aplicar la técnica de *prompt engineering* para diseñar *prompts* específicos que maximicen el rendimiento del modelo adaptado.
- Evaluar la coherencia de las frases generadas por el LLM adaptado mediante pruebas con usuarios representativos de la comunidad sorda guatemalteca, utilizando una escala de puntuación de 1 a 5, donde 5 representa una alta calidad y coherencia en las interpretaciones, y 1 indica una falta de coherencia en las mismas.

CAPÍTULO 4

Alcance

El alcance del proyecto se enfoca en el *fine-tuning* de un modelo de procesamiento de lenguaje natural, adaptado exclusivamente a la gramática y al vocabulario de LENSEGUA utilizado en la Ciudad de Guatemala. Por lo tanto, este enfoque puede presentar limitaciones cuando el modelo se enfrenta a variaciones de LENSEGUA relacionadas con otras regiones del país. Además, el conjunto de datos empleado no abarca todo el vocabulario de LENSEGUA, sino que se limita a las palabras y frases más utilizadas por esta comunidad.

El proyecto incluye también el desarrollo de una aplicación web para facilitar la evaluación del sistema. Esta aplicación le permitirá a un grupo selecto de usuarios ingresar texto que utilice la gramática mencionada y recibir respuestas gramaticalmente correctas en español. Cabe destacar que dicha herramienta no será de acceso público. Las pruebas se llevarán a cabo con usuarios seleccionados para medir la precisión del sistema y su capacidad para mejorar la comunicación entre personas sordas y oyentes.

Es importante señalar que la disponibilidad del sistema está sujeta a la disponibilidad del servicio de ChatGPT, en el cual se basa el modelo. Además, dado que cada interpretación o consulta realizada al modelo implica un costo asociado, la cantidad de consultas está limitada al crédito disponible en la plataforma de OpenAI.

CAPÍTULO 5

Marco Teórico

5.1. Discapacidad Auditiva

La discapacidad auditiva es una condición que limita la capacidad de una persona para escuchar. Esta puede manifestarse en distintos niveles de intensidad, desde una pérdida leve de audición hasta la sordera total. Además, esta condición puede tener diversas causas, como factores genéticos o exposición prolongada a ruidos fuertes [41]. Es importante destacar que, independientemente de su origen o grado, la discapacidad auditiva puede representar un desafío significativo en la vida diaria de quienes la experimentan, afectando su comunicación, educación e integración social [42].

5.1.1. Causas de la sordera

La Federación Mundial de Sordos (WFD) reporta que en todo el mundo hay 70 millones de personas sordas, de las cuales 34 millones son niños. Además, se estima que alrededor de 3 de cada 1,000 bebés nacen con sordera [42]. En este contexto, es importante destacar que la sordera en los niños puede manifestarse durante el período prenatal debido a diversos factores. Por ejemplo, hay mutaciones genéticas (hereditarias y no hereditarias) que pueden aumentar la probabilidad de sordera congénita. Asimismo, existen infecciones intrauterinas, como la rubéola o el herpes, que también pueden desencadenar la sordera en el feto. Otra causa común es la exposición a medicamentos por parte de la madre que pueden provocar diversas malformaciones [24].

La prematuridad de un bebé es también un factor que puede influir significativamente en el desarrollo de problemas auditivos. Concretamente, los bebés prematuros tienen un mayor riesgo de desarrollar sordera debido a la posibilidad de que sus oídos no hayan alcanzado su completo desarrollo al momento del nacimiento. Además, los bebés prematuros comúnmente experimentan enfermedades que aumentan la probabilidad de desarrollar sordera, como la hiperbilirrubinemia (ictericia grave). Por otro lado, durante el parto pueden surgir complicaciones que resultan en asfixia perinatal, evento que puede ocasionar daños al sistema auditivo y nervioso del recién nacido [24].

Cabe destacar que la sordera no solo aparece durante el período prenatal o perinatal, sino que también puede manifestarse en cualquier etapa de la vida, incluyendo la infancia, la adolescencia y la edad adulta. En niños y jóvenes, infecciones recurrentes del oído (otitis media crónica) pueden provocar pérdida de audición si no se tratan adecuadamente. A su vez, la meningitis es una infección

que también puede ocasionar daños en el nervio auditivo y, consecuentemente, resultar en pérdida auditiva. Finalmente, la exposición continua a ruidos fuertes puede dañar gradualmente la audición a lo largo del tiempo. Esto es especialmente relevante en los jóvenes y adultos que utilizan con más frecuencia audífonos a un volumen alto [42].

5.1.2. Clasificación de la sordera

Una persona sorda, según la Organización Mundial de la Salud (OMS), es aquella cuya capacidad auditiva no supera los 20 decibelios (dB) en ambas orejas [42]. Sin embargo, es importante tener en cuenta que la sordera puede manifestarse en diferentes grados, los cuales dependen de la gravedad de la pérdida auditiva y de la ubicación específica del problema en el oído.

En términos de grado de pérdida auditiva, la sordera se puede clasificar en leve, moderada, grave y profunda. La pérdida auditiva leve se refiere a una persona que puede escuchar algunos sonidos relacionados con el habla, pero no puede escuchar susurros. A una persona con pérdida auditiva moderada le cuesta entender el habla a un volumen normal. Por otro lado, una persona con pérdida auditiva grave no escucha ningún sonido a volumen normal, solo puede oír sonidos fuertes. Por último, una persona con pérdida auditiva profunda (o con sordera total) no escucha nada, exceptuando algunos sonidos extremadamente fuertes [21].

Al considerar la ubicación de la pérdida auditiva, esta puede clasificarse en tres tipos principales: conductiva, neurosensorial o mixta. La pérdida auditiva conductiva ocurre cuando hay un bloqueo que impide el paso del sonido del oído externo al medio. En cambio, la pérdida auditiva neurosensorial ocurre cuando hay daño en el oído interno o en el nervio auditivo. Esto resulta en que los sonidos no puedan ser procesados y posteriormente transmitidos al cerebro de forma correcta. Finalmente, la pérdida auditiva mixta, como su nombre sugiere, involucra una combinación de factores tanto conductivos como neurosensoriales. Esto puede ser el resultado de múltiples condiciones que afectan tanto el oído externo como el interno [21].

5.1.3. Impacto de la discapacidad auditiva en las personas

Actualmente se reporta que el 80 % de personas sordas residen en países de bajos ingresos, como Guatemala. En estos países, las condiciones precarias dificultan el acceso a servicios de salud auditiva adecuados, así como a tecnologías de asistencia y educación especializada. Como resultado, es frecuente encontrar niños, jóvenes e incluso adultos con pérdida de audición no solo no tratada, sino también no diagnosticada [42].

La sordera no tratada, según Dave Cutten, puede obstaculizar el desarrollo lingüístico y comunicativo de los niños, lo que incluye impedimentos en el desarrollo del vocabulario y dificultades para comprender y utilizar el lenguaje de manera efectiva [7]. Es fundamental destacar que los niños con sordera identificada suelen aprender a comunicarse a través de la lengua de señas, lo que mitiga en gran medida estos problemas. Sin embargo, independientemente de si los niños con sordera aprenden lengua de señas o no, existe una brecha lingüística entre la comunidad sorda y la oyente que tiene implicaciones psicológicas significativas. Por ejemplo, las personas sordas tienden a sentirse excluidas, solas, frustradas, enojadas y avergonzadas debido a que no pueden socializar fácilmente con sus familiares, compañeros y miembros de sus comunidades. Además, esta situación puede afectar negativamente el bienestar emocional de las personas sordas, generando sentimientos de ansiedad, depresión y baja autoestima [5].

En el caso de los adolescentes y adultos, la discapacidad auditiva puede provocar efectos psicológicos y emocionales similares a los descritos anteriormente. Además, esta condición puede influir negativamente en las oportunidades educativas y laborales de las personas, lo que a su vez puede conducir a una mayor exclusión y discriminación en la sociedad [4]. Por ejemplo, en Guatemala, de

las 10 escuelas para personas sordas, ninguna ofrece educación secundaria. Esto, a su vez, ocasiona una brecha educativa significativa para los adolescentes sordos, limitando así su futuro académico y profesional [27].

5.1.4. Barreras y desafíos enfrentados por la comunidad sorda en Guatemala

En 2007, Elizabeth Parks y Jason Parks realizaron un estudio con el objetivo de analizar la situación sociolingüística de la comunidad sorda en Guatemala. A través de entrevistas y encuestas, identificaron que menos del 50 % de los sordos guatemaltecos reciben educación especializada. Además, observaron un bajo nivel de alfabetización, así como un pobre dominio del español entre este grupo. Según los autores, esta situación se atribuye a la presencia limitada de solo diez escuelas para los veintidós departamentos del país. De estas diez, tres son instituciones privadas, lo que las vuelve inaccesibles para muchas familias debido a limitaciones monetarias [27].

En cuanto a educación superior, según Edith Paz, únicamente dos universidades en Guatemala ofrecen servicios de intérpretes para estudiantes sordos, aunque estos servicios están limitados exclusivamente a los departamentos de computación. Es importante destacar que, si bien los estudiantes tienen la opción de contratar intérpretes privados, esta posibilidad está restringida a aquellos provenientes de familias de altos recursos. Debido a esto, muchas personas sordas encuentran dificultades para acceder a empleos con un salario digno. Por ejemplo, el salario promedio mensual para un guatemalteco es de Q2,000, sin embargo, las personas sordas suelen ganar –en promedio– menos de Q600 mensuales [27].

5.2. Lengua de Señas de Guatemala (LENSEGUA)

La lengua de señas es un sistema de comunicación utilizado por personas sordas y con discapacidad auditiva. Este utiliza gestos, movimientos de las manos, expresiones faciales y posturas corporales para transmitir ideas y emociones [25]. Es importante mencionar que, al igual que los idiomas hablados, las lenguas de señas varían considerablemente entre países, existiendo más de 300 variaciones en todo el mundo [40]. En Guatemala –particularmente–, la Lengua de Señas de Guatemala (LENSEGUA) se ha establecido como la principal forma de comunicación para la comunidad sorda. Esta se ha desarrollado a lo largo de los años, incorporando elementos de la cultura guatemalteca y adaptándose a las necesidades de las personas sordas en el país [27].

5.2.1. Historia

El primer registro de la utilización de la lengua de señas en Guatemala se remonta a la escuela Fray Pedro Ponce de León. Esta institución, establecida en la Ciudad de Guatemala en 1946, fue fundada específicamente para la educación de niños y niñas sordas. Sin embargo, según Edith Paz, tenía una filosofía oralista que prohibía el uso de señas y gestos para la comunicación. A pesar de esta restricción, los estudiantes (durante el transcurso de 20 años) desarrollaron un sistema de señas para comunicarse entre ellos, tanto dentro de las aulas como en público. Esta lengua de señas (también conocida como GSM) fue evolucionando con el paso del tiempo, incorporando influencias de sistemas similares utilizados en España, Cuba, Costa Rica, El Salvador y Estados Unidos [27].

A finales del siglo XX, se inauguraron otras diez escuelas para los jóvenes con discapacidad auditiva. A diferencia de la Escuela Fray Pedro Ponce de León, todas estas fomentaban el uso de la lengua de señas. Entre ellas, dos enseñaban *American Sign Language* (ASL), mientras que las otras una variación de la lengua de señas formalizada a finales de la década de los sesenta (GSM) [27].

En el 2001, el Comité Pro Ciegos y Sordos de Guatemala, una institución privada no lucrativa, en conjunto con otros colaboradores, publicó el primer manual oficial de la Lengua de Señas de Guatemala (LENSEGUA). Este manual contiene instrucciones sobre cómo realizar una gran cantidad de gestos, así como su equivalente en español. Sin embargo, cabe destacar que este manual no tomó relevancia hasta el 2021, cuando el Congreso de la República de Guatemala aprobó la ‘Ley que Reconoce y Aprueba la Lengua de Señas de Guatemala’ (Decreto Número 135-96). Esta ley, como indica su nombre, reconoce a LENSEGUA como un medio de comunicación compuesto por un conjunto de movimientos corporales y una gramática propia de las personas sordas. Asimismo, establece que el Ministerio de Educación debe promover la introducción de LENSEGUA al sistema educativo nacional [6].

Más específicamente, el Decreto Número 135-96 establece que las instituciones educativas (públicas o privadas) deben capacitar a los docentes en el uso de LENSEGUA para poder apoyar a sus estudiantes sordos. Para esta capacitación, se recomienda el uso de plataformas digitales y audiovisuales. De igual manera, indica que los centros educativos deben modificar los cursos para ser inclusivos con aquellos estudiantes con problemas auditivos [6].

En cuanto al ámbito laboral, el reglamento establece que las empresas y el estado deben ofrecer oportunidades laborales a personas con discapacidades. Además, se considera fundamental proporcionar capacitación y herramientas adecuadas para ayudar a estas personas a enfrentar y superar sus limitaciones [6].

5.2.2. Variaciones regionales

Aunque LENSEGUA es reconocida como la forma predominante de comunicación para la comunidad sorda en Guatemala, existen variaciones regionales que reflejan la diversidad cultural y lingüística dentro del país. Estas diferencias incluyen, pero no se limitan a, cambios sutiles en el vocabulario y los gestos utilizados. Por ejemplo, según Edith Paz, mientras que la Ciudad de Guatemala, Cobán y Quetzaltenango muestran similitudes significativas, San Marcos presenta un sistema ligeramente diferente, el cual se inspira aún más en el ASL [27].

5.2.3. Características

LENSEGUA, como se mencionó anteriormente, utiliza una gramática propia, que difiere significativamente del español. Por ejemplo, LENSEGUA no utiliza artículos ni identificadores de género. Esto quiere decir que no se emplean palabras como ‘el’ o ‘la’ para indicar el género de los sustantivos, ni se utilizan artículos definidos o indefinidos como en el español. Asimismo, en LENSEGUA no está contemplado el uso de preposiciones. Las palabras como ‘a’, ‘de’ y ‘en’ simplemente son omitidas, debido a que se sobreentienden según el contexto [13].

Otra característica particular de LENSEGUA es que los verbos no se conjugan según la persona, el número o el tiempo verbal. Por ejemplo, en lugar de conjugaciones específicas como ‘hablo’, ‘hablas’ y ‘habla’, se utiliza un solo signo para el verbo ‘hablar’, que puede ser interpretado en diferentes contextos según la situación. En caso sea relevante aclarar el tiempo verbal, se utilizan señas adicionales como ‘ayer’, ‘hoy’ y ‘mañana’. Estos adverbios de tiempo ayudan a situar la acción en el pasado, presente o futuro, sin necesidad de modificar la forma del verbo [13].

Por otro lado, LENSEGUA también se caracteriza por omitir varios signos de puntuación. A diferencia del español u otros idiomas escritos, donde se utilizan puntos, comas y otros símbolos para delimitar oraciones y expresar pausas, en LENSEGUA estas pausas y divisiones se representan mediante cambios en la velocidad, el ritmo y la posición de los gestos [13].

Finalmente, otra singularidad de LENSEGUA es el orden gramatical de una oración o idea. En

LENSEGUA, primero se indica el tiempo, luego el lugar, el sujeto, el objeto y, por último, el verbo. Esto difiere del español, donde primero se establece el sujeto, luego el verbo y el objeto. Por ejemplo, mientras en español se dice 'yo jugué fútbol ayer', en LENSEGUA se expresaría como 'ayer yo fútbol jugar' [13].

Según la organización En-Señas, estas diferencias lingüísticas causan que la interpretación de LENSEGUA al español sea una tarea complicada. Esto principalmente debido a que la traducción directa de una secuencia de señas al español no garantiza coherencia lingüística. Por tal motivo, los intérpretes deben adaptar el mensaje, considerando tanto el significado literal como la estructura gramatical del español, para asegurar una comunicación efectiva y comprensible [13].

5.3. Natural Language Processing (NLP)

El *natural language processing* (NLP), o procesamiento de lenguaje natural, es un campo de la inteligencia artificial que se enfoca en la comprensión y manipulación del lenguaje humano. Este campo no se limita a un solo modelo, sino que abarca una variedad de técnicas y algoritmos diseñados para interpretar y generar texto de manera efectiva [3]. Estos modelos se pueden utilizar en diversos contextos, desde análisis de sentimientos hasta traducciones automáticas, lo que demuestra su versatilidad y aplicación práctica en diferentes áreas de la tecnología y la comunicación [8].

5.3.1. Técnicas y fundamentos

En NLP, el preprocesamiento del texto es una etapa en la cual se preparan y limpian los datos antes de aplicar técnicas de análisis más avanzadas. Este proceso incluye varias etapas, como la *tokenización*. Esto consiste en dividir el texto en unidades más pequeñas, como palabras, con el fin de facilitar su posterior análisis. Además, el preprocesamiento también suele involucrar el *stemming*, que consiste en reducir las palabras a su forma base o raíz. Esto ayuda a normalizar el texto y a eliminar las variaciones morfológicas. Por último, se suele llevar a cabo la eliminación de *stopwords*, que son palabras comunes pero que no aportan un significado contextual importante al texto, como 'el', 'la', 'de' y 'en'. En algunos casos, cabe destacar que puede ser necesario realizar un preprocesamiento más exhaustivo. Por ejemplo, dependiendo de las necesidades, se puede considerar el manejo de sinónimos para reducir el vocabulario de un texto [8].

Posterior al preprocesamiento, se puede aplicar *text feature extraction*. Este término engloba diferentes métodos, como *bag of words* y *N-grams*. Sin embargo, todos tienen como objetivo crear estructuras de datos que permiten representar de manera adecuada la información contenida en el texto. Por ejemplo, *bag of words* genera un listado de palabras que constituyen el vocabulario de un texto, acompañado de la frecuencia con la que cada palabra aparece. Esta información puede ser útil para identificar la importancia relativa de las palabras en un documento. *N-grams*, por otro lado, es una técnica en la cual se generan secuencias de palabras de longitud N. Estas secuencias se pueden utilizar para identificar las relaciones y estructuras entre las palabras en un texto [8].

Una vez realizado el preprocesamiento del texto y creado las estructuras de datos adecuadas, se puede introducir dicha información en un modelo (o arquitectura) para llevar a cabo una variedad de tareas. Estas pueden incluir el análisis de sentimientos y la generación de texto. El análisis de sentimientos busca comprender las emociones expresadas en el texto, determinando si son positivas, negativas o neutras. Por otro lado, la generación de texto implica crear contenido de manera automática, ya sea completando frases, resumiendo artículos o incluso generando respuestas a preguntas del usuario [10].

5.3.2. Modelos principales

Existe una amplia variedad de modelos diseñados para llevar a cabo las tareas de procesamiento de lenguaje natural mencionadas anteriormente. La elección entre estos depende de varios factores, como la complejidad de la tarea, la disponibilidad y calidad de los datos, y los recursos computacionales disponibles.

Aprendizaje automático

Entre uno de los modelos más simples está el clasificador *Naïve Bayes*. Este es un modelo probabilístico que –como indica su nombre– emplea el teorema de Bayes. En otras palabras, funciona a través de determinar la probabilidad de que un cierto evento ocurra dado que otro evento ya ha ocurrido (asumiendo la independencia condicional). Para su funcionamiento correcto, este clasificador suele recibir el vocabulario y la frecuencia con la que aparecen las respectivas palabras. A partir de esta información, por ejemplo, se puede determinar la probabilidad de que un texto pertenezca a una clase específica, como ‘positivo’ o ‘negativo’, dadas las palabras que contiene. En la detección de spam, se puede calcular la probabilidad de que un correo electrónico sea spam o no, basándose en la frecuencia de ciertas palabras o características en el mensaje [30].

La regresión logística, similarmente, es otro modelo utilizado en el aprendizaje automático para problemas de clasificación. A pesar de su nombre, la regresión logística se utiliza principalmente para problemas de clasificación binaria, donde el objetivo es predecir la pertenencia a una de dos categorías distintas. A diferencia del clasificador *Naïve Bayes*, que se basa en la probabilidad condicional, la regresión logística utiliza una función logística para modelar la relación entre las variables de entrada y la probabilidad de pertenencia a una clase específica. Esto permite su aplicación en una variedad de tareas de NLP, donde se busca clasificar el texto en categorías específicas [9].

Aprendizaje profundo

Por otro lado, los modelos de aprendizaje automático se caracterizan por la utilización de redes neuronales. Una red neuronal es un modelo, compuesto por nodos interconectados (o neuronas), capaz de aprender patrones complejos en datos. Estos modelos, debido a su arquitectura, son capaces de realizar una mayor cantidad de tareas relacionadas con NLP, como generación de texto, traducción automática, resumen de documentos, respuesta a preguntas, entre otras [17].

Es importante destacar que para entrenar y utilizar eficazmente estos modelos es necesario realizar una etapa de transformación de los datos mediante el proceso de *embedding*. El *embedding* es un proceso que convierte palabras o frases en vectores numéricos, permitiendo a los modelos de aprendizaje automático comprender su significado y relación. Además, se utiliza el *padding* para igualar la longitud de las secuencias de entrada, lo que facilita su procesamiento en los modelos [20].

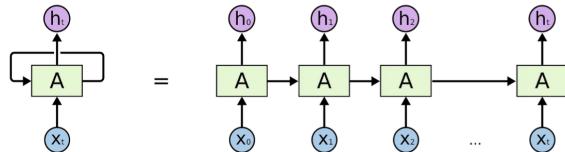


Figura 1: Arquitectura de red neuronal recurrente.

Las *recurrent neural networks* (RNN), o redes neuronales recurrentes, son comúnmente relacionadas con tareas de procesamiento de lenguaje natural. Estas redes trabajan con datos secuenciales,

como series de tiempo o texto, debido a su capacidad para capturar dependencias entre los datos. Más específicamente, como se puede observar en la 1, las capas de estas redes toman como entrada el dato actual a procesar así como la salida de la capa recurrente en el paso anterior. Esto les permite mantener un registro que se actualiza constantemente, y que –como resultado– les permite capturar información contextual a lo largo de una secuencia [32]. Por tal motivo, estas redes neuronales pueden ser sumamente útiles para sistemas de autocompletado de texto, por ejemplo [29].

Una desventaja de los RNNs es que debido a su estructura recurrente pueden tener dificultades para capturar dependencias a largo plazo. Esto se debe a que la información se propaga a través de las diferentes capas, y en cada una se realizan transformaciones que pueden ocasionar que la información relevante se diluya o se pierda. Sin embargo, cabe destacar que existen arquitecturas RNN con ciertas modificaciones capaces de mitigar –hasta cierto punto– esta problemática, como las redes *long-short term memory* (LSTM) [29].

En general, las redes neuronales tradicionales tienen una limitación relacionada con la cantidad de información que pueden procesar y/o recordar. Sin embargo, existen otros tipos de modelos, como los los *large language models* (LLMs), o modelos de lenguaje grande, que tienen la capacidad de superar estas limitaciones. Estos modelos específicos tienen la característica de poder ser entrenados con grandes cantidades de datos (millones). Esto, obviamente, les permite poder capturar patrones más complejos en el lenguaje, así como también generar texto con mayor coherencia [1].

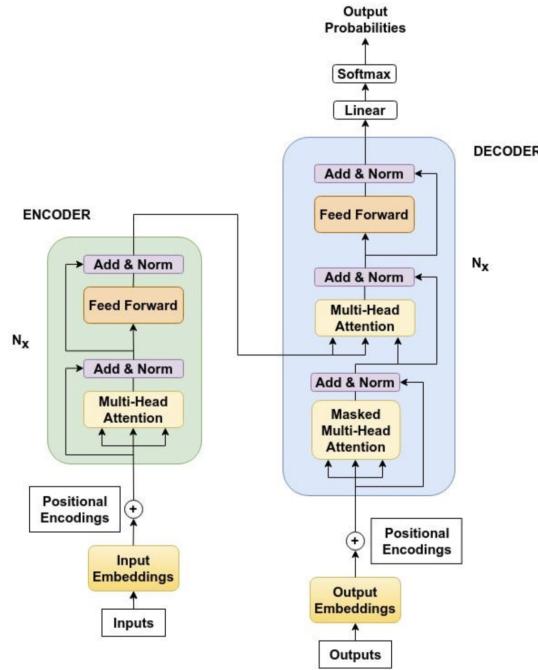


Figura 2: Arquitectura de transformer

Una de las arquitecturas más utilizadas para los LLMs es la del transformador. Un transformador es una red neuronal compuesta, comúnmente, por un *encoder* (codificador) y/o un *decoder* (decodificador). El *encoder* se encarga de procesar los *tokens* de entrada, determinando cuál es la relevancia relativa de cada uno en su contexto. A través de esto, dicho componente es capaz de crear representaciones contextuales de cada *token*. Por otro lado, el *decoder* utiliza el resultado del *encoder* para generar secuencias de salida. Sin embargo, en modelos que emplean únicamente *decoders*, estos asumen la responsabilidad de procesar la entrada y generar la salida de forma simultánea [36].

Cabe destacar que el *encoder* y *decoder* están estructurados de una forma similar. Por ejemplo, como se puede observar en la 2, ambos tienen capas llamadas *multi-head attention*, y *feed forward*. Las capas de *multi-head attention* le permiten al modelo poder enfocarse en diferentes partes de una secuencia simultáneamente, para así determinar cuál de todas es relevante. Por otro lado, las capas *feedforward* representan capas ocultas que procesan los datos y permiten al modelo aprender patrones complejos en las secuencias de entrada [15].

5.4. Generative Pre-trained Transformer (GPT)

Existen diversos modelos que utilizan la arquitectura de transformadores. Sin embargo, uno de los más populares es el *generative pretrained transformer* (GPT), o transformador generativo preentrenado. Este modelo fue presentado originalmente en el 2018 por OpenAI, en su artículo *Improving Language Understanding by Generative Pre-Training*. En esta publicación, los autores describen detalladamente la arquitectura del GPT. Además, resaltan que al entrenar un modelo con extensas cantidades de texto, este adquiere conocimientos sobre el mundo real y desarrolla la capacidad de procesar dependencias a largo plazo. Esto resulta en que el GPT sea capaz de responder preguntas de cualquier índole, así como clasificar texto [19].

En la actualidad, este modelo se utiliza principalmente en un *chatbot* desarrollado por OpenAI, conocido como ChatGPT. La versión gratuita de esta herramienta utiliza la tercera versión del modelo GPT para llevar a cabo interacciones conversacionales naturales con los usuarios. Este modelo fue preentrenado con más de 45 *terabytes* (o 45,000 *gigabytes*) de texto plano [23], en comparación con la versión piloto, que fue preentrenada únicamente con 40 *gigabytes* [33].

5.4.1. Fundamentos de GPTs

Arquitectura

Un GPT, a diferencia de otros transformadores, se distingue por su uso exclusivo de decodificadores en su arquitectura. De hecho, emplea múltiples decodificadores apilados, los cuales dependen de la salida del decodificador anterior para generar texto coherente y relevante [16]. Esta característica le permite al modelo capturar relaciones contextuales a diferentes niveles de abstracción, lo que, a su vez, le posibilita analizar y replicar el lenguaje natural [2].

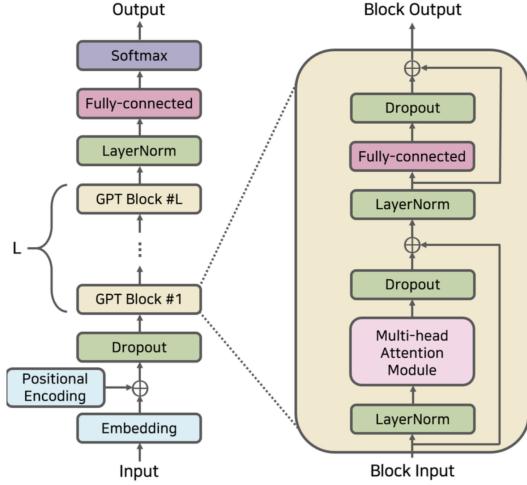


Figura 3: Arquitectura de Generative Pre-trained Transformer (GPT).

Como se puede observar en la 3, la entrada de estos modelos primero pasa por una capa de *embedding*. Como se mencionó en secciones anteriores, en este proceso se convierten las palabras o frases en vectores numéricos. Cabe destacar que la entrada está limitada a 2048 palabras, por lo cual también se utiliza *padding* para ajustar las secuencias más cortas y asegurar una longitud uniforme antes de ingresarlas al modelo. Al concluir esa etapa, se continúa a la fase de *positional encoding*, en la cual se asignan vectores de posición a cada vector numérico con el propósito de poder identificar su ubicación en la secuencia [34].

Después de algunas capas variables, se procede a la utilización de los decodificadores apilados. Cabe destacar que todos están estructurados de una forma similar, conteniendo capas de normalización, *dropout*, *multi-head attention*, y *fully connected* (o *feed forward*). Posteriormente, el resultado del último *decoder* se normaliza y se pasa a una capa oculta, seguido por una función *softmax*. Esta última genera una distribución de probabilidad, la cual ayuda a identificar cuál es la próxima palabra más probable en la secuencia, según el contexto proporcionado por el modelo [34].

Pre-entrenamiento

Con base en la arquitectura previamente descrita, se puede llevar a cabo el proceso de preentrenamiento del modelo. Esta etapa tiene este nombre debido a que se le está enseñando al modelo a comprender y generar texto, sin una tarea específica en mente. Por tal razón, se puede utilizar cualquier tipo de texto, desde artículos de prensa y libros, hasta transcripciones de videos y audios [33].

En cada paso del preentrenamiento, el modelo recibe una frase incompleta como ejemplo. Este tiene la tarea de predecir la siguiente palabra o token considerando el contexto proporcionado. Si la predicción es correcta, el modelo continúa sin necesidad de ajustes. Sin embargo, si la predicción no es la esperada, el modelo realiza ajustes internos para darle prioridad a esa palabra en particular, considerando el contexto de la frase. Al realizar este proceso reiteradas veces, el modelo se vuelve capaz de generar texto coherente y relevante [34].

5.4.2. Fine-Tuning

Fine-tuning, o ajuste fino, es un proceso a través del cual se adapta un modelo preentrenado para realizar una tarea específica. Para esto, se le proporciona a un modelo ejemplos de entrada y salida relacionados a la tarea que se desea enseñar. Por ejemplo, para un traductor de idiomas, estas serían frases en el idioma de origen y la traducción correspondiente en el idioma destino. A partir de estos, el modelo ajusta los pesos de sus capas neuronales, los cuales son parámetros internos que determinan cómo se combinan las características de los datos de entrada para producir la salida. Este ajuste tiene como objetivo minimizar la diferencia entre las predicciones del modelo y los resultados reales, permitiendo que el modelo aprenda a resolver la nueva tarea de manera efectiva [26].

Es importante mencionar que la cantidad de datos para el proceso de ajuste fino depende de la complejidad de la tarea. Para tareas simples puede ser suficiente decenas o cientos de ejemplos, mientras que para tareas más complejas se pueden necesitar miles o incluso millones de ejemplos para lograr un ajuste preciso del modelo. En otras palabras, a mayor complejidad usualmente se requiere una cantidad proporcionalmente mayor de datos [26].

5.4.3. Prompt Engineering

En ciertos casos, los modelos pueden tener dificultades para aprender nuevas tareas a través del *fine-tuning*, lo que puede resultar en respuestas incoherentes o inesperadas. Para abordar este tipo de problemas, se utiliza el *prompt engineering*, una técnica que busca mejorar la formulación de las instrucciones dadas al modelo, de manera que interprete correctamente la tarea y genere respuestas más precisas. Más específicamente, esta técnica consiste en estructurar cuidadosamente las entradas, proporcionando ejemplos claros, instrucciones detalladas o indicaciones específicas sobre el formato y el contenido esperado. Esto permite guiar al modelo para que genere respuestas más consistentes y alineadas con los objetivos deseados [39].

Una ventaja importante del *prompt engineering* es que, en algunos casos, permite que el modelo realice nuevas tareas sin necesidad de un *fine-tuning*. Sin embargo, para tareas más complejas, esta técnica por sí sola puede no ser suficiente. Por lo tanto, combinar *prompt engineering* con *fine-tuning* suele ser más efectivo. Mientras que el *fine-tuning* adapta el modelo a tareas específicas, el *prompt engineering* mejora la formulación de las instrucciones para optimizar el rendimiento. Utilizar ambas técnicas de manera complementaria maximiza las capacidades del modelo y mejora los resultados en tareas más desafiantes [39].

5.5. Explainable Artificial Intelligence (XAI)

La Inteligencia Artificial Explicable (XAI), es un conjunto de técnicas que buscan hacer comprensibles las decisiones y predicciones de los modelos de inteligencia artificial. Esto resulta especialmente útil en modelos de *deep learning* (como *transformadores*) ya que estos sistemas, debido a su compleja arquitectura, funcionan como “cajas negras”. En otras palabras, el proceso mediante el cual se generan las predicciones no es fácilmente interpretable por los usuarios, lo que dificulta identificar cómo ciertas entradas llevan a un resultado específico [12].

Para abordar este desafío de interpretabilidad, las técnicas de XAI se han desarrollado siguiendo dos enfoques principales: intrínseco y *post-hoc*. El enfoque intrínseco integra la explicabilidad en el diseño del modelo desde su creación, incorporando mecanismos de transparencia durante el proceso de desarrollo. Estos modelos se construyen con arquitecturas inherentemente interpretables, como árboles de decisión o reglas lógicas. Por otro lado, el enfoque *post-hoc* ofrece interpretaciones tras el entrenamiento del modelo, sin alterar su estructura interna [12]. Entre las herramientas *post-hoc*

más destacadas se encuentra LIME (*Local Interpretable Model-agnostic Explanations*), que permite visualizar la contribución relativa de cada variable de entrada en las predicciones de un modelo, ayudando así a entender qué características tienen mayor influencia en los resultados generados [11].

5.5.1. Local Interpretable Model-agnostic Explanations (LIME)

LIME es una técnica *post-hoc* introducida por Ribeiro et al. en el 2016. Su principal característica es que puede aplicarse a cualquier tipo de modelo, lo que la hace *model-agnostic*. Además, proporciona explicaciones locales, es decir, enfocadas en cada predicción individual [28].

Para generar estas explicaciones locales, LIME utiliza un proceso basado en la perturbación de los datos de entrada. La idea principal es crear múltiples versiones modificadas de los datos originales que se desean procesar para generar una predicción. Este proceso implica alterar sistemáticamente las características, ya sea variando valores numéricos o eliminando elementos. Por ejemplo, si se trabaja con características numéricas, las perturbaciones pueden consistir en variar los valores dentro de ciertos rangos o eliminarlos temporalmente. El objetivo es crear un conjunto de variaciones que permitan observar cómo el modelo responde a diferentes combinaciones de las características de entrada [37].

Una vez generadas estas perturbaciones, cada entrada modificada se procesa a través del modelo original para obtener nuevas predicciones. Al comparar cómo cambia la salida del modelo para cada perturbación respecto a la predicción original, es posible determinar la importancia relativa de cada característica en la decisión final [37]. Este análisis permite identificar qué elementos de los datos de entrada tienen mayor influencia en las predicciones del modelo, revelando patrones y dependencias que no son evidentes a simple vista. Los resultados proporcionan una comprensión clara de cómo diferentes variables afectan el resultado final, lo que resulta fundamental para validar si el modelo está basando sus decisiones en características relevantes [28].

CAPÍTULO 6

Antecedentes

Actualmente, existen diversos proyectos dedicados a abordar la problemática planteada en este trabajo. No obstante, la mayoría están diseñados específicamente para la lengua de señas estadounidense o india. Además, estos enfoques utilizan analizadores sintácticos y la definición de gramáticas específicas para organizar las palabras de manera que se generen oraciones coherentes en el idioma destino. Sin embargo, estos métodos presentan limitaciones significativas, como la complejidad inherente a la definición de una gramática libre de contexto y la capacidad restringida para manejar oraciones largas o complejas. Como resultado, estos sistemas tienden a ser eficientes únicamente con oraciones cortas, lo que limita su versatilidad y aplicabilidad en escenarios más complejos [38][18].

Considerando estas limitaciones, en China surgió una propuesta innovadora que consistió en adaptar un modelo de lenguaje natural preentrenado para corregir errores gramaticales y sintácticos en oraciones en chino mediante la técnica de *fine-tuning*. Los autores de este proyecto crearon un conjunto de datos desde cero, compuesto por oraciones gramaticalmente incorrectas y sus correspondientes versiones corregidas. Para ampliar este conjunto de datos, aprovecharon otros modelos, como ChatGPT, solicitando la generación de ejemplos adicionales que luego corrigieron manualmente. Este enfoque permitió entrenar un modelo que no solo comprendía la gramática china, sino que también adquirió la capacidad de corregirla de manera efectiva, superando las limitaciones de los métodos tradicionales basados en gramáticas y analizadores sintácticos [14].

CAPÍTULO 7

Metodología

Para garantizar el cumplimiento de los objetivos establecidos en el proyecto, se desarrolló una metodología que comprendió varias etapas. En primer lugar, se recopilaron textos en español que aplicaban la gramática de LENSEGUÁ. Posteriormente, se llevó a cabo el proceso de *fine-tuning* del modelo GPT-3.5-Turbo utilizando dicho conjunto de frases. Luego, se exploraron y aplicaron técnicas de *prompt engineering* para diseñar *prompts* específicos que orientaran al modelo en la generación de interpretaciones más precisas. Por último, se procedió a evaluar la calidad y coherencia de las interpretaciones generadas.

7.1. Creación de conjunto de datos

7.1.1. Recolección de datos

El proceso de recolección de datos consistió en buscar y seleccionar diversas frases en español que aplicaran los principios gramaticales de LENSEGUÁ. Para ello, se exploraron recursos en línea, como grupos en redes sociales donde usuarios sordos comparten sus pensamientos y experiencias utilizando esta gramática en sus interacciones escritas. En particular, se identificó un grupo privado en la red social Facebook llamado 'Lengua de Señas de Guatemala (LSG)'. Se envió una solicitud a los administradores del grupo para obtener acceso, y, una vez aprobada, se procedió a recopilar diversos comentarios que aplicaban esta gramática. No obstante, para proteger la privacidad de los usuarios, no se almacenó ninguna información personal, y se excluyeron comentarios que contuvieran información sensible o identificativa. Tras la recolección y almacenamiento de los datos en un archivo CSV, se generaron -de forma manual- las versiones gramaticalmente correctas en español de cada frase recopilada. En el Cuadro 1 se presenta la estructura utilizada para almacenar la información, mostrando ejemplos de frases en LENSEGUÁ y sus correspondientes interpretaciones en español.

| LENSEGUA | Español |
|---------------------------------|----------------------------|
| aeropuerto dónde pregunta | ¿Dónde está el aeropuerto? |
| antes tu estar enojado pregunta | ¿Estabas enojado? |
| haber carro mucho | Hay tráfico. |
| pasado yo pasaporte perder | Perdí mi pasaporte. |
| ... | ... |

Tabla 1: Estructura del conjunto de datos desarrollado.

Seguidamente, se establecieron contactos con organizaciones guatemaltecas que trabajan con la comunidad sorda, entre ellas la Asociación Educativa para el Sordo (ASEDES). Esta organización colaboró activamente en la generación de más ejemplos específicos que ilustraban la aplicación de la gramática de LENSEGUA en diferentes contextos, desde situaciones cotidianas hasta emergencias. Ellos se centraron en utilizar un vocabulario que tuviera equivalentes directos en LENSEGUA, ya que no todas las palabras en español tienen una traducción literal en lengua de señas. Además, colaboraron con la creación de las versiones gramaticalmente correctas en español para cada una de estas frases, manteniendo la misma estructura descrita anteriormente.

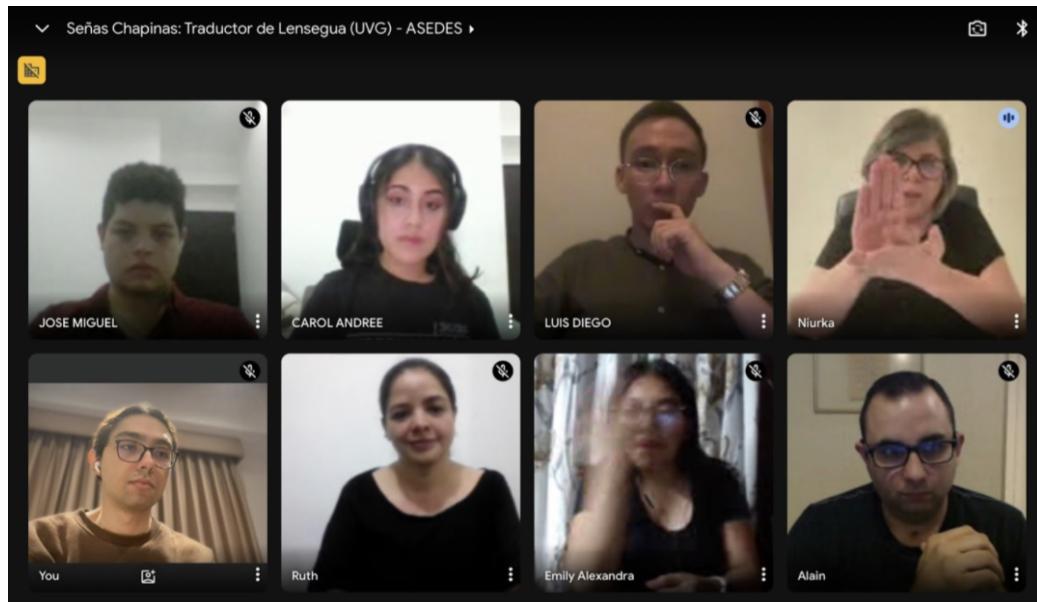


Figura 4: Reunión de equipo de trabajo con ASEDES.

Durante esta etapa, se optó por recibir un curso virtual sobre la gramática de LENSEGUA ofrecido por la organización Sordos Latinos Guatemala. Este curso proporcionó una comprensión detallada de las reglas y estructuras gramaticales específicas de LENSEGUA, lo que permitió generar ejemplos adicionales -propios- con mayor precisión. Además, la capacitación también facilitó la validación de los textos anteriormente obtenidos, asegurando que el contenido estuviera alineado con las normas gramaticales de LENSEGUA.

Por último, se tomó como referencia en el trabajo de Fan *et al.* (2023) [14] y se procedió a utilizar la plataforma de Claude.ai para generar más datos. Más específicamente, se le proporcionó a esta herramienta de generación de texto (basada en inteligencia artificial) el archivo CSV anteriormente desarrollado y se solicitó que generara entradas adicionales para ampliar el *dataset*. Este proceso tuvo como objetivo incrementar tanto la cantidad como la diversidad de los ejemplos disponibles en

el *dataset*. Posteriormente, se revisaron las frases generadas para asegurar que cumplieran con las normas gramaticales de LENSEGUA y se verificó que no hubiera duplicados en el contenido.

Cabe destacar que no se utilizó ChatGPT para generar los datos adicionales, ya que no se considera óptimo *fine-tune*ar un modelo con salidas generadas por la misma herramienta. Este acercamiento no aportaría nueva información y resultaría en redundancia en el aprendizaje [14]. Por ello, se eligió Claude.ai, una plataforma independiente, para garantizar que los datos fueran verdaderamente nuevos, permitiendo así que el proceso de *fine-tuning* en GPT-3.5-Turbo se basara en entradas totalmente nuevas y variadas.

7.1.2. Data augmentation

Una vez generados los datos iniciales, se procedió a dividir el conjunto de datos en dos subconjuntos: un 80 % se destinó al entrenamiento y un 20 % a la evaluación. Esta división fue necesaria para evaluar el rendimiento del modelo de manera efectiva y garantizar que el modelo final no se sobreajustara a los datos de entrenamiento.

Después de esta separación, se desarrolló un programa en Python con el propósito de realizar *data augmentation* en el conjunto de entrenamiento. Es decir, ampliar el *dataset* existente mediante la generación de nuevas frases. El enfoque consistió en crear variaciones de las frases originales mediante la sustitución de palabras por otras similares, manteniendo la coherencia contextual. Por ejemplo, una oración como ‘papá cirujano ser’ podría transformarse en ‘abuelo doctor ser’ y ‘tío odontólogo ser’, entre otras (ver la Figura 5).

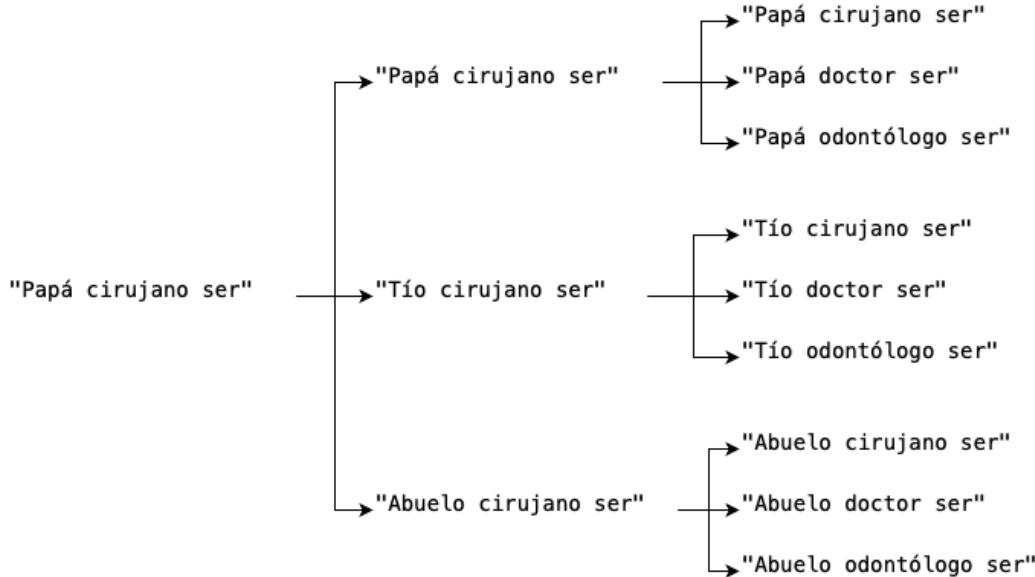


Figura 5: Generación recursiva de nuevas frases.

Para ello, se definió un conjunto de listas que agrupaban palabras intercambiables, como profesiones, lugares, objetos, entre otros. El programa analizaba cada oración del conjunto de entrenamiento y, al identificar términos presentes en estas listas, generaba todas las posibles combinaciones de oraciones mediante sustituciones recursivas. De esta manera, si una oración contenía varias palabras intercambiables, se creaban múltiples variantes de la misma, ampliando significativamente el *dataset* de entrenamiento.

Es importante mencionar que el sistema de *data augmentation* también modificaba las interpretaciones teóricas definidas en el *dataset*. Esto aseguraba que cada nueva frase en LENSEGUA tuviera su correspondiente versión en español, aumentando así la cantidad de datos disponibles para el *fine-tuning* del modelo GPT-3.5-Turbo.

7.1.3. Preparación del conjunto de datos

Con el *dataset* de entrenamiento ampliado y el conjunto de evaluación definido, se procedió a preparar los datos utilizando Python. Este proceso incluyó la corrección automatizada de mayúsculas y la adición de signos de puntuación, entre otros ajustes. Además, se organizaron los datos en un formato compatible con OpenAI, utilizando un formato JSON con la siguiente estructura:

```
1 'messages': [
2     {'role': 'system', 'content': DEFAULT_SYSTEM_PROMPT},
3     {'role': 'user', 'content': question + ' ->'},
4     {'role': 'assistant', 'content': answer},
5 ]
```

En esta estructura, el *DEFAULT SYSTEM PROMPT* constituye el mensaje inicial que establece el contexto y las directrices fundamentales para el comportamiento del modelo. El campo *question* contiene la frase que aplica la gramática de LENSEGUA, mientras que el campo *answer* proporciona la interpretación gramaticalmente correcta en español.

Al finalizar con la preparación, los datos procesados se guardaron en archivos JSONL, asegurando así su disponibilidad para el siguiente paso en el proceso de *fine-tuning* del modelo.

7.2. Fine-tuning

7.2.1. Identificación de parámetros óptimos

Para asegurar un rendimiento óptimo del modelo *fine-tuneado* GPT-3.5-Turbo, se realizaron pruebas preliminares con una porción del conjunto de datos (se excluyeron las frases generadas mediante *data augmentation*). Estas pruebas permitieron experimentar con distintas configuraciones para identificar los parámetros más adecuados. Al emplear un subconjunto del *dataset*, se optimizó el proceso sin incurrir en los altos costos asociados a múltiples entrenamientos completos, logrando además acelerar el proceso ajuste fino.

Los parámetros ajustados incluyeron:

- **Epochs:** Número de veces que el modelo recorre el conjunto de datos de entrenamiento completo. Ajustar este parámetro fue clave para evitar tanto el subentrenamiento como el sobreentrenamiento.
- **Batch size:** Cantidad de muestras procesadas antes de actualizar los parámetros del modelo. Valores pequeños favorecen la generalización, mientras que valores grandes aceleran el entrenamiento.
- **Learning Rate Multiplier:** Controla la magnitud de los ajustes que realiza el modelo en función de los errores durante el entrenamiento. Un ajuste adecuado de este parámetro fue crucial para asegurar la convergencia del modelo.

7.2.2. Fine-tuning del modelo GPT-3.5-Turbo

Una vez identificados los parámetros óptimos, se realizó el *fine-tuning* del modelo GPT-3.5-Turbo en la plataforma de OpenAI. Para ello, se creó una nueva tarea de *fine-tuning* y se seleccionó el modelo base correspondiente. Luego, se cargaron los archivos JSONL generados previamente. El archivo de entrenamiento incluía 4,192 entradas, combinando frases originales y aquellas generadas mediante *data augmentation*. Por su parte, el archivo de validación contenía 200 entradas para evaluar el rendimiento del modelo durante el ajuste. Finalmente, se configuraron los parámetros y se inició el proceso de *fine-tuning*.

Durante el proceso de ajuste, se monitoreó la pérdida de entrenamiento y validación para verificar que el modelo no presentara *overfitting*. El *overfitting* ocurre cuando un modelo aprende demasiado bien los datos de entrenamiento, afectando así su capacidad de generalización.

7.2.3. Evaluación del modelo GPT-3.5-Turbo fine-tuneado

Distancia de Levenshtein

Tras completar el proceso de *fine-tuning*, se procedió a evaluar el desempeño del modelo en la generación de interpretaciones utilizando la distancia de Levenshtein. Esta métrica mide el número mínimo de operaciones necesarias (inserciones, eliminaciones o sustituciones) para transformar una cadena de texto en otra [31]. Se consideró que esta métrica era particularmente adecuada para este contexto, ya que el objetivo era que el modelo generara interpretaciones sin introducir variaciones significativas en el vocabulario ni en la estructura de las frases.

Para llevar a cabo esta evaluación, se desarrolló un programa en Python que permitió al modelo *fine-tuneado* generar interpretaciones para cada frase del conjunto de validación, utilizando un *prompt* básico que definía el rol del sistema como intérprete de LENSEGUA. A partir de esto, se compararon las interpretaciones generadas con las interpretaciones teóricas predefinidas y se calculó la distancia de Levenshtein para cada par. Finalmente, se promediaron las 200 distancias obtenidas, lo que permitió determinar un valor representativo del desempeño del modelo.

Con el fin de establecer una línea de comparación, se repitió el análisis con el modelo GPT-3.5-Turbo estándar, utilizando las mismas 200 frases y el mismo *prompt*. En este caso, el modelo que presentó la menor distancia promedio fue considerado el más efectivo para la tarea de interpretación.

Local Interpretable Model-agnostic Explanations (LIME)

La técnica de *Local Interpretable Model-agnostic Explanations* (LIME) se utilizó para obtener una comprensión más profunda del impacto del *fine-tuning*. LIME permitió identificar, para frases individuales, qué palabras tuvieron mayor influencia en la interpretación generada por cada modelo. De este modo, se buscaba determinar cómo los elementos característicos de LENSEGUA influían en las respuestas de cada modelo, y si el modelo *fine-tuneado* mostraba una recepción más favorable hacia estos elementos en comparación con el modelo estándar.

El análisis se realizó utilizando Python y la librería LIME. A este programa se le proporcionaron tres frases de prueba, entre ellas “pasado yo ir no”. Cada frase fue sometida a un proceso de perturbación, que consistió en generar cien variaciones mediante la eliminación aleatoria de palabras. Estas frases modificadas se enviaron a ambos modelos (el modelo *fine-tuneado* y el modelo GPT-3.5-Turbo estándar) para que generaran sus interpretaciones utilizando el *prompt* básico previamente mencionado.

El Cuadro 2 presenta un ejemplo del resultado este proceso. En la primera fila, se muestra la frase seleccionada y su correspondiente interpretación teórica, que en este caso es “No fui”. Seguidamente, se detallan algunas perturbaciones generadas a partir de la frase original, junto con las interpretaciones producidas por el modelo *fine-tuneado*. Es importante mencionar que este mismo procedimiento se realizó utilizando el modelo GPT-3.5-Turbo estándar, generando así un conjunto comparable de interpretaciones para cada perturbación.

Para comparar las interpretaciones generadas con respecto a las interpretaciones teóricas, se utilizó la distancia de Levenshtein normalizada. Este valor, que oscila entre 0 y 1, permite determinar la similitud de entre dos frases. Un valor de 1 indica que las frases son idénticas, mientras que valores más bajos reflejan mayores diferencias.

| Frase en LENSEGUA: pasado yo ir no | | |
|------------------------------------|----------------|----------------------------------|
| Interpretación teórica: No fui. | | |
| Frase / Perturbación | Interpretación | Distancia de Levenshtein (norm.) |
| pasado yo ir no | No fui. | 1.0 |
| yo | Yo. | 0.2857 |
| pasado yo no | No fui. | 1.0 |
| pasado | En el pasado. | 0.1538 |
| ... | ... | ... |

Tabla 2: Ejemplos de perturbaciones interpretadas por el modelo *fine-tuneado* junto con sus distancias de Levenshtein normalizadas correspondientes.

A partir de los resultados obtenidos mediante las perturbaciones y sus correspondientes distancias de Levenshtein normalizadas, LIME permitió cuantificar la influencia de cada palabra en las interpretaciones generadas tanto por el modelo estándar como el *fine-tuneado*. Un valor positivo indicaba que una palabra ayudaba al modelo a generar interpretaciones más cercanas a la teórica, mientras que un valor bajo o negativo sugería que su presencia confundía al modelo, resultando en respuestas más distantes de lo esperado.

7.3. Prompt engineering

Durante la evaluación del modelo *fine-tuneado*, se identificaron varios aspectos que afectaron negativamente la calidad de las interpretaciones generadas. En particular, se observaron dificultades en el manejo de expresiones temporales, lo que llevaba a errores de interpretación y redundancias en las respuestas. Para abordar estas limitaciones, se implementó una fase de *prompt engineering* con el objetivo de mejorar la coherencia y precisión de las interpretaciones.

Se desarrollaron nuevas versiones del *prompt* de manera iterativa, incorporando instrucciones más precisas y delimitadores específicos. Cada nueva versión se diseñó a partir de la evaluación de la versión anterior, lo que permitió un refinamiento continuo de las instrucciones. Las modificaciones incluyeron la adición de reglas explícitas para el procesamiento de expresiones características de LENSEGUA, así como ejemplos contextuales. Estos cambios se orientaron a potenciar la capacidad del modelo para interpretar correctamente el contexto y a minimizar las ambigüedades en las respuestas.

Para evaluar cada *prompt*, se utilizó el programa de Python previamente desarrollado en la fase de *fine-tuning*. Este programa permitió generar interpretaciones para cada frase del conjunto de validación, utilizando las distintas versiones de los *prompts*. A partir de estas interpretaciones, se calculó la distancia de Levenshtein promedio para medir la similitud entre las respuestas generadas

y las interpretaciones teóricas predefinidas. De esta manera, se logró determinar una distancia promedio asociada a cada *prompt*, lo que proporcionó una medida cuantitativa del rendimiento de cada versión. Cabe destacar que este proceso de evaluación se realizó tanto con el modelo *fine-tuneado* como con el modelo GPT-3.5-Turbo estándar, lo que facilitó la comparación del impacto de cada *prompt* en ambos modelos.

7.4. Implementación y evaluación del sistema

Uno de los objetivos principales de este proyecto fue que miembros de la comunidad sorda evaluaran la coherencia y precisión de las interpretaciones generadas por el modelo *fine-tuneado*. Para ello, se desarrolló una plataforma web que le permitió a usuarios interactuar directamente con el sistema y evaluar su rendimiento en tiempo real. Además, se diseñó una encuesta dirigida a intérpretes, estudiantes y usuarios familiarizados con LENSEGUA, con el propósito de calificar el desempeño del modelo *fine-tuneado* en comparación con la versión estándar de GPT-3.5-Turbo. Esta combinación de interacción directa a través de la plataforma web y una evaluación estructurada proporcionó una visión completa sobre la efectividad de las interpretaciones generadas, ofreciendo una base sólida para analizar el rendimiento del sistema.

7.4.1. Desarrollo de plataforma web

Para facilitar la interacción con el modelo *fine-tuneado*, se desarrolló una plataforma web local (no accesible desde el internet) para que permitía a los usuarios probar el sistema de manera controlada. Esta aplicación se estructuró en dos componentes principales: el *backend*, responsable del procesamiento de las solicitudes a través de Python y Flask, y el *frontend*, encargado de la visualización de los resultados en tiempo real.

El *backend* del sistema estableció una conexión con el API de OpenAI, lo que permitió la interacción directa tanto con el modelo *fine-tuneado* como con otros modelos de OpenAI. Además, se encargó de gestionar las solicitudes de los usuarios: cuando un usuario introducía una frase, el *backend* enviaba la solicitud a la API, procesaba la respuesta generada por el modelo y devolvía la interpretación correspondiente. Se implementó también un sistema de almacenamiento de historial de solicitudes, que permitió a los usuarios visualizar el registro completo de sus consultas.

El *frontend* se desarrolló utilizando HTML y CSS, lo que permitió crear una interfaz de usuario intuitiva y eficiente. La página web adoptó un formato similar al de un *chat*, facilitando a los usuarios la entrada de frases y la recepción de respuestas en tiempo real. Además, se incluyó la opción de seleccionar entre el modelo *fine-tuneado* y el modelo GPT-3.5-Turbo estándar. La interfaz fue optimizada para dispositivos de escritorio, garantizando así una experiencia clara y amigable.

7.4.2. Desarrollo de encuesta

Para evaluar la coherencia de las frases generadas por el modelo *fine-tuneado*, se creó una encuesta mediante Google Forms. La encuesta comenzaba con un consentimiento informado que detallaba el propósito del estudio y el proceso de evaluación, asegurando que los participantes comprendieran plenamente la metodología antes de iniciar.

La encuesta se organizó en secciones, cada una enfocada en evaluar una frase específica mediante tres preguntas. Las dos primeras preguntas solicitaban a los participantes que calificaran, en una escala del 1 al 5 (donde 1 representaba baja coherencia y 5 alta coherencia), las interpretaciones generadas tanto por el modelo estándar GPT-3.5-Turbo como por el modelo *fine-tuneado*. La tercera

pregunta requería una comparación directa entre ambas interpretaciones, pidiendo a los participantes que seleccionaran la que consideraban más coherente o adecuada para la frase en cuestión.

Además, la encuesta incluía una sección adicional dedicada a evaluar los diferentes *prompts* desarrollados. En esta sección, los participantes debían seleccionar, entre varias opciones, la interpretación que encontraban más clara y comprensible. Todas estas interpretaciones fueron generadas por el modelo *fine-tuneado*, pero utilizando diferentes *prompts*. Esta metodología permitió no solo evaluar la coherencia general del modelo *fine-tuneado*, sino también validar cuál *prompt* resultaba más efectivo para la generación de interpretaciones precisas.

Es fundamental mencionar que la encuesta se limitó a participantes que eran personas sordas, familiares de personas sordas, estudiantes de LENSEGUА o intérpretes. Esta selección garantizó que las evaluaciones provinieran de individuos con un conocimiento relevante sobre la lengua, lo que contribuyó a la validez de los resultados obtenidos.

7.4.3. Pruebas con usuarios

En la etapa final de evaluación del modelo, se organizó una reunión presencial con la asociación guatemalteca de sordos En-Señas. La sesión inició con una presentación detallada del modelo *fine-tuneado*, en la que se explicó su propósito y funcionamiento. Además, los participantes, en su mayoría personas sordas y un intérprete, pudieron observar el sistema en acción y hacer preguntas sobre su desarrollo. Posteriormente, se les pidió completar la encuesta previamente descrita, donde evaluaron la coherencia de las interpretaciones generadas por el modelo. Esta interacción directa fue clave para recopilar comentarios y sugerencias en tiempo real, lo que contribuyó a una evaluación más completa del sistema. Las respuestas de estos participantes, como se detallará más adelante, permitieron confirmar la efectividad del modelo desarrollado, resaltando su utilidad en la interpretación de frases que aplican la gramática de LENSEGUА.



Figura 6: Reunión presencial en En-Señas donde se presentó la solución desarrollada.

Para ampliar el alcance de la evaluación, la encuesta se compartió también con otras organizaciones y clubes de LENSEGUА. Esto permitió obtener un conjunto más amplio de respuestas, enriqueciendo así el análisis de la coherencia y utilidad del modelo *fine-tuneado*.

CAPÍTULO 8

Resultados

En este capítulo se presentan los resultados obtenidos durante las distintas etapas del proyecto, desde el proceso de *fine-tuning* del modelo GPT-3.5-Turbo hasta la fase de *prompt engineering*. También se incluyen todos los *prompts* desarrollados a lo largo del proceso, así como las distancias de Levenshtein calculadas para evaluar la similitud entre las interpretaciones generadas y las teóricas. A través de estos resultados, se busca ofrecer una visión clara del impacto de cada etapa en el logro de los objetivos establecidos.

8.1. Fine-tuning

Como se mencionó anteriormente, antes de proceder con el *fine-tuning* del modelo GPT-3.5-Turbo, se llevaron a cabo experimentos preliminares para identificar los parámetros que generaban la menor pérdida de validación. Esta métrica indica la eficacia del modelo para predecir correctamente las salidas a partir de las entradas proporcionadas. Una menor pérdida de validación sugiere que el modelo está aprendiendo de manera efectiva y puede generalizar mejor a datos no vistos. En el Cuadro 3, se presentan los valores de los parámetros ajustados durante estas pruebas, junto con la pérdida de validación correspondiente a cada configuración.

| Epochs | Batch Size | Learning Rate Multiplier | Pérdida de Validación |
|--------|------------|--------------------------|-----------------------|
| 2 | 16 | 0.2 | 0.1267 |
| 1 | 16 | 0.3 | 0.1334 |
| 1 | 8 | 0.5 | 0.2383 |
| 2 | 1 | 2 | 0.3163 |
| 3 | 1 | 2 | 0.4418 |
| 6 | 1 | 2 | 0.5966 |

Tabla 3: Valores de parámetros y pérdidas de validación en experimentos preliminares

Tras analizar los resultados de estas pruebas, se decidió llevar a cabo el *fine-tuning* final utilizando el conjunto de datos de entrenamiento completo y los parámetros que demostraron generar la menor pérdida de validación. Los parámetros seleccionados fueron: 2 *epochs*, un *batch size* de 16 y un *learning rate multiplier* de 0.2.

| Parámetro | Valor |
|--------------------------|-------|
| Epochs | 2 |
| Batch size | 16 |
| Learning Rate Multiplier | 0.2 |

Tabla 4: Parámetros utilizados para el *fine-tuning* del modelo GPT-3.5-Turbo.

Durante el proceso de *fine-tuning*, la plataforma de OpenAI generó una gráfica que se actualizaba con cada paso del entrenamiento, lo que facilitó el monitoreo de la evolución de la pérdida en tiempo real. La gráfica final, presentada en la Figura 7, confirmó que el modelo no presentaba signos de *overfitting*, es decir que el modelo resultante tenía la capacidad para adaptarse a nuevos datos. Esta observación fue fundamental para garantizar que el modelo pudiera aplicarse efectivamente en situaciones prácticas.

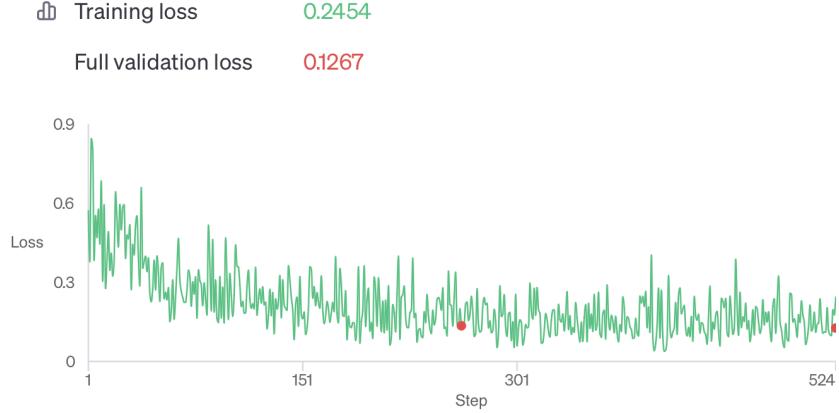


Figura 7: Evolución de la pérdida durante el *fine-tuning* final del modelo GPT-3.5-Turbo.

Una vez completado el *fine-tuning* del modelo GPT-3.5-Turbo, se utilizó la técnica de *Local Interpretable Model-agnostic Explanations* (LIME) para evaluar el impacto de este proceso en las interpretaciones generadas por el modelo. LIME permitió identificar, para frases específicas, qué palabras influían más en las respuestas de ambos modelos, facilitando el análisis de cómo cada uno priorizaba ciertos elementos del texto al generar sus respuestas. Para ello, se seleccionaron tres frases representativas de la gramática de LENSEGUA, diseñadas para evaluar la capacidad de los modelos en interpretar estructuras lingüísticas clave, como marcadores temporales, términos interrogativos y otras palabras funcionales.

Las Figuras 8 y 9 presentan los resultados de los análisis LIME realizados para la frase “antes tu policía llamar pregunta”. Cada visualización proporciona información detallada sobre el impacto de cada palabra en las respuestas generadas por ambos modelos, codificándolas por colores y asociándolas a un valor numérico específico. En ambas figuras, las palabras resaltadas en azul (con valores cercanos a -1) fueron aquellas que generaron confusión, resultando en respuestas menos alineadas con la interpretación teórica establecida. En contraste, las palabras resaltadas en naranja (con valores cercanos a 1) contribuyeron de manera positiva, ayudando a formular interpretaciones más precisas.

Estos resultados se resumen en el Cuadro 5, donde destaca un cambio significativo en la palabra “pregunta”, que pasó de tener una influencia negativa (-0.09) en el modelo estándar a una positiva (0.09) en el modelo adaptado. Esta transformación sugiere que el *fine-tuning* mejoró la capacidad del modelo para reconocer y procesar marcadores interrogativos. Adicionalmente, las palabras “policía” y “llamar” también mostraron un aumento en su relevancia en el modelo adaptado (0.43 y 0.20, respectivamente), en comparación con el modelo estándar (0.13 y 0.07).

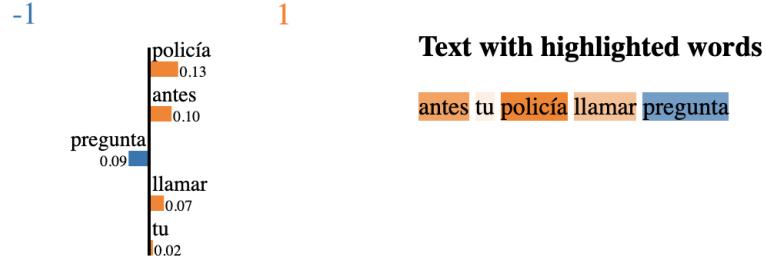


Figura 8: Resultados de LIME para la interpretación generada por el modelo GPT-3.5-Turbo estándar para la frase “antes tu policía llamar pregunta”.

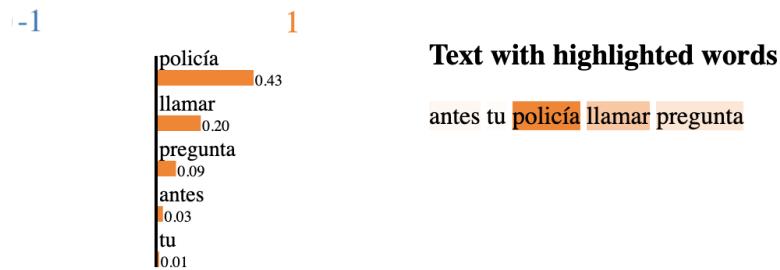


Figura 9: Resultados de LIME para la interpretación generada por el modelo *fine-tuneado* para la frase “antes tu policía llamar pregunta”.

| Frase en LENSEGUA: antes tu policía llamar pregunta | | |
|---|------------------------------|----------------------------|
| Interpretación teórica: ¿Llamaste a la policía? | | |
| Palabra | Valores de Influencia (LIME) | |
| | Modelo GPT-3.5-Turbo | Modelo <i>fine-tuneado</i> |
| antes | 0.10 | 0.03 |
| tu | 0.02 | 0.01 |
| policía | 0.13 | 0.43 |
| llamar | 0.07 | 0.20 |
| pregunta | -0.09 | 0.09 |

Tabla 5: Contribución de cada palabra según los Valores de Influencia (LIME) en las interpretaciones generadas por el modelo GPT-3.5-Turbo estándar y su versión *fine-tuneada* para la frase “antes tu policía llamar pregunta”.

En el Cuadro 6 se presentan los resultados para la frase “futuro hospital él ir”. El marcador temporal “futuro” mostró el cambio más significativo, pasando de una influencia negativa de -0.04

en el modelo estándar (ver Figura 10) a una positiva de 0.09 en el modelo adaptado (ver Figura 11). Esto indica que el modelo ajustado reconoció mejor la importancia de “futuro” como un elemento clave para la interpretación de la frase. Por otro lado, la palabra “hospital” aumentó su relevancia (de 0.34 a 0.51), mientras que las influencias de “él” e “ir” experimentaron una ligera reducción.

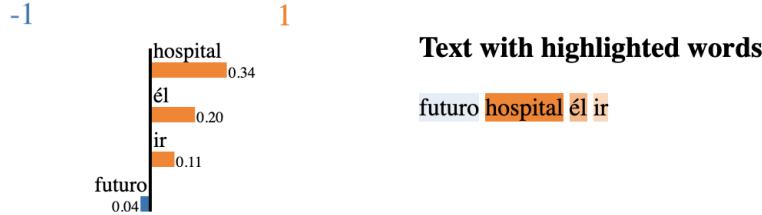


Figura 10: Resultados de LIME para la interpretación generada por el modelo GPT-3.5-Turbo estándar para la frase “futuro hospital él ir”.



Figura 11: Resultados de LIME para la interpretación generada por el modelo *fine-tuneado* para la frase “futuro hospital él ir”.

| Frase en LENSEGUA: futuro hospital él ir | | |
|---|------------------------------|----------------------------|
| Interpretación teórica: Él irá al hospital. | | |
| Palabra | Valores de Influencia (LIME) | |
| | Modelo GPT-3.5-Turbo | Modelo <i>fine-tuneado</i> |
| futuro | -0.04 | 0.09 |
| hospital | 0.34 | 0.51 |
| él | 0.20 | 0.16 |
| ir | 0.11 | 0.02 |

Tabla 6: Contribución de cada palabra según los Valores de Influencia (LIME) en las interpretaciones generadas por el modelo GPT-3.5-Turbo estándar y su versión *fine-tuneada* para la frase “futuro hospital él ir”.

El Cuadro 7 muestra los resultados de los análisis LIME para la frase “pasado yo ir no”, combinando la información de las Figuras 12 y 13. En este caso, se destaca un aumento significativo en el impacto del término “pasado”, que subió de 0.08 en el modelo estándar a 0.24 en el modelo *fine-tuneado*. Además, la palabra “no” mostró un notable incremento en su relevancia, pasando de 0.12 a 0.44. Finalmente, la influencia de “ir” se redujo de 0.37 a 0.17, mientras que “yo” mostró una leve mejora, pasando de -0.03 a 0.06.

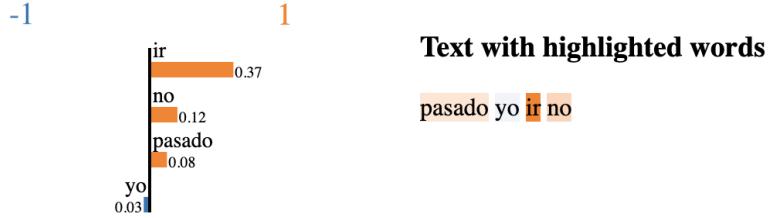


Figura 12: Resultados de LIME para la interpretación generada por el modelo GPT-3.5-Turbo estándar para la frase “pasado yo ir no”.

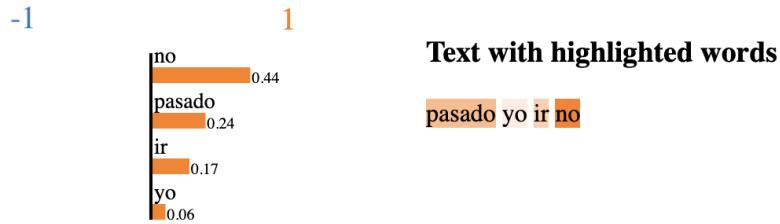


Figura 13: Resultados de LIME para la interpretación generada por el modelo *fine-tuneado* para la frase “pasado yo ir no”.

| Frase en LENSEGUA: pasado yo ir no | | |
|------------------------------------|------------------------------|----------------------------|
| Interpretación teórica: No fui. | | |
| Palabra | Valores de Influencia (LIME) | |
| | Modelo GPT-3.5-Turbo | Modelo <i>fine-tuneado</i> |
| pasado | 0.08 | 0.24 |
| yo | -0.03 | 0.06 |
| ir | 0.37 | 0.17 |
| no | 0.12 | 0.44 |

Tabla 7: Contribución de cada palabra según los Valores de Influencia (LIME) en las interpretaciones generadas por el modelo GPT-3.5-Turbo estándar y su versión *fine-tuneada* para la frase “pasado yo ir no”.

8.2. Prompt engineering

Durante las fases iniciales, se utilizó el siguiente *prompt* para guiar al modelo *fine-tuneado* en la generación de interpretaciones. Este *prompt* establecía las instrucciones necesarias para que el modelo actuara como un intérprete en LENSEGUA. Sin embargo, durante las pruebas realizadas, se identificaron varios aspectos que limitaron la calidad de las respuestas. En particular, se presentaron dificultades al procesar expresiones temporales, lo que resultó en errores y redundancias.

Prompt (versión 1)

Eres un intérprete experto en Lengua de Señas de Guatemala (LENSEGUA). El usuario te proporcionará frases en español escritas utilizando la gramática de LENSEGUA. Tu tarea es interpretarlas y convertirlas en español gramaticalmente correcto.

Con el objetivo de mejorar la calidad de las interpretaciones, se llevó a cabo una fase de *prompt engineering*. En esta etapa, se realizaron ajustes iterativos sobre el *prompt* original, incorporando instrucciones más precisas para corregir errores recurrentes. Además, para evaluar la efectividad de cada variante, como se mencionó en la sección de metodología, se calculó la distancia de Levenshtein promedio utilizando el conjunto de validación, el modelo *fine-tuneado* y las diferentes versiones de *prompts*. En este contexto, se propuso un segundo *prompt*.

Prompt (versión 2)

Soy un intérprete experto en Lengua de Señas de Guatemala (LENSEGUA). El usuario me proporcionará frases en español escritas utilizando la gramática de LENSEGUA. Mi tarea es interpretarlas y convertirlas en español gramaticalmente correcto.

Para esta segunda versión, se decidió formular las instrucciones en primera persona en lugar de tercera. Esto se hizo para que el modelo asumiera un rol más activo en el proceso de interpretación. Sin embargo, a pesar de este cambio, la distancia de Levenshtein promedio obtenida con las interpretaciones generadas por el modelo *fine-tuneado* y este nuevo *prompt* fue de 5.84, ligeramente superior al valor de 5.815 obtenido con la primera versión. Además, tras una revisión de las interpretaciones, se identificó que los errores anteriormente mencionados aún persistían. Ante esto, se retomó el *prompt* original y se elaboró una tercera versión del *prompt* que especificaba claramente la exclusión de ciertos términos.

Prompt (versión 3)

Eres un intérprete experto en Lengua de Señas de Guatemala (LENSEGUA). El usuario te proporcionará frases en español escritas utilizando la gramática de LENSEGUA. Tu tarea es interpretarlas y convertirlas en español gramaticalmente correcto, teniendo en cuenta la siguiente regla específica:

- Las palabras “pasado”, “futuro” y “antes” indican la conjugación verbal. Debes conjugar correctamente los verbos en tu respuesta sin incluir las palabras “pasado”, “futuro” o “antes”.

Esta tercera versión mostró una mejora, con una distancia de Levenshtein promedio de 5.37. No obstante, se identificaron problemas en la interpretación de expresiones comunes en LENSEGUA que no tienen un equivalente directo en español. Aunque el modelo *fine-tuneado* había sido expuesto a expresiones como ‘carro mucho’ (interpretado correctamente como ‘tráfico’) y ‘bien mucho’ (como ‘muy bien’) durante su entrenamiento, las respuestas que generaba eran inconsistentes; como si en ocasiones el modelo “olvidara” estas interpretaciones o no las aplicara de manera uniforme. Para asegurar respuestas más coherentes, se realizaron ajustes adicionales en el *prompt*, añadiendo recordatorios específicos sobre cómo manejar dichas expresiones. Además, se incluyó una instrucción que prevenía al modelo de interpretar frases como preguntas, a menos que incluyeran términos como

‘pregunta’ o ‘si o no’. Estos cambios contribuyeron significativamente a mejorar la estabilidad y la precisión en las respuestas del modelo.

Prompt (versión 4)

Eres un intérprete experto en Lengua de Señas de Guatemala (LENSEGUA). El usuario te proporcionará frases en español escritas utilizando la gramática de LENSEGUA. Tu tarea es interpretarlas y convertirlas en español gramaticalmente correcto, teniendo en cuenta las siguientes reglas específicas:

- *No debes interpretar las frases como preguntas, ni agregar una entonación interrogativa, a menos que incluyan las palabras “pregunta” o “sí o no”.*
- *Las palabras “pasado”, “futuro” y “antes” indican la conjugación verbal. Debes conjugar correctamente los verbos en tu respuesta sin incluir las palabras “pasado”, “futuro” o “antes”.*
- *La frase “carro mucho” debe interpretarse como “tráfico”.*
- *La frase “bien mucho” debe interpretarse como “muy bien”.*

La cuarta versión del *prompt* alcanzó una distancia de Levenshtein promedio de 4.30, calculada a partir de las interpretaciones generadas por el modelo *fine-tuneado*. Al analizar detenidamente las interpretaciones individuales, se observó que la mayoría eran correctas. Sin embargo, en algunos casos, el modelo alteraba el orden de los elementos dentro de las oraciones, lo que, en comparación con lo esperado, afectaba negativamente la métrica de desempeño. Para intentar solventar esta situación, se incorporó una instrucción adicional que enfatizaba la necesidad de mantener una estructura y secuencia específica en las interpretaciones finales.

Prompt (versión 5)

Eres un intérprete experto en Lengua de Señas de Guatemala (LENSEGUA). El usuario te proporcionará frases en español escritas utilizando la gramática de LENSEGUA. Tu tarea es interpretarlas y convertirlas en español gramaticalmente correcto, teniendo en cuenta las siguientes reglas específicas:

- *No debes interpretar las frases como preguntas, ni agregar una entonación interrogativa, a menos que incluyan las palabras “pregunta” o “sí o no”.*
- *Las palabras “pasado”, “futuro” y “antes” indican la conjugación verbal. Debes conjugar correctamente los verbos en tu respuesta sin incluir las palabras “pasado”, “futuro” o “antes”.*
- *La frase “carro mucho” debe interpretarse como “tráfico”.*
- *La frase “bien mucho” debe interpretarse como “muy bien”.*
- *Mantén el orden de los elementos de la interpretación final lo más similar posible al texto original cuando sea gramaticalmente válido en español. Por ejemplo, si la frase en LENSEGUA es “Por favor cuando tu salir luz apagar”, la interpretación debe ser “Por favor cuando salgas apaga la luz” en lugar de “Por favor apaga la luz cuando salgas”.*

Esta última versión del *prompt* obtuvo una distancia de Levenshtein promedio de 3.38. Este resultado indica una notable mejora en la coherencia y precisión de las interpretaciones, demostrando la efectividad del proceso iterativo de *prompt engineering*.

Es importante señalar que también se calculó la distancia de Levenshtein promedio para cada versión del *prompt* utilizando el modelo GPT-3.5-Turbo estándar y el *dataset* de validación. Esto permitió comparar el impacto de las modificaciones en cada versión del *prompt* sobre ambos modelos. Los valores obtenidos se detallan en el Cuadro 8.

| Prompt | Distancia de Levenshtein Promedio | |
|-----------|-----------------------------------|----------------------------|
| | Modelo GPT-3.5-Turbo | Modelo <i>fine-tuneado</i> |
| Versión 1 | 10.065 | 5.815 |
| Versión 2 | 10.755 | 5.840 |
| Versión 3 | 8.660 | 5.370 |
| Versión 4 | 8.635 | 4.300 |
| Versión 5 | 8.680 | 3.375 |

Tabla 8: Distancias de Levenshtein promedio calculadas a partir de las interpretaciones generadas por el modelo GPT-3.5-Turbo (estándar) y el modelo *fine-tuneado*, utilizando diversos *prompts*.

Los resultados indican que el modelo *fine-tuneado* superó consistentemente al modelo estándar, logrando menores distancias promedio para todos los *prompts* evaluados. Por ejemplo, con la versión 1 del *prompt*, el modelo estándar alcanzó una distancia de 10.065, mientras que el modelo *fine-tuneado* logró una distancia significativamente menor de 5.815. Asimismo, la menor distancia del modelo *fine-tuneado* se registró con la versión 5 del *prompt* (3.375), mientras que el mejor desempeño del modelo original fue con la versión 3, con una distancia de 8.660.

Aunque el *prompt* se fue mejorando de forma iterativa, el impacto de estos cambios se pudo observar únicamente en las interpretaciones generadas por el modelo *fine-tuneado*. Esto debido a que con cada versión nueva del *prompt*, se logró obtener una distancia de Levenshtein promedio menor. En cambio, las interpretaciones del modelo estándar no mostraron mejoras significativas, manteniendo distancias relativamente altas a pesar de los cambios. Esto sugiere que, si bien los *prompts* se optimizaron progresivamente, el modelo estándar no logró beneficiarse de estas actualizaciones. Esto posiblemente debido a que la tarea era demasiado compleja para resolverse únicamente con *prompt engineering*.

8.3. Retroalimentación de la comunidad sorda

En la última fase del proyecto, se recopiló retroalimentación sobre el desempeño del modelo *fine-tuneado*. Esta evaluación se llevó a cabo durante una reunión presencial en las oficinas de En-Señas, tal como se planteó en la metodología. Los participantes interactuaron con el sistema a través de la aplicación web local desarrollada, que se muestra en la Figura 14. Ellos destacaron la utilidad de la herramienta, especialmente porque la comunidad sorda utiliza frecuentemente ChatGPT para mejorar la redacción de sus comentarios escritos. Sin embargo, señalaron una limitación importante: el modelo estándar a menudo genera respuestas imprecisas, ya que no conoce la estructura gramatical y los matices lingüísticos de LENSEGUA. En contraste, los usuarios afirmaron que la versión *fine-tuneada* demostró una mejor comprensión de las características propias de LENSEGUA, lo cual se vio reflejado en las interpretaciones generadas.

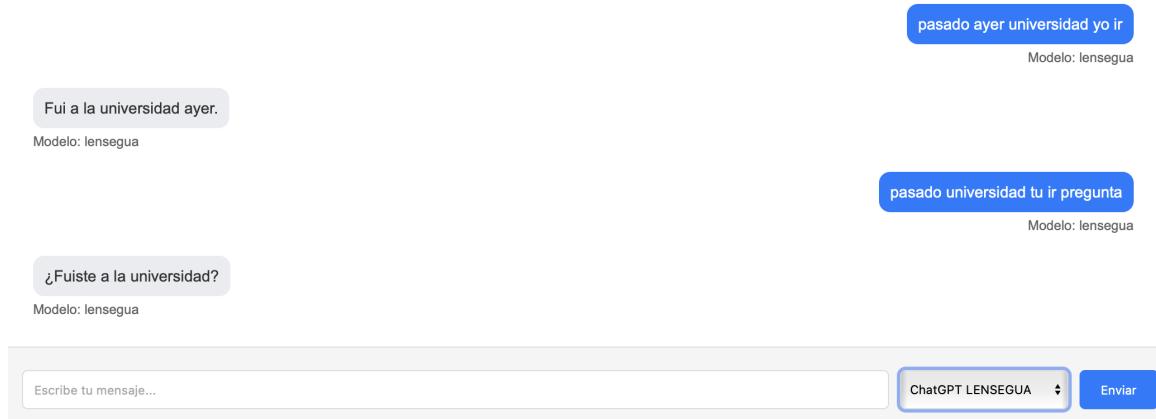


Figura 14: Aplicación web desarrollada para facilitar interacción con modelo *fine-tuneado*.

Para complementar la evaluación, se llevó a cabo una encuesta en Google Forms, que fue respondida no solo por los participantes de la reunión en En-Señas, sino también por otras personas sordas, intérpretes, estudiantes y usuarios familiarizados con LENSEGUA. En esta encuesta, un total de 19 personas evaluaron las interpretaciones generadas por ambos modelos en una escala del 1 al 5, donde 1 correspondía a menor coherencia y 5 a mayor coherencia o claridad. Cabe resaltar que dichas interpretaciones se generaron utilizando la quinta versión del *prompt* desarrollado.

En este contexto, el Cuadro 9 presenta un ejemplo comparativo de las interpretaciones de la frase “ayer cine tu ir pregunta”. El modelo original generó la interpretación “Ayer fuiste al cine, ¿verdad?”, y obtuvo una puntuación promedio de 3.4. En cambio, el modelo *fine-tuneado* interpretó la frase como “¿Fuiste al cine ayer?”, alcanzando una puntuación de 4.8. Los 19 participantes estuvieron de acuerdo en que la interpretación del modelo *fine-tuneado* fue más precisa y adecuada.

| Frase en LENSEGUA: ayer cine tu ir pregunta | | |
|---|-------------------------------|---------------------|
| Modelo | Interpretación | Puntuación Promedio |
| GPT-3.5-Turbo | Ayer fuiste al cine, ¿verdad? | 3.4 |
| <i>Fine-Tuneado</i> | ¿Fuiste al cine ayer? | 4.8 |

Tabla 9: Puntuación promedio otorgada por usuarios finales a las interpretaciones generadas por el modelo GPT-3.5-Turbo estándar y su versión *fine-tuneada* para la frase “ayer cine tu ir pregunta”.

En el Cuadro 10 se analiza la frase “pasado yo medicina comprar para mamá”. El modelo original generó la interpretación “En el pasado compré medicina para mi mamá”, obteniendo una puntuación promedio de 3.8. Por su parte, el modelo *fine-tuneado* simplificó la interpretación a “Compré medicina para mi mamá”, logrando una puntuación de 4.9. Si bien ambas interpretaciones son correctas, todos los participantes coincidieron en que la del modelo *fine-tuneado* fue más apropiada. Esto probablemente debido a que el modelo original añadió información -“en el pasado”- que suele omitirse.

| Frase en LENSEGUa: | | |
|--------------------|--|---------------------|
| Modelo | Interpretación | Puntuación Promedio |
| GPT-3.5-Turbo | En el pasado compré medicina para mi mamá. | 3.8 |
| Fine-Tuneado | Compré medicina para mi mamá. | 4.9 |

Tabla 10: Puntuación promedio otorgada por usuarios finales a las interpretaciones generadas por el modelo GPT-3.5-Turbo estándar y su versión *fine-tuneada* para la frase “pasado yo medicina comprar para mamá”.

El Cuadro 11 presenta los resultados para la frase “ayer abuelo llamar tu pregunta”. El modelo original generó “Ayer tu abuelo te llamó para preguntarte algo”, y obtuvo una puntuación de 1.8. En contraste, el modelo *fine-tuneado* generó la frase “¿Ayer te llamó tu abuelo?”, obteniendo 4.9 puntos. Los participantes seleccionaron esta última interpretación como la más clara y correcta, posiblemente ya que capturaba de manera más fiel la idea que se buscaba transmitir.

| Frase en LENSEGUa: ayer abuelo llamar tu pregunta | | |
|---|--|---------------------|
| Modelo | Interpretación | Puntuación Promedio |
| GPT-3.5-Turbo | Ayer tu abuelo te llamó para preguntarte algo. | 1.8 |
| Fine-Tuneado | ¿Ayer te llamó tu abuelo? | 4.9 |

Tabla 11: Puntuación promedio otorgada por usuarios finales a las interpretaciones generadas por el modelo GPT-3.5-Turbo estándar y su versión *fine-tuneada* para la frase “ayer abuelo llamar tu pregunta”.

En el Cuadro 12 se muestra la evaluación de la frase “antes tienda tu amiga ver pregunta”. El modelo original generó “Tu amiga vio la tienda y te preguntó algo”, y obtuvo una puntuación de 1.4. El modelo *fine-tuneado* produjo “¿Viste a tu amiga en la tienda?”, alcanzando 4.1 puntos. Nuevamente, todos los participantes coincidieron en que la interpretación generada por el modelo adaptado era la más clara y coherente.

| Frase en LENSEGUa: antes tienda tu amiga ver pregunta | | |
|---|--|---------------------|
| Modelo | Interpretación | Puntuación Promedio |
| GPT-3.5-Turbo | Tu amiga vio la tienda y te preguntó algo. | 1.4 |
| Fine-Tuneado | ¿Viste a tu amiga en la tienda? | 4.1 |

Tabla 12: Puntuación promedio otorgada por usuarios finales a las interpretaciones generadas por el modelo GPT-3.5-Turbo estándar y su versión *fine-tuneada* para la frase “antes tienda tu amiga ver pregunta”.

Finalmente, en el Cuadro 13 se analiza la frase “tu opinión cuál pregunta”. El modelo original la interpretó como “Tu opinión sobre cuál es la pregunta”, obteniendo 1.4 puntos. Por otro lado, el modelo *fine-tuneado* generó “¿Cuál es tu opinión?”, logrando la máxima puntuación de 5.0 puntos. Todos los participantes estuvieron de acuerdo en que esta última interpretación fue la más adecuada.

| Frase en LENSEGUA: tu opinión cuál pregunta | | |
|---|---------------------------------------|---------------------|
| Modelo | Interpretación | Puntuación Promedio |
| GPT-3.5-Turbo | Tu opinión sobre cuál es la pregunta. | 1.4 |
| Fine-Tuneado | ¿Cuál es tu opinión? | 5.0 |

Tabla 13: Puntuación promedio otorgada por usuarios finales a las interpretaciones generadas por el modelo GPT-3.5-Turbo estándar y su versión *fine-tuneada* para la frase “tu opinión cuál pregunta”.

En la última fase de la encuesta, se solicitó a los participantes evaluar cinco interpretaciones de una misma frase. Cada una de las interpretaciones fue generada utilizando un *prompt* distinto, en conjunto con el modelo *fine-tuneado*. El objetivo fue identificar cuál de los cinco *prompts* producía la interpretación más adecuada y coherente según personas conocedoras de la gramática de LENSEGUA.

Como se puede observar en el Cuadro 14, las primeras tres interpretaciones obtuvieron calificaciones promedio bajas (1.4, 1.1 y 1.4, respectivamente), ya que no lograron transmitir correctamente la idea original y contenían errores. En lugar de indicar que se esperaba que no hubiera tráfico, estas interpretaciones mencionaron lo opuesto, lo que distorsionó el mensaje. Por otro lado, la cuarta y quinta versión generaron interpretaciones mucho más adecuadas y claras, logrando transmitir el mensaje de manera efectiva. Es importante mencionar que estas últimas dos interpretaciones eran muy similares; sin embargo, debido a ligeras variaciones en la redacción, los usuarios indicaron que la interpretación generada por el quinto *prompt* fue superior, obteniendo una puntuación promedio de 5.0.

Finalmente, de los 19 participantes, 18 eligieron la interpretación generada con el quinto *prompt* como su favorita, mientras que solo uno optó por la interpretación generada con el cuarto *prompt*. Este resultado no solo sugiere que el quinto *prompt* fue el más efectivo, sino que también coincide con las puntuaciones de distancia de Levenshtein promedio mencionadas anteriormente.

| Frase en LENSEGUA: hospital yo necesitar ir ahora ojalá mucho carro no porque emergencia | | |
|--|--|---------------------|
| Prompt | Interpretación | Puntuación Promedio |
| Versión 1 | Necesito ir al hospital ahora espero que haya muchos carros porque es una emergencia. | 1.4 |
| Versión 2 | Necesito ir al hospital ahora espero que haya muchos carros disponibles no tengo carro porque es una emergencia. | 1.1 |
| Versión 3 | Necesito ir al hospital ahora porque es una emergencia y ojalá haya mucho carro. | 1.4 |
| Versión 4 | Necesito ir al hospital ahora espero que no haya mucho tráfico porque es una emergencia. | 4.6 |
| Versión 5 | Necesito ir al hospital ahora mismo. Espero que no haya mucho tráfico porque es una emergencia. | 5.0 |

Tabla 14: Puntuación promedio otorgada por usuarios finales a las interpretaciones generadas por el modelo GPT-3.5-Turbo *fine-tuneado*, utilizando diferentes *prompts*, para la frase “hospital yo necesitar ir ahora ojalá mucho carro no porque emergencia”.

CAPÍTULO 9

Análisis de resultados

El proceso de *fine-tuning* funcionó por dos razones principales. En primer lugar, se contó con el apoyo de intérpretes de LENSEGUA para la generación del *dataset*, lo que garantizó que los datos reflejaran fielmente la gramática y las estructuras de LENSEGUA. La calidad y autenticidad de estos datos fue esencial para que el modelo capturara los matices del lenguaje, evitando interpretaciones inconsistentes o erróneas. Además, la cantidad de datos recopilados y la variación dentro del *dataset* fue fundamental. Sin una muestra representativa y suficientemente diversa, el modelo no habría podido desarrollar la capacidad de interpretar tanto frases simples como complejas, ni hubiera aprendido a generalizar de forma efectiva en escenarios no vistos durante el entrenamiento.

La optimización de hiperparámetros también fue importante para el éxito del *fine-tuning*. A través de las diversas pruebas, se identificó que un entrenamiento con 2 *epochs* proporcionaba el balance óptimo entre aprendizaje y generalización, evitando el sobreajuste que se observó con *epochs* adicionales. Por ejemplo, con 3 *epochs* se observó que el modelo comenzaba a memorizar patrones específicos en lugar de aprender reglas generales de la lengua. De manera similar, el *learning rate multiplier* de 0.2 demostró ser ideal para permitir ajustes graduales en los pesos del modelo, garantizando un aprendizaje estable y progresivo. Este valor contrastó significativamente con pruebas realizadas con valores más altos (como 2.0), los cuales generaban actualizaciones demasiado bruscas que impedían una convergencia adecuada del modelo.

La efectividad del *fine-tuning* se pudo validar mediante el análisis LIME, el cual reveló cambios significativos en la forma en que el modelo procesaba elementos lingüísticos clave de LENSEGUA. Más específicamente, este análisis demostró que el modelo *fine-tuneado* logró aprender a interpretar palabras contextuales específicas que, aunque no deben aparecer explícitamente en las interpretaciones finales, son esenciales para la comprensión del mensaje en LENSEGUA, como “pregunta”, “futuro” y “pasado”. Estas palabras cumplen un rol único, porque representan conceptos que en el español estándar se comunican mediante el uso de signos de interrogación o conjugaciones verbales.

El análisis mostró que el modelo estándar asignaba puntuaciones de importancia negativas o muy bajas a estas palabras. Por ejemplo, “pregunta” recibió un valor de -0.09, “futuro” -0.04 y “pasado” 0.08. Estos resultados sugieren que el modelo percibía estas palabras como elementos disruptivos para la interpretación. Esta percepción errónea puede atribuirse a que el modelo GPT-3.5 original no fue entrenado específicamente para trabajar con las estructuras lingüísticas de LENSEGUA.

Por lo tanto, el modelo estándar simplemente intentaba acoplar todas las palabras disponibles para generar una oración coherente en español, añadiendo conectores, preposiciones, artículos y otros elementos típicos de la gramática estándar. Sin embargo, al operar de esta manera, el modelo no lograba capturar el significado implícito que estas palabras tienen en LENSEGUA.

En contraste, el modelo adaptado mediante *fine-tuning* demostró un cambio significativo en la forma de interpretar estas palabras, asignando valores positivos a “pregunta” (0.09), “futuro” (0.09) y “pasado” (0.24). Esto evidencia que el modelo aprendió a reconocer la relevancia de estos términos en la construcción de interpretaciones coherentes en el contexto de LENSEGUA, lo cual fue fundamental para garantizar interpretaciones alineadas con el significado original.

El refinamiento iterativo de los *prompts* también fue fundamental para optimizar el desempeño del modelo *fine-tuneado*. Esto debido a que, en primer lugar, se desarrollaron instrucciones más específicas y detalladas que explicaban claramente al modelo cómo debía procesar las estructuras gramaticales de LENSEGUA. Se establecieron formatos precisos para las respuestas esperadas, definiendo exactamente cómo debían estructurarse las interpretaciones y qué elementos debían incluirse o excluirse. Por ejemplo, se especificó que ciertos marcadores gramaticales, como “pregunta”, no debían aparecer en la interpretación final, sino que debían usarse como guías para ajustar la estructura de la oración resultante.

La efectividad de estas mejoras en los *prompts* se comprobó mediante el análisis de las distancias de Levenshtein, una métrica que mide cuántas ediciones se necesitan para convertir una interpretación generada por el modelo en una interpretación de referencia establecida. En este caso, un valor menor indica mayor similitud y, por lo tanto, mejor rendimiento del modelo. Con la primera versión del *prompt*, el modelo *fine-tuneado* ya mostraba un rendimiento superior al modelo estándar, con una distancia promedio de 5.815 frente a 10.065, lo que representa una mejora inicial del 42.23 %. Esta diferencia significativa en el rendimiento base confirma que el proceso de *fine-tuning* estableció una base sólida para la comprensión de las estructuras lingüísticas de LENSEGUA.

Las iteraciones sucesivas en el diseño de los *prompts* permitieron mejorar progresivamente el desempeño del modelo *fine-tuneado*. En realidad, la versión final del *prompt* logró reducir la distancia de Levenshtein promedio del modelo *fine-tuneado* a 3.375, lo que representa una mejora del 41.96 % respecto a la primera versión. Esta reducción significativa demuestra que la especificación más precisa de las instrucciones permitió al modelo generar interpretaciones más cercanas a las referencias definidas. Cada iteración en el diseño de los *prompts* aportó mayor claridad sobre cómo manejar casos específicos de la gramática de LENSEGUA, lo que se reflejó directamente en la calidad de las interpretaciones.

Por otro lado, los resultados del modelo estándar no siguieron esta tendencia de mejora. De hecho, las métricas de este modelo se mantuvieron consistentemente elevadas con todos los *prompts*. Entre la primera y la tercera versión del *prompt*, la distancia promedio calculada solo disminuyó un 13.96 %. Más aún, de la tercera versión a la última, se observó un ligero aumento del 0.23 % en la distancia de Levenshtein promedio. Estos resultados indican que incluso las instrucciones más detalladas y estructuradas no pudieron compensar la falta de un entendimiento básico de las estructuras lingüísticas de LENSEGUA que proporciona el *fine-tuning*. El *prompt engineering* funciona mejor cuando el modelo ya tiene una comprensión fundamental del lenguaje, permitiendo que las instrucciones refinadas guíen y optimicen este conocimiento base.

Los resultados de las encuestas realizadas con la comunidad sorda reafirmaron esta diferencia en el rendimiento. Los participantes mostraron una clara preferencia por las interpretaciones del modelo *fine-tuneado*, destacando su coherencia y claridad. Este modelo logró captar matices específicas de la lengua, generando interpretaciones más acordes con las expectativas de los usuarios. En contraste, el modelo estándar se limitó a agregar conectores y preposiciones a las frases originales, produciendo oraciones gramaticalmente correctas pero que a menudo no transmitían adecuadamente la idea principal.

Por ejemplo, la frase en LENSEGUA “ayer abuelo llamar tu pregunta” fue interpretada de manera muy diferente por ambos modelos. El modelo estándar de GPT-3.5-Turbo generó la interpretación “Ayer tu abuelo te llamó para preguntarte algo”, que recibió una puntuación promedio de 1.8 por parte de los usuarios. Aunque es gramaticalmente correcta, esta respuesta realmente no captó la esencia de la frase original. Por otro lado, el modelo *fine-tuneado* interpretó la misma frase como “¿Ayer te llamó tu abuelo?”, obteniendo una puntuación promedio de 4.9. Esta interpretación no solo era más concisa, sino que también mantuvo la idea que se buscaba trasmitir.

El Cuadro 14 muestra evidencia adicional del impacto positivo del refinamiento de los *prompts* en la efectividad del modelo *fine-tuneado*. A medida que se ajustaron los *prompts*, los participantes de la encuesta evaluaron las interpretaciones generadas para la frase “hospital yo necesitar ir ahora ojalá mucho carro no porque emergencia”. La primera versión del *prompt* produjo una interpretación confusa, “Necesito ir al hospital ahora espero que haya muchos carros porque es una emergencia”, recibiendo una puntuación promedio de 1.4. En lugar de expresar que se esperaba que no hubiera tráfico, la interpretación resultante indicó lo opuesto, lo que distorsionó completamente el mensaje. Sin embargo, la versión final logró una interpretación precisa y contextualmente adecuada, alcanzando una puntuación máxima de 5.0.

La interpretación final, “Necesito ir al hospital ahora mismo. Espero que no haya mucho tráfico porque es una emergencia”, no solo reflejó correctamente el significado original, sino que también trasmittió la urgencia del mensaje. Esta mejora sustancial se debe a que los últimos *prompts* eran más completos e introdujeron aclaraciones específicas para abordar problemas detectados en las interpretaciones anteriores. Nuevamente, se incluyeron instrucciones detalladas sobre cómo manejar términos temporales, cómo interpretar correctamente las palabras clave, y cómo estructurar las interpretaciones finales de manera que transmitieran el tono y el contexto adecuados.

En conclusión, este análisis demuestra que la combinación de *fine-tuning* y *prompt engineering* fue fundamental para enseñarle al modelo GPT-3.5-Turbo a interpretar este tipo de frases. Los resultados obtenidos confirman que la tarea mencionada era demasiado compleja para ser resuelta únicamente mediante *prompt engineering*. Nuevamente, el modelo estándar, incluso con *prompts* refinados, no logró generar interpretaciones satisfactorias consistentemente. Sin embargo, es importante destacar que el *fine-tuning* por sí solo, sin un *prompt* óptimo, aunque genera buenos resultados, tampoco alcanza el nivel máximo de rendimiento.

CAPÍTULO 10

Conclusiones

- El objetivo general de adaptar un LLM para asimilar la gramática de LENSEGUA se logró con éxito. El modelo *fine-tuneado* demostró una capacidad significativamente mejorada para analizar oraciones en LENSEGUA e interpretarlas correctamente al español, superando las limitaciones del modelo base y ofreciendo una herramienta valiosa para la comunidad sorda guatemalteca.
- Ante la ausencia de un conjunto de datos preexistente que reflejara la gramática de LENSEGUA, se decidió involucrar a intérpretes y personas sordas en la generación de frases representativas. Este enfoque, alineado con los principios de *responsible AI*, permitió recopilar una variedad de frases que capturaban con precisión las características lingüísticas de LENSEGUA. Sin este apoyo, se considera que el *fine-tuning* del sistema no hubiera sido tan efectivo, lo que habría limitado su capacidad para generar interpretaciones precisas en la práctica.
- Aplicar la técnica de *fine-tuning* en el modelo GPT-3.5-Turbo utilizando el conjunto de datos desarrollado demostró ser una estrategia efectiva. Mientras que las interpretaciones del modelo estándar presentaron una distancia de Levenshtein promedio de 10.065, las del modelo *fine-tuneado* lograron obtener una distancia promedio de 5.815, lo que refleja una mejora del 42.23 %. Este proceso de adaptación permitió al modelo asimilar las características gramaticales y semánticas de LENSEGUA, resultando en una notable mejora en la calidad de sus respuestas.
- La implementación de *prompt engineering* facilitó la creación de *prompts* más sofisticados, lo que llevó a interpretaciones más precisas por parte del modelo *fine-tuneado*. Sin embargo, estos avances tuvieron un impacto limitado en las interpretaciones generadas por el modelo estándar. Por lo tanto, si bien el *prompt engineering* maximizó el rendimiento del modelo adaptado, la tarea de interpretación no pudo haberse resuelto únicamente mediante esta técnica debido a la complejidad de la misma.
- Los resultados de la encuesta revelaron que el modelo *fine-tuneado* superó al modelo estándar en la tarea de generación de interpretaciones. Los participantes otorgaron una calificación promedio de 4.74 (en una escala de 1 a 5) a las interpretaciones generadas por el modelo *fine-tuneado*, mientras que las del modelo estándar solo alcanzaron 2.36. Esta diferencia no solo resalta la efectividad de la metodología aplicada, sino que también refleja la satisfacción de la comunidad sorda con el producto final.

CAPÍTULO 11

Recomendaciones

- Se recomienda buscar el apoyo de diversas asociaciones que trabajen con la comunidad sorda en Guatemala, además de ASEDES, para la elaboración de más frases representativas. Dado que existen variaciones regionales de LENSEGUÁ, la inclusión de múltiples perspectivas enriquecería el conjunto de datos y garantizaría una representación más completa de la gramática y vocabulario utilizado.
- Se recomienda ampliar el conjunto de datos para incluir frases más complejas y largas, así como párrafos completos. Esto permitirá que el modelo *fine-tuneado* esté expuesto a una mayor diversidad de estructuras lingüísticas, asegurando que sea capaz de interpretar frases más elaboradas.
- Se recomienda evaluar el uso de modelos más actualizados, como GPT-4o, para realizar el *fine-tuning*. Este modelo podría ofrecer mejoras significativas en la calidad de las interpretaciones debido a su entrenamiento con un *corpus* más amplio, así como optimizaciones en su arquitectura que le permiten gestionar de manera más eficiente tareas complejas.
- Se recomienda continuar explorando técnicas de *prompt engineering* para mejorar aún más las interpretaciones del modelo. Si bien esta técnica ya permitió desarrollar *prompts* más avanzados que incrementaron la precisión del modelo *fine-tuneado*, es posible que ajustes adicionales optimicen aún más los resultados. Además, se sugiere probar el uso exclusivo de *prompt engineering* con modelos más recientes, como GPT-4o, para evaluar si esta combinación permite generar interpretaciones correctas sin necesidad de un *fine-tuning* específico.
- Se recomienda crear una plataforma accesible en línea para que la comunidad sorda pueda interactuar con el modelo. Como el desarrollo de esta plataforma no estuvo dentro del alcance del proyecto, solo se implementó una página web local para realizar pruebas y demostrar el funcionamiento del sistema. No obstante, la retroalimentación obtenida indicó que la comunidad sorda considera importante tener acceso a esta herramienta.
- Se recomienda implementar prácticas de MLOps que permitan evaluar periódicamente el rendimiento del modelo *fine-tuneado*. Esto incluiría procesos automatizados para monitorear la calidad de las predicciones generadas y, en caso de que el rendimiento disminuya, activar un *pipeline* de re-entrenamiento o ajuste del modelo para mantener su efectividad a lo largo del tiempo.

Bibliografía

- [1] Amazon Web Services: *¿Qué son los modelos de lenguaje de gran tamaño? - Explicación sobre los LLM de IA - AWS*, 2023. <https://aws.amazon.com/es/what-is/large-language-model/>.
- [2] Amazon Web Services: *What is GPT AI? - Generative Pre-Trained Transformers Explained - AWS*, 2024. <https://aws.amazon.com/what-is/gpt/>.
- [3] Amazon Web Services: *What is Natural Language Processing? - NLP - AWS*, 2024. <https://aws.amazon.com/what-is/nlp/>.
- [4] Better Health Australia: *Hearing loss - how it affects people*, 2024. <https://www.betterhealth.vic.gov.au/health/conditionsandtreatments/hearing-loss-how-it-affects-people#:~:text=Hearing%20loss%20can%20affect%20a>.
- [5] Brentwood Hearing Center: *How Does Hearing Loss Affect Communication?*, 2020. <https://brentwoodhearingcenter.com/how-does-hearing-loss-affect-communication/>.
- [6] Congreso de Guatemala: *LENSEGUA, ley que fomenta la inclusión social*, 2022. https://www.congreso.gob.gt/noticias_congreso/9131/2022/4#gsc.tab=0.
- [7] Cutten, D.: *How can hearing impairment affect language development?*, 2018. <https://www.hiddenhearing.co.uk/hearing-blog/hearing-aids/how-can-a-hearing-impairment-affect-language-development>.
- [8] Data Kwery: *The Fundamentals of Natural Language Processing: A Beginner's Guide - DataKwery*, 2023. <https://www.datakwery.com/post/fundamentals-of-nlp-guide/>.
- [9] DeepLearningAI: *A Basic Introduction to Logistic Regression for Machine Learning*, 2022. https://www.deeplearning.ai/the-batch/logistic-regression-follow-the-curve/?_gl=1.
- [10] DeepLearningAI: *Natural Language Processing (NLP) - A Complete Guide*, 2023. <https://www.deeplearning.ai/resources/natural-language-processing/>.
- [11] Dieber, Jürgen y Sabrina Kirrane: *Why model why? Assessing the strengths and limitations of LIME*. arXiv preprint arXiv:2012.00093, 2020.
- [12] Doshi-Velez, Finale y Been Kim: *Towards a rigorous science of interpretable machine learning*. arXiv preprint arXiv:1702.08608, 2017.
- [13] En-Señas: *Características de LENSEGUA*. En-Señas, 2022.

- [14] Fan, Yixin, Feng Jiang, Peifeng Li y Haizhou Li: *Grammargpt: Exploring open-source llms for native chinese grammatical error correction with supervised fine-tuning*. En *CCF International Conference on Natural Language Processing and Chinese Computing*, páginas 69–80. Springer, 2023.
- [15] Ferrer, J.: *How Transformers Work: A Detailed Exploration of Transformer Architecture*, 2024. <https://www.datacamp.com/tutorial/how-transformers-work>.
- [16] Gupta, Y.: *Chat GPT and GPT 3 Detailed Architecture Study-Deep NLP Horse*, 03 2023. <https://medium.com/nerd-for-tech/gpt3-and-chat-gpt-detailed-architecture-study-deep-nlp-horse-db3af9de8a5d>.
- [17] Hardesty, L.: *Explained: Neural networks*, 2017. <https://news.mit.edu/2017/explained-neural-networks-deep-learning-0414>.
- [18] Jeyasheeli, P Golda Indumathi, N: *Sentence Generation for Indian Sign Language Using NLP*. Webology, 18(SI01):196–210, 2021.
- [19] Karthik, A. Ilya, T.: *Improving Language Understanding by Generative Pre-Training*, 2018. <https://www.mikecaptain.com/resources/pdf/GPT-1.pdf>.
- [20] Koehrsen, W.: *Neural Network Embeddings Explained*, 2018. <https://towardsdatascience.com/neural-network-embeddings-explained-4d028e6f0526?gi=5bc1084d4154>.
- [21] La Asociación Americana del Habla, Lenguaje y Audición: *Tipo, grado y configuración de la pérdida de audición SERIE INFORMATIVA DE AUDIOLOGÍA*, 2020. <https://www.asha.org/siteassets/ais/ais-type-degree-and-configuration-of-hearing-loss-spanish.pdf>.
- [22] Martínez, B.: *Aprendamos lengua de señas para una verdadera inclusión*, 2021. <https://www.prensalibre.com/guatemala/comunitario/aprendamos-lengua-de-señas-para-una-verdadera-inclusión/>.
- [23] McKinsey Company: *WHat is generative AI?*, 2024. <https://www.mckinsey.com/featured-insights/mckinsey-explainers/what-is-generative-ai>.
- [24] National Deaf Children’s Society: *Deafness causes before birth / Deafness in childhood*, 2023. <https://www.ndcs.org.uk/information-and-support/childhood-deafness/causes-of-deafness/>.
- [25] National Deaf Children’s Society: *What is sign language? / Communication*, 2023. <https://www.ndcs.org.uk/information-and-support/language-and-communication/sign-language/what-is-sign-language/>.
- [26] OpenAI: *Fine-tuning*, 2024. <https://platform.openai.com/docs/guides/fine-tuning>.
- [27] Parks, E. Parks, J.: *Encuesta sociolingüística de la comunidad sorda en Guatemala*. . 2015. https://www.academia.edu/download/60283266/Encuesta_sociolinguistica_de_la_comunidad_sorda_en_Guatemala20190813-53108-c6xgg6.pdf.
- [28] Ribeiro, Marco Tulio, Sameer Singh y Carlos Guestrin: *"Why should i trust you?.Explaining the predictions of any classifier*. En *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, páginas 1135–1144, 2016.
- [29] Rokon, O.: *RNN vs. LSTM vs. Transformers: Unraveling the Secrets of Sequential Data Processing*, 2023. <https://medium.com/@mroko001/rnn-vs-lstm-vs-transformers-unraveling-the-secrets-of-sequential-data-processing-c4541c4b09f>
- [30] Scikit-learn: *1.9. Naive Bayes — scikit-learn 0.21.3 documentation*, 2019. https://scikit-learn.org/stable/modules/naive_bayes.html.

- [31] Serva, Maurizio y Filippo Petroni: *Indo-European languages tree by Levenshtein distance*. Euphysics letters, 81(6):68005, 2008.
- [32] Sherstinsky, A.: *Fundamentals of Recurrent Neural Network (RNN) and Long Short-Term Memory (LSTM) Network*, 2018. <https://arxiv.org/abs/1808.03314>.
- [33] Sinha, S.: *From GPT-1 to GPT-4: A Look at the Evolution of Generative AI*, 2023. <https://hgs.cx/blog/from-gpt-1-to-gpt-4-a-look-at-the-evolution-of-generative-ai/>.
- [34] Taneja, A.: *Deep Dive into the Positional Encodings of the Transformer Neural Network Architecture: With Code!*, 2023. <https://www.linkedin.com/pulse/deep-dive-positional-encodings-transformer-neural-network-ajay-taneja#:~:text=Positional%20Encodings%20can%20be%20looked>.
- [35] UNICEF: *II Encuesta Nacional de Discapacidad en Guatemala*, 2016. <https://www.unicef.org/guatemala/informes/ii-encuesta-nacional-de-discapacidad-en-guatemala>.
- [36] Vaswani, A., Vaswani, B., Shazeer, N., Niki J., Uszkoreit, J.: *Attention Is All You Need*, 2017. <https://arxiv.org/pdf/1706.03762.pdf>.
- [37] Vu, Minh N, Truc D Nguyen, NhatHai Phan, Ralucca Gera y My T Thai: *Evaluating explainers via perturbation*. 2019.
- [38] Wazalwar, Sampada S Shrawankar, Urmila: *Interpretation of sign language into English using NLP techniques*. Journal of Information and Optimization Sciences, 38(6):895–910, 2017.
- [39] Weiss, A.: *Prompt engineering: definición, ejemplos y buenas prácticas*, 2023. <https://www.ionos.mx/digitalguide/paginas-web/creacion-de-paginas-web/prompt-engineering/>.
- [40] Woll, B.: *The History of Sign Language Linguistics*. Oxford, 2013. <https://doi.org/10.1093/oxfordhb/9780199585847.013.0005>.
- [41] World Federation of the Deaf: *International Advocacy*, 2023. <https://wfdeaf.org/our-work/international-advocacy/>.
- [42] World Health Organization: *Deafness and hearing loss*, 02 2023. <https://www.who.int/news-room/fact-sheets/detail/deafness-and-hearing-loss#:~:text=A%20person%20who%20is%20not>.

Anexos

Frase con gramática de LENSEGUA: “ayer cine tu ir pregunta”

Interpretación de ChatGPT: “¿Fuiste al cine ayer?”

¿Qué puntaje le daría del **1** al **5**, siendo 1 el más bajo y 5 el más alto, a la interpretación generada por ChatGPT en términos de claridad y coherencia?

19 respuestas

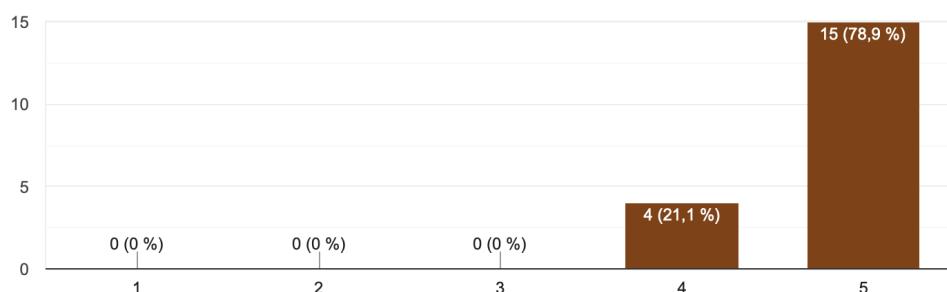


Figura 15: Distribución de calificaciones otorgadas por los usuarios a la interpretación de la frase No. 1 generada por el modelo *fine-tuneado* con la última versión del *prompt*.

Frase con gramática de LENSEGUA: "ayer cine tu ir pregunta"

Interpretación de ChatGPT: "Ayer fuiste al cine, ¿verdad?"

¿Qué puntaje le daría del 1 al 5, siendo 1 el más bajo y 5 el más alto, a la interpretación generada por ChatGPT en términos de claridad y coherencia?

19 respuestas

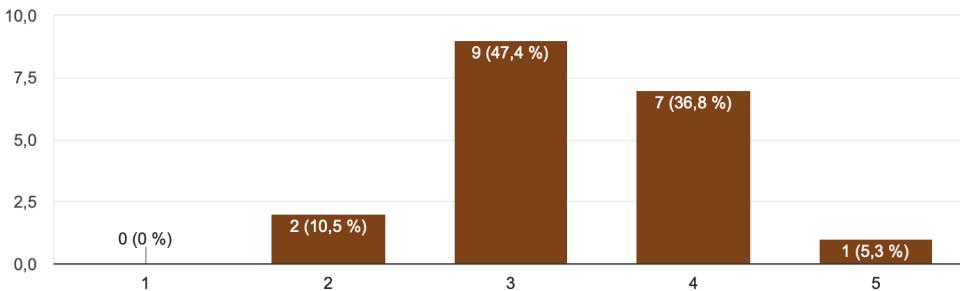


Figura 16: Distribución de calificaciones otorgadas por los usuarios a la interpretación de la frase No. 1 generada por el modelo estándar con la última versión del *prompt*.

Frase con gramática de LENSEGUA: "ayer cine tu ir pregunta"

Interpretación 1: "¿Fuiste al cine ayer?"

Interpretación 2: "Ayer fuiste al cine, ¿verdad?"

De las dos interpretaciones presentadas, ¿cuál le parece más clara o adecuada según su criterio?

19 respuestas

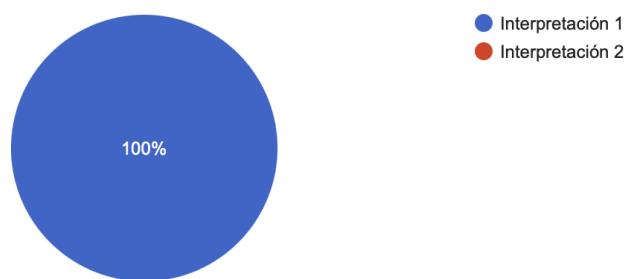


Figura 17: Preferencia de los usuarios entre dos interpretaciones de la frase No. 1: Interpretación 1 (modelo *fine-tuneado*) e Interpretación 2 (modelo estándar).

Frase con gramática de LENSEGUA: "pasado yo medicina comprar para mamá"

Interpretación de ChatGPT: "Compré medicina para mi mamá."

¿Qué puntaje le daría del **1** al **5**, siendo 1 el más bajo y 5 el más alto, a la interpretación generada por ChatGPT en términos de claridad y coherencia?

19 respuestas

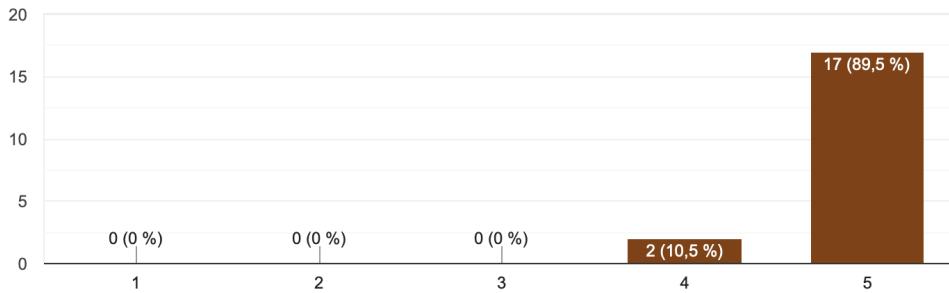


Figura 18: Distribución de calificaciones para la interpretación de la frase No. 2 generada por el modelo *fine-tuneado* con la última versión del *prompt*.

Frase con gramática de LENSEGUA: "pasado yo medicina comprar para mamá"

Interpretación de ChatGPT: "En el pasado compré medicina para mi mamá."

¿Qué puntaje le daría del **1** al **5**, siendo 1 el más bajo y 5 el más alto, a la interpretación generada por ChatGPT en términos de claridad y coherencia?

19 respuestas

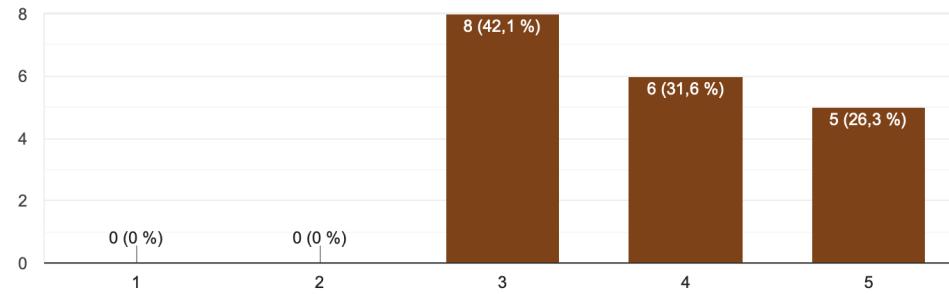


Figura 19: Distribución de calificaciones para la interpretación de la frase No. 2 generada por el modelo estándar con la última versión del *prompt*.

Frase con gramática de LENSEGUA: “pasado yo medicina comprar para mamá”

Interpretación 1: “Compré medicina para mi mamá.”

Interpretación 2: “En el pasado compré medicina para mi mamá.”

De las dos interpretaciones presentadas, ¿cuál le parece más clara o adecuada según su criterio?

19 respuestas



Figura 20: Preferencia de los usuarios entre dos interpretaciones de la frase No. 2: Interpretación 1 (modelo *fine-tuneado*) e Interpretación 2 (modelo estándar).

Frase con gramática de LENSEGUA: “ayer abuelo llamar tu pregunta”

Interpretación de ChatGPT: “¿Ayer te llamó tu abuelo?”

¿Qué puntaje le daría del 1 al 5, siendo 1 el más bajo y 5 el más alto, a la interpretación generada por ChatGPT en términos de claridad y coherencia?

19 respuestas

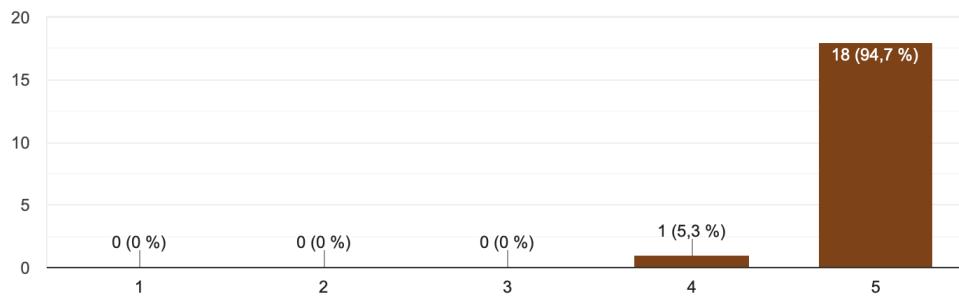


Figura 21: Distribución de calificaciones para la interpretación de la frase No. 3 generada por el modelo *fine-tuneado* con la última versión del *prompt*.

Frase con gramática de LENSEGUA: "ayer abuelo llamar tu pregunta"

Interpretación de ChatGPT: "Ayer tu abuelo te llamó para preguntarte algo."

¿Qué puntaje le daría del **1** al **5**, siendo **1** el más bajo y **5** el más alto, a la interpretación generada por ChatGPT en términos de claridad y coherencia?

19 respuestas

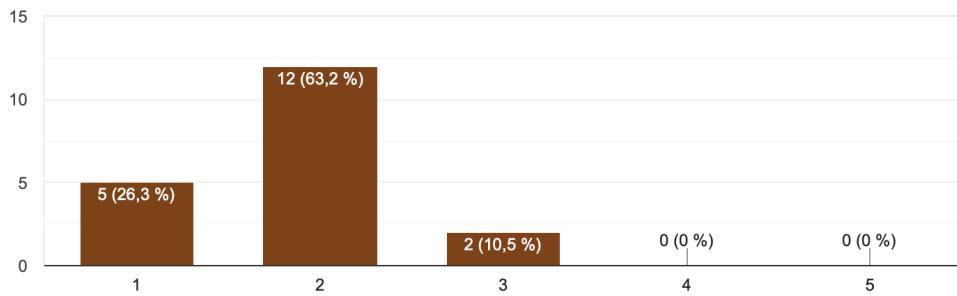


Figura 22: Distribución de calificaciones para la interpretación de la frase No. 3 generada por el modelo estándar con la última versión del *prompt*.

Frase con gramática de LENSEGUA: "ayer abuelo llamar tu pregunta"

Interpretación 1: "¿Ayer te llamó tu abuelo?"

Interpretación 2: "Ayer tu abuelo te llamó para preguntarte algo."

De las dos interpretaciones presentadas, ¿cuál le parece más clara o adecuada según su criterio?

19 respuestas

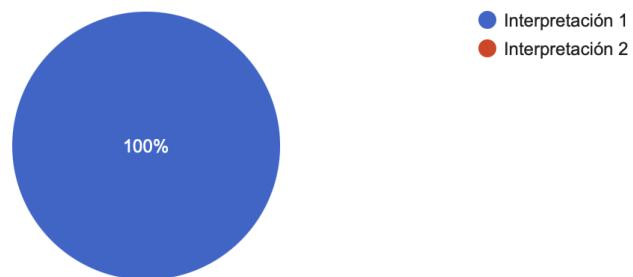


Figura 23: Preferencia de los usuarios entre dos interpretaciones de la frase No. 3: Interpretación 1 (modelo *fine-tuneado*) e Interpretación 2 (modelo estándar).

Frase con gramática de LENSEGUA: "antes tienda tu amiga ver pregunta"

Interpretación de ChatGPT: "¿Viste a tu amiga en la tienda?"

¿Qué puntaje le daría del 1 al 5, siendo 1 el más bajo y 5 el más alto, a la interpretación generada por ChatGPT en términos de claridad y coherencia?

19 respuestas

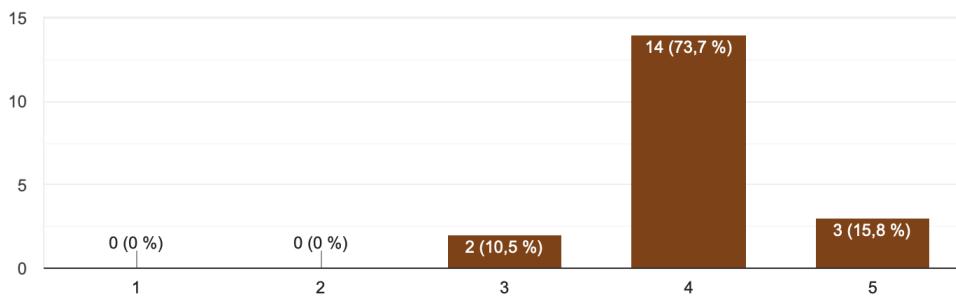


Figura 24: Distribución de calificaciones para la interpretación de la frase No. 4 generada por el modelo *fine-tuneado* con la última versión del *prompt*.

Frase con gramática de LENSEGUA: "antes tienda tu amiga ver pregunta"

Interpretación de ChatGPT: "Tu amiga vio la tienda y te preguntó algo."

¿Qué puntaje le daría del 1 al 5, siendo 1 el más bajo y 5 el más alto, a la interpretación generada por ChatGPT en términos de claridad y coherencia?

19 respuestas

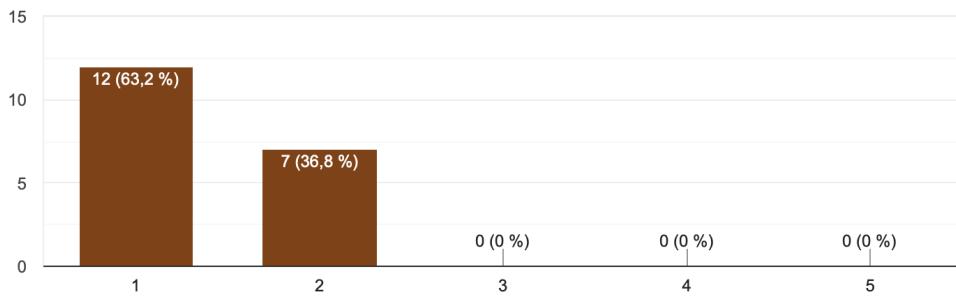


Figura 25: Distribución de calificaciones para la interpretación de la frase No. 4 generada por el modelo estándar con la última versión del *prompt*.

Frase con gramática de LENSEGUA: “antes tienda tu amiga ver pregunta”

Interpretación 1: ¿Viste a tu amiga en la tienda?

Interpretación 2: “Tu amiga vio la tienda y te preguntó algo.”

De las dos interpretaciones presentadas, ¿cuál le parece más clara o adecuada según su criterio?

19 respuestas



Figura 26: Preferencia de los usuarios entre dos interpretaciones de la frase No. 4: Interpretación 1 (modelo *fine-tuneado*) e Interpretación 2 (modelo estándar).

Frase con gramática de LENSEGUA: “tu opinión cuál pregunta”

Interpretación de ChatGPT: “¿Cuál es tu opinión?”

¿Qué puntaje le daría del **1** al **5**, siendo 1 el más bajo y 5 el más alto, a la interpretación generada por ChatGPT en términos de claridad y coherencia?

19 respuestas

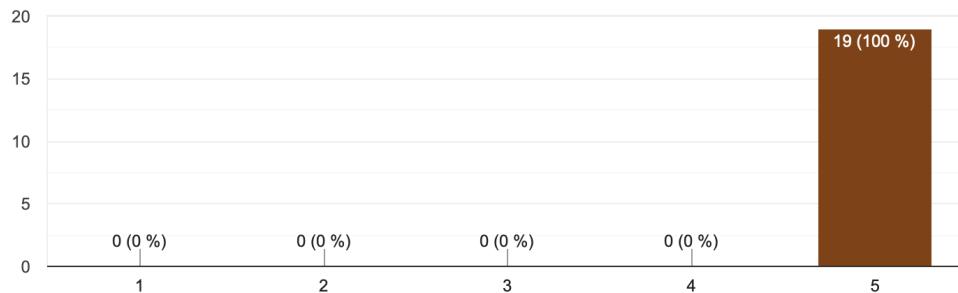


Figura 27: Distribución de calificaciones para la interpretación de la frase No. 5 generada por el modelo *fine-tuneado* con la última versión del *prompt*.

Frase con gramática de LENSEGUA: "tu opinión cuál pregunta"

Interpretación de ChatGPT: "Tu opinión sobre cuál es la pregunta."

¿Qué puntaje le daría del 1 al 5, siendo 1 el más bajo y 5 el más alto, a la interpretación generada por ChatGPT en términos de claridad y coherencia?

19 respuestas

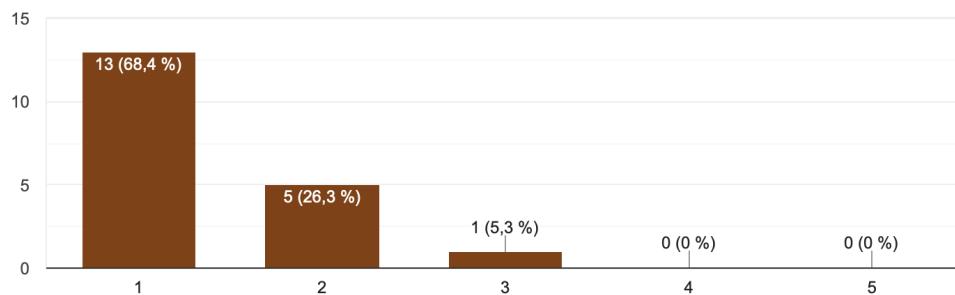


Figura 28: Distribución de calificaciones para la interpretación de la frase No. 5 generada por el modelo estándar con la última versión del *prompt*.

Frase con gramática de LENSEGUA: "tu opinión cuál pregunta"

Interpretación 1: "¿Cuál es tu opinión?"

Interpretación 2: "Tu opinión sobre cuál es la pregunta."

De las dos interpretaciones presentadas, ¿cuál le parece más clara o adecuada según su criterio?

19 respuestas



Figura 29: Preferencia de los usuarios entre dos interpretaciones de la frase No. 5: Interpretación 1 (modelo *fine-tuneado*) e Interpretación 2 (modelo estándar).

Frase con gramática de LENSEGUA: "hospital yo necesitar ir ahora ojalá mucho carro no porque emergencia"

Interpretación de ChatGPT: "Necesito ir al hospital ahora espero que haya muchos carros porque es una emergencia."

¿Qué puntaje le daría del **1** al **5**, siendo 1 el más bajo y 5 el más alto, a la interpretación generada por ChatGPT en términos de claridad y coherencia?

19 respuestas

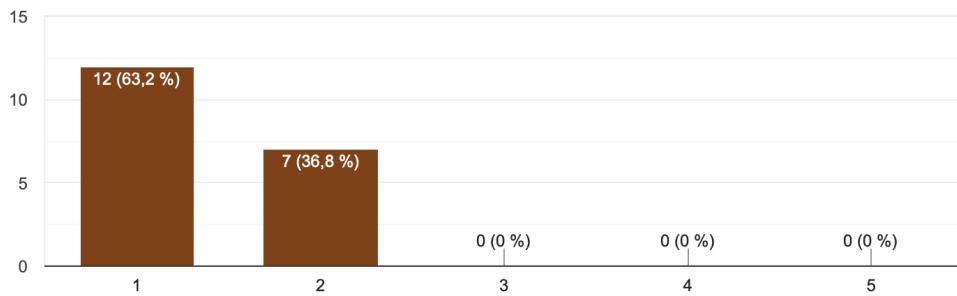


Figura 30: Distribución de calificaciones para la interpretación de la frase No. 6 generada por el modelo *fine-tuneado* con la primera versión del *prompt*.

Frase con gramática de LENSEGUA: "hospital yo necesitar ir ahora ojalá mucho carro no porque emergencia"

Interpretación de ChatGPT: "Necesito ir al hospital ahora espero que haya muchos carros disponibles no tengo carro porque es una emergencia."

¿Qué puntaje le daría del **1** al **5**, siendo 1 el más bajo y 5 el más alto, a la interpretación generada por ChatGPT en términos de claridad y coherencia?

19 respuestas

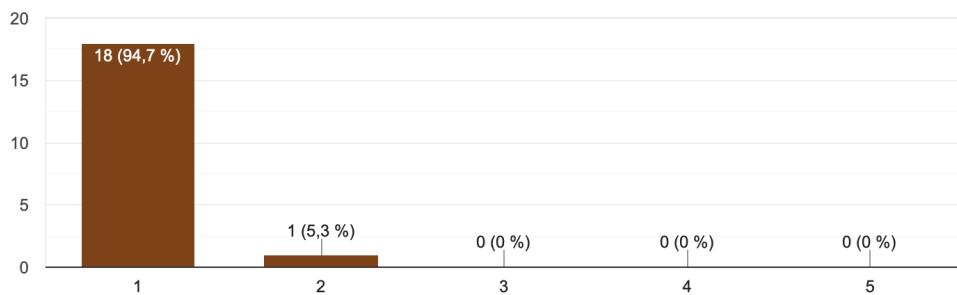


Figura 31: Distribución de calificaciones para la interpretación de la frase No. 6 generada por el modelo *fine-tuneado* con la segunda versión del *prompt*.

Frase con gramática de LENSEGUA: "hospital yo necesitar ir ahora ojalá mucho carro no porque emergencia"

Interpretación de ChatGPT: "Necesito ir al hospital ahora porque es una emergencia y ojalá haya mucho carro."

¿Qué puntaje le daría del **1** al **5**, siendo 1 el más bajo y 5 el más alto, a la interpretación generada por ChatGPT en términos de claridad y coherencia?

19 respuestas

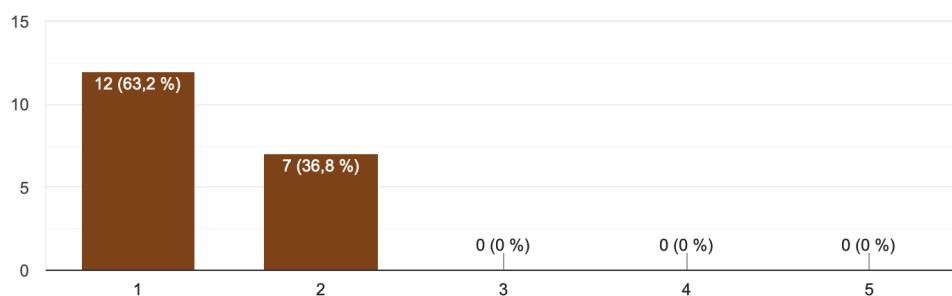


Figura 32: Distribución de calificaciones para la interpretación de la frase No. 6 generada por el modelo *fine-tuneado* con la tercera versión del *prompt*.

Frase con gramática de LENSEGUA: "hospital yo necesitar ir ahora ojalá mucho carro no porque emergencia"

Interpretación de ChatGPT: "Necesito ir al hospital ahora espero que no haya mucho tráfico porque es una emergencia."

¿Qué puntaje le daría del **1** al **5**, siendo 1 el más bajo y 5 el más alto, a la interpretación generada por ChatGPT en términos de claridad y coherencia?

19 respuestas

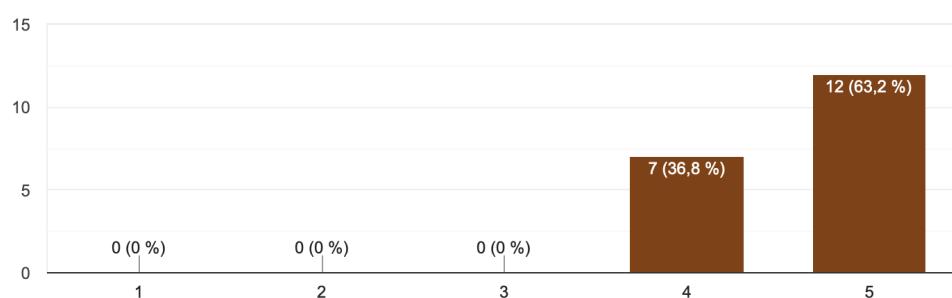


Figura 33: Distribución de calificaciones para la interpretación de la frase No. 6 generada por el modelo *fine-tuneado* con la cuarta versión del *prompt*.

Frase con gramática de LENSEGUA: "hospital yo necesitar ir ahora ojalá mucho carro no porque emergencia"

Interpretación de ChatGPT: "Necesito ir al hospital ahora mismo. Espero que no haya mucho tráfico porque es una emergencia."

¿Qué puntaje le daría del **1** al **5**, siendo 1 el más bajo y 5 el más alto, a la interpretación generada por ChatGPT en términos de claridad y coherencia?

19 respuestas

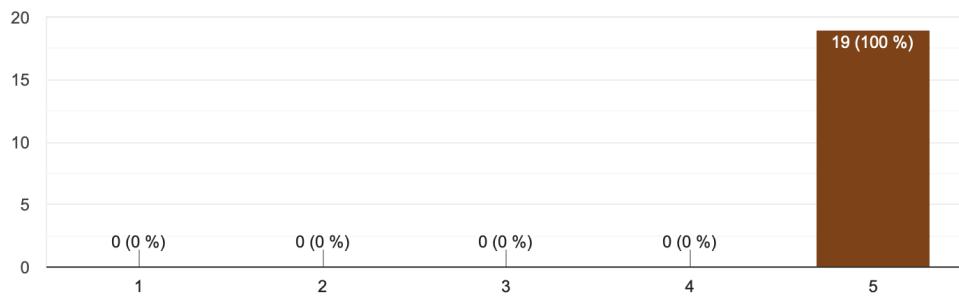


Figura 34: Distribución de calificaciones para la interpretación de la frase No. 6 generada por el modelo *fine-tuneado* con la última versión del *prompt*.

Frase con gramática de LENSEGUA: "hospital yo necesitar ir ahora ojalá mucho carro no porque emergencia"

Interpretación 1: "Necesito ir al hospital ahora espero que haya muchos carros porque es una emergencia."

Interpretación 2: "Necesito ir al hospital ahora espero que haya muchos carros disponibles no tengo carro porque es una emergencia."

Interpretación 3: "Necesito ir al hospital ahora porque es una emergencia y ojalá haya mucho carro."

Interpretación 4: "Necesito ir al hospital ahora espero que no haya mucho tráfico porque es una emergencia."

Interpretación 5: "Necesito ir al hospital ahora mismo. Espero que no haya mucho tráfico porque es una emergencia."

De las cuatro interpretaciones presentadas, ¿cuál le parece más clara o adecuada según su criterio?

19 respuestas

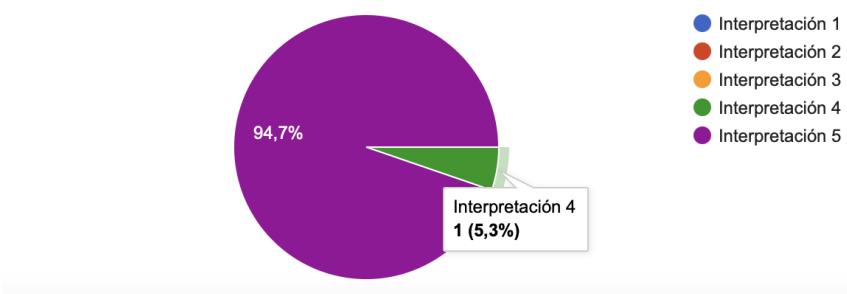


Figura 35: Preferencia de los usuarios entre las interpretaciones de la frase No. 6: Interpretación 1 (versión 1 del *prompt*), Interpretación 2 (versión 2 del *prompt*), Interpretación 3 (versión 3 del *prompt*), Interpretación 4 (versión 4 del *prompt*) e Interpretación 5 (versión 5 del *prompt*).