UNIVERSIDAD DEL VALLE DE GUATEMALA
Facultad de Ingeniería

Señas Chapinas: Traductor de LENSEGUA

Implementación del Modelo Llama para la Interpretación Automática de LENSEGUA

Trabajo de graduación en modalidad de Megaproyecto Tecnológico presentado por Roberto Vallecillos Chinchilla para optar al grado académico de Licenciado en Ingeniería en Sistemas de Computación y Tecnología de la Información

Guatemala, noviembre del 2024

Vo.Bo.:

(f) _____

Ing. Alberto Suriano

Tribunal Examinador:

(f) _____

(f) _____

(f) _____

Fecha de aprobación: .

Índice

Lista de Figuras

Resumen

Este trabajo presenta el desarrollo de un modelo de traducción automática para el lenguaje de señas guatemalteco (LENSEGUA), basado en el modelo LLaMa 3.0 de Meta. La elección de este modelo se fundamenta en su disponibilidad como código abierto y en su capacidad de manejo de parámetros avanzados. La metodología empleada incluye el uso de Google Colab y la librería Transformers, destacando el método de *Low-Rank Adaptation*(LoRA) para optimizar el proceso de ajuste fino.

Se implementó un formato de datos inspirado en el modelo Alpaca, que integra instrucciones y ejemplos de señas, facilitando así el aprendizaje y la adaptación del modelo. La evaluación del rendimiento se llevó a cabo utilizando métricas como BLEU, que comparan la calidad de las traducciones generadas con traducciones de referencia.

Los resultados iniciales mostraron una puntuación BLEU de 0.125, que mejoró a 0.268 tras realizar ajustes en el entrenamiento, indicando que el modelo tiene potencial para ser una herramienta efectiva en la interpretación del lenguaje de señas. Aunque la calidad del modelo no fue el objetivo central del estudio, se validó la capacidad de los LLMs para traducir lenguas menos comunes, lo que sugiere un amplio potencial para futuras aplicaciones en la comunicación a través de señas.

CAPÍTULO 1

Introducción

Según la Organización Mundial de la Salud, 488 millones de personas tienen alguna condición de sordera, o con baja audición (Wolmark, 2023). Solo en Guatemala, hay un total de 529,920 personas conocidas que tienen alguna condición de sordera respectiva (Congreso de la República de Guatemala, 2022). Existen varios métodos para poder ayudarlas, siendo uno de estos, el aprendizaje de lenguaje de señas y fomentar su aprendizaje, por aplicación o habla común, donde se tiene su propia gramática y se tienen palabras y expresiones relacionadas para cada seña en específico.

El proyecto principal, Señas Chapinas, es una aplicación para poder traducir el lenguaje de señas basado en inteligencia artificial (IA). El procedimiento de la aplicación consiste en un input, que sería una seña para traducir, y basado en *training* y en *Machine Learning*, determinar que seña es. Una vez que se traduzcan todas las señas y se determinen cuáles son las señas respectivamente, se procede a la interpretación a través de un *Large Language Model* (LLM), para formar oraciones coherentes.

Este trabajo de graduación representa uno de los módulos del proyecto principal. Consiste en poder trabajar en un sistema de interpretación de palabras de lenguaje de señas por medio de un LLM para poder formar oraciones con coherencia gramatical y completar el sistema de traducción del lenguaje de señas. Una interpretación que se pueda entender y que sea fácil de comunicar. Para realizar esto, el LLM que se va a utilizar es conocido como LLaMa, específicamente la versión 3.0, de Meta, la versión más reciente de LLaMa, (Meta, 2024). Se va a entrenar este LLM para poder entender el contexto y la gramática del lenguaje de señas, y así, poder devolver resultados que tenga sentido.

CAPÍTULO 2

Justificación

La IA es unos de los campos que se están expandiendo de forma más rápida en tiempos recientes (Metz, 2012). Una de estas maneras está relacionada a la traducción de diferentes idiomas a otros. Un uso particular de este sistema de traducción esta dedicado a ayudar a personas con problemas de escucha (Toche, 2021). En Guatemala, según el Congreso de la República, el 10.2 % de la población sufre una discapacidad. De ese porcentaje, un total de más de doscientas cincuenta mil personas sufren de una discapacidad auditiva (Congreso de la Republica de Guatemala, 2022). De estas personas que no pueden escuchar, varias solo pueden comunicarse con el lenguaje de señas. El reflejo de esta situación se puede observar en las transmisiones televisivas nacionales, donde se tiene una traducción en tiempo real a lenguaje de señas para las personas con problemas de audición.

Existen diferentes modelos de LLM, LLaMa es uno de los más populares hoy en día. El proyecto principal estipula usar varios modelos, y en este caso se quiere probar LLaMa, uno de los mayores LLMs, diseñado por Meta, la compañía detrás de Facebook (Meta, 2024). De todos los modelos que existen se escoge este, debido que Meta es una de las compañías más importantes en cuanto a IA y en la recolección de datos, justo al lado de OpenAI, el LLM más famoso hoy en día, debido a su naturaleza de open source (Kumar, 2024). Otro de los modelos considerados por el proyecto principal contempla el uso del modelo de OpenAI. Basado en esto, poder comparar y contrastar cómo funcionan ambos en un ambiente real, mostrando las fortalezas y debilidades de cada uno cuando se quiere escoger un modelo LLM con que trabajar.

CAPÍTULO 3

Objetivos

3.1. Objetivo General

1. Desarrollar un modelo de inteligencia artificial (IA) que pueda interpretar palabras traducidas del lenguaje de señas a oraciones gramaticalmente coherentes para el ambiente guatemalteco.

3.2. Objetivos Específicos

- Preprocesar información para que esta pueda ser entendida por un modelo de IA.
- Desarrollar una herramienta en base a LLaMa que reciba traducciones del lenguaje de señas.
- Realizar pruebas del modelo a terceros expertos para demostrar su funcionalidad.
- Combinar el funcionamiento de esta herramienta con una aplicación de traducción de lenguaje de señas.

CAPÍTULO 4

Marco Teórico

4.1. Guatemala y la Sordera

En 2016, se realizó la II Encuesta Nacional de Discapacidad en Guatemala. De todos los participantes de la encuesta, un total de 16.8 % sufre de una discapacidad auditiva parcial o total. Según el Congreso de la República, el 10.2 % de la población sufre una discapacidad. De ese porcentaje, un total de más doscientas cincuenta mil personas sufren de una discapacidad auditiva[7]. Y de estas personas que no pueden escuchar, varias solo pueden comunicarse con el lenguaje de señas.

4.1.1. Sordera

Según el Congreso de Guatemala en el Decreto 2020-3, se define como síndrome a la discapacidad de la pérdida parcial o total de la audición y sus diferentes modalidades[9].

4.1.2. Lengua de Señas

En el mismo decreto, se define como un sistema de comunicación natural con gramática única, solo utiliza los sentidos visuales y táctiles para referirse a gestos, formas, mímicas y movimientos en un territorio en específico, utilizado principalmente por personas sordas para poder comunicarse.

Gramática del Lenguaje de Señas

La gramática de un lenguaje es reconocido como el conjunto de reglas que forman a una lengua en base de su estructura, todo con base en su análisis sintáctico.

El idioma español tiene su propio abecedario para poder comunicarse en caso de sonidos respectivos, y también al combinar todo en oraciones, se forma una estructura de Sujeto, Verbo, Objeto, en donde el Sujeto, el actor, va de primero, el verbo, la acción, va de segundo, y el objeto, a que o quien el actor está afectando, vá de último. Por ejemplo, “El niño corre la pelota.” Se tiene al sujeto, el niño, el verbo, corre, y el objeto, la pelota. La combinación del objeto y el verbo es conocido como el predicado.

En cuanto a lengua de señas, también tiene una estructura de Sujeto, Verbo, Objeto solo que esta no tiene el beneficio de palabras transitorias como artículos, preposiciones, y otros para que sea entendible. Tiene característica de poco vocabulario por una comunicación más rápida. También está permitido el uso de Sujeto, Objeto, Verbo. El verbo tampoco tiene conjugación, ni hay tiempos verbales.

También en el sistema de Sujeto, Objeto, Verbo, existe el Tiempo y el Lugar, para dar detalles más específicos al comunicarse.

Ejemplo de una oración en español usando Sujeto, Verbo, Objeto

Hoy, yo voy a limpiar mi casa.

Ejemplo de una oración en lenguaje de señas.

HOY CASA YO LIMPIAR.

4.1.3. LENSEGUA

Desde el 2000, LENSEGUA, o El Lenguaje de Señas de Guatemala fue diseñado para ayudar a las personas que tienen problemas de . Esto eudición importante debido a que no hay un estándar mundial de la lengua de señas, cada país y región tiene sus propias reglas generales en cómo poder comunicarse con la lengua de señas.

En 2020, en la IX Legislatura, se aprobó el Decreto 3-2020, que decreta la Ley que Reconoce y Aprueba a la Lengua de Señas de Guatemala. En esta misma, se decreta que "LENSEGUA" sea el medio de comunicación conformado por un conjunto específico de gestos, formas, mímicos movimientos como la gramática respectiva las personas sordas en la República de Guatemala.

En 2021, se realizó el Acuerdo Gubernativo 121-2021, donde explica y sigue adelante en la educación y fomento de LENSEGUA para estudiantes con discapacidades y también a docentes para ayudar atender a estas personas. También explica que el Ministerio de Educación promoverá haciendas de educación inclusiva para estos estudios y ejercer -LENSEGUA-

Este es el estándar en Guatemala, sin embargo, debido a las diferentes regiones de Guatemala, este también ha cambiado, similar al español con ciertos languages coloquiales, LENSEGUA también tiene su propio situación en que existen situaciones únicas para diferentes señas-

4.2. Inteligencia Artificial

La inteligencia artificial se reconoce como un sistema informático, que simula el razonamiento humano al enfrentarse con decisiones y aprendiendo de esas mismas decisiones y nueva información para tomar nuevas decisiones.

Este término fue inicialmente adoptado en 1956, en la Conferencia de Dartmouth, en donde se predijo que en 25 años, las computadoras llegarían a un umbral en donde podrían resolver cualquier problema que el ser humano no pudiera por sí mismo y más avanzado que nunca.

4.2.1. Deep Learning

Deep learning se reconoce como todo el conjunto de machine learning donde utiliza redes neuronales, o también conocidas como redes neuronales profundas para simular el comportamiento del

cerebro humano. Estos están configurados principalmente por tener más de tres capas, aunque se conocen por utilizan miles de capas.

Red Neuronal

Una red neuronal artificial, es un método de entrenamiento para máquinas que se dedica a procesar datos, simulando las neuronas en el cerebro. Cada nodo está conectado de manera que está simulando por capas, y así mismo, aprende de sus errores y mejora constantemente su modelo para que se adapte a máquinas y resuelva problemas complejos con alta precisión. Estas se diseñan para determinar información de las capas.

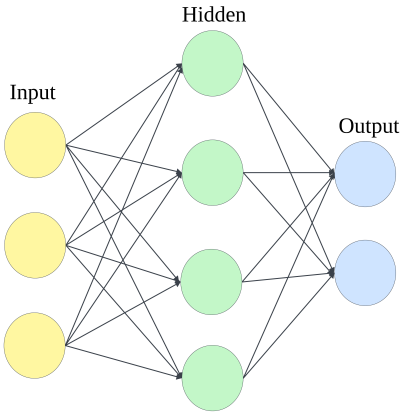

Figura 4.1: Estructura de una Red Neuronal

4.2.2. Modelo Básico de una Red Neuronal

Capa de Entrada

Es la capa donde se ingresa información cuál sea necesaria en forma de vector.

Capas Ocultas

Estas capas ocultas entre las capas de entrada y de salida, afecta ta a la información, transformándola basado en sus diferentes pesos, sesgos y también la información en las funciones de activación.

Capas de Salida

La capa donde se obtienen los resultados finales ya transformados por las capas anteriores.

4.2.3. Fine Tuning

Fine-Tuning es el proceso de ajustar o adaptar un modelo para que pueda realizar tareas específicas o función en un caso único.

Es una herramienta de entrenamiento semi supervisado, en vez de volver a entrenar a un LLM por completo desde 0, lo que sería un proceso muy costoso y drena a una gran cantidad de recursos similares, utiliza lo ya entrenado y agrega una capa superficial de entrenamiento con informacion dedicada a la nueva actividad el modelo.

4.2.4. Modelos LLM

Un modelo LLM (*Large Language Model*) es un tipo de modelo de inteligencia artificial diseñado para comprender y generar texto en lenguaje natural. Estos modelos, están entrenados en grandes cantidades de texto y utilizan técnicas de aprendizaje automático o semi automática para generar respuestas coherentes y contextualmente apropiadas. Estos modelos pueden ser escalables, y también están basados en los contextos alrededor. [10] [2]

NLP

NLP (*Natural Language Processing*) es un parte de la inteligencia artificial la cual se dedica a entender, interpretar, y en general lenguaje humano.

Lingüística computacional

El proceso en que el lenguaje humano natural puede ser dividido en un sistema para que una computadora pueda entender individualmente en cada paso para que la computadora la pueda utilizar.

NLU

NLU (*Natural Language Understanding*) es un subconjunto de NPL, donde está dedicada a entender las palabras que están en las oraciones, y entender el significado mientras también se entiende el contexto respectivo en su alrededor.

NLG

NLG (*Natural Language Generation*), es el subconjunto de NLP donde se dedica a generar lenguaje natural hablado o escrito para ser entendido. Genera de una manera que basado en el input pueda ser entendible el humano que lo lea.

Transformador

Un transformador es una arquitectura de redes neuronales que aprenden dependiendo del contexto, y aprende basado en la información que tiene a su alrededor.

Utiliza el método de atención para revisar cómo las palabras que están a su alrededor lo afecta a los *tokens*. [14]

Atención

Simulando el proceso de atención humana, el proceso de atención asigna niveles de importancia en una palabra. En cada palabra, se calcula un peso de la palabra en el proceso de embedding, y lo coloca en un vector en su contexto para determinar la importancia de la misma manera. Estos pesos también pueden cambiar basado en el entrenamiento y uso del modelo.[15]

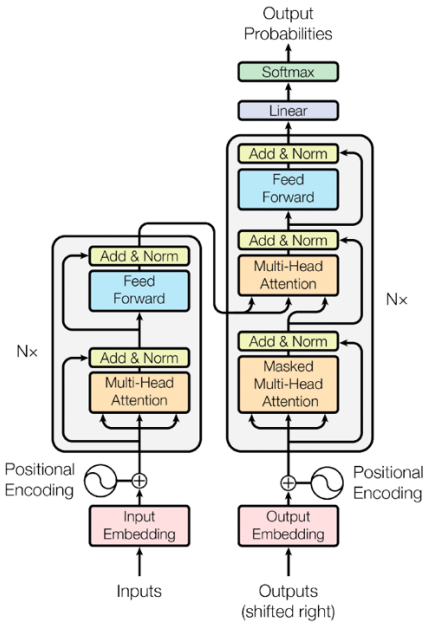

Figura 4.2: Estructura de un transformador.[1]

La entrada de texto se descompone en *tokens* y se incorpora en representación de vectores. Aquí se captura el significado semántico y sintáctico de la información.

Codificación Posicional

El proceso de codificación policial consiste en que cada palabra tiene una codificación en su posición para proveer información sobre la posición de los *tokens*, y de esta manera, entender y colocar el orden adecuado y que lo tenga guardada.

Codificación Posicional Absoluta

El proceso de *embedding* debe guardar la información de su posicióón en la oración. Para esto, se utiliza un vector de dimensionalidad igual para representar una oración, y adentro su posición, haciendo que cada vector tenga su propio vector respectivo.

Es la forma más sencilla de trabajar, y también los vectores posicionales que se aprenden durante en entrenamiento, y también utiliza funciones sinusoidales como funciones de activación haciendo más eficiente.

Sin embargo, este sí tiene un límite, no puede aprender más información y se queda sin aprender

más. De la misma manera, el embedding hace que no haya diferencia entre los lugares, inclusive si están a una distancia de 100, y hace que se pierda el significado semántico de las palabras.

Codificación Posicional Relativa

A diferencia entre codificación absoluta, en la codificación relativa los embeddings se concentran en la distancia de pares en otros, donde altera el mecanismo de incorporar información relativa basada en su posición. En el caso del modelo T5, este utilizó un sesgo para representar cada posición, donde no importe su posición absoluta, siempre habrá un sesgo si dos palabras están a la par. La matriz para todas las palabras en su posición relativa permite que sus sesgos sean los mismos.

No obstante, guardar toda información tomaría mucho recursos, e incluso su alta escalabilidad de guardar información, se necesitaría más recursos que afectaría su rendimiento. También afecta el funcionamiento de los Transformadores debido a que afecta los valores de los key-values caches.

Codificación Posicional Rotacional

Codificación Posicional Rotacional, o RoPe combina los atributos de codificación absoluta y relativa. En vez de sumar al vector posicional, aplica una rotación. Esto ayuda a que se guarde la información sin problema al final y después al principio, y también guarda la información de los vectores relativos en diferentes contextos, ya que se rota los vectores por la misma cantidad. El producto punto de los vectores son constantes.

Hay un codificador general que analiza el texto de ingreso y crea una cantidad de diferentes estados para proteger el contexto y definición del texto. Se guardan diferentes capas para la arquitectura.

Atención Multicabezal

Atención multicabezal es el uso de atención única simultáneamente con los pesos de atención, lo que permite que el modelo puede capturar de diferentes tipos de relaciones y las diferentes partes de la secuencia del texto de entrada.

Capa Normalización

Una capa de normalización se aplica en cada componente de la arquitectura para estabilizar el proceso de aprendizaje y generar diferentes inputs.

Capa de salida

En la capa de salida, basado en el diseño específico, se utiliza una proyección para poder predecir el siguiente token que se aplicaría.

Funciones de activación

Gated Linear Unit (GLU por sus siglas en inglés) es una la función de activación en la red neuronal de los transformadores. Similar a una puerta o garita, controla el flujo de información

En la librería de transformadores, se utilizá *Switched Gated Linear Units* (swiGLU en siglas en ingles), es la combinación de GLU más la función de activación Swish, una función que combina la características de una función de activación lineal y una función de activación ReLU. Esto hace que swiGLU sea más flexible y pueda capturar información más compleja con un simple GLU.

Estas se utilizán en los transformadores debido que necesita pasar información, dependiendo del contexto, y miran que tanto de la información puede pasar o no a través de la puerta.

4.2.5. Ingeniería de instrucciones

Ingeniería de instrucciones, también conocido como Prompt Engineering, es el proceso que se utiliza para entrenar LLMs.

Indicador

Un indicador o instrucción o prompt, es un texto de entrada donde se indica un modelo de lenguaje para poder realizar la información más específica y relevante, para poder optimizar el uso máximo de la inteligencia artificial. Se utiliza para que el modelo pueda entender el contexto de la petición que se indica, y puede ser customizable al trabajar en esto.

In Learning Context (Aprendizaje en Contexto)

El aprendizaje por contexto es definido como que un LLM aprenda y sepa a partir de su contexto respectivo. Este está siendo afectado por el contexto del indicar para que ayude aprender.

Para demostrar esto, tenemos una fecha: 20 de julio del 2024. En el aprendizaje por contexto, le pedimos en el indicar que cambie esto todo aún formato en específico en el tiempo de día/mes/año o 20/7/2024. Lo que permite esto, es que por su propia cuenta, pueda aprender múltiples veces hasta que la próxima vez que alguien pregunte por una fecha, le de en el formato el cual se haya entrenado.

4.3. META

Meta Platforms, Inc., o Meta, es una empresa tecnológica, la cual es fundadora de FaceBook, con Mark Zuckerberg como CEO, y dueña de Instagram, WhatsApp y Messenger.

Debido al surgimiento de popularidad de modelos de lenguaje, como GPT-3, una de las prioridades de Meta fue crear su propio LLM para uso en general.

Zuckerberg menciona que la IA debe ser *Open Source* para el futuro, y está es la razón por la que su propio LLM, LLaMa, es *Open Source*.

4.3.1. LlaMa

LLaMa, o por sus siglas en inglés, Large Language Model Meta AI, es un LLM diseñado por la compañía Meta, lanzado a principios del 2023.

Una de las mayores diferencias entre LLaMa y otros modelos LLM es *Open Source*, lo que significa que cualquier persona tiene acceso a su código base y poder modificarlo.

LLaMa es diferente a GPT debido no solo a su uso general no comercial, pero también por cantidad menos intensiva de recursos para poder correrlo y entrenarlo. Esta diferencia permite que sea más accesible para el público, y que sea más poderosa para ciertos casos, como la escritura [13].

4.3.2. Arquitectura de LLaMa

LLaMa está diseñada similarmente con la arquitectura de transformadores mencionada previamente y su mismo procedimiento, con ciertas diferencias:

Utilizando una función de activación de SwiGLU en vez GLU. Normalización de raíz cuadrada promedio en las capas ocultas

4.3.3. Entrenamiento

Todo el entrenamiento de la LLaMa, según Meta, fue entrenado por libros en dominio público, sitios web *Open Source*, fuente de código LaTeX para sitios científicos entre otros. Para poder mostrar que todos los datos conseguidos de la LLM fueron obtenidos de manera legal y solamente para uso público, un projecto llamada RedPajama fue utilizado para poder igualar el proceso de entrenamiento set de entrenamiento utilizado, y mostrar todas la cantidad de *tokens* utilizados, que fue en un total de 30 trillones de *tokens* diferentes [12].

4.3.4. LLaMA 3.0

La versión mas actual al tiempo de escribir este trabajo de graduación es la versión 3.0 que 15 trillones de *tokens*, en diferentes idiomas respectivos. Tienen dos modelos dependiendo de la cantidad de parámetros, uno de 8.000 millones y otro de 70.000 millones (8B Y 70B respectivamente.

Lo más importante es que tienen versiones en diferentes medios, al poder ser usado con otras aplicaciones y productos de meta, integrándose con Meta IA, y siendo usado por FaceBook y Messenger.

4.4. Análisis de Métricas

Al final del entrenamiento, se realizaron pruebas para poder probar la calidad del modelo, y ver sus ventajas y desventajas.

Existen varias métricas para demostrar el funcionamiento de una traducción de un lenguaje a otro para poder verificar que el texto generado se asemeja a una traducción humana respectiva[4].

4.4.1. BLEU (Bilingual Evaluation Understudy)

BlEU o *Bilingual Evaluation Understudy* es una métrica que mide la calidad de la traducción que compara la salida que se genera con las traducciones originales de referencia. Su función es que el resultado de la traducción de la LLM se asemeje lo más posible con la traducción de la que fue entrenada.

Esta funciona midiendo sus N-grams con la precisión, que compara el texto de referencia, n-gramas siendo unigramas (una palabra), bigramas (dos palabras) o trigramas (tres palabras). Basado

en esto, se toma una penalización de brevedad que mira el largo de la semilla y penaliza si es más corto de lo que fue entramado, y la precisión de n-gramas, que calcula la proporción del texto generado con el de referencia para obtener la puntuación final. Esta se toma de 0 a 1, con 1 siendo la más precisa y 0 la menos.[11]

4.4.2. LIME (Local Interpretable Model-Agnostic Explanations)

LIME, o *Local Interpretable Model-Agnostic Explanations*, es una técnica que busca proporcionar interpretaciones de las decisiones de un modelo de machine learning al centrarse en sus predicciones en un vecindario local alrededor de una entrada específica. LIME proporciona una forma de entender cómo un modelo llega a su decisión para un caso particular.

Para LIME, se crea un modelo interpretable que se ajusta a las predicciones del modelo complejo en un área pequeña alrededor de la instancia cual se analizanda. Esto se hace al perturbar la entrada original y observar cómo cambian las predicciones del modelo.[3]

CAPÍTULO 5

Metodología

El Modelo de LLaMa fue seleccionado debido a ser parte de una de las compañías de información más grándes, siendo esta Meta (Facebook), a su vez, también el hecho que tenga un código *Open Source*. [13]

La versión más actual al momento de realizar este trabajo es la versión de LLaMa 3.0 que ha sido entrenado, siete veces mejor que LLaMa 2 y que también alcanza los 15 trillones de *tokens* respectivos. LLaMa 3 viene en dos diferentes modelos, 8B y 70B, que representan la cantidad de parámetros en millones. Por temas de recursos disponibles, se utilizó la versión de 8B para poder trabajar en este módulo.

Ambos modelos de LLaMa, necesitan una alta cantidad de memoria para que se pueda correr correctamente, y que sea eficiente al poder entrenar, y obtener un resultado de alta calidad. Esto es lo que lleva al proceso de cuantización. Este proceso hace que baje la precisión exacta de los pesos del *Large Language Model* (LLM), (de 32 bits a 8 bits por ejemplo), y sacrifica la precisión final del modelo, para abrir camino a más eficiencia en cuanto a recursos, menos memoria para poder cargar, y también menos tiempo para realizar los cálculos de los pesos.

En este caso, se cargaron todos los modelos en una cuantización de 8 bits para poder entrenarlos de manera rápida y eficiente, debido a los límites de recursos disponibles.

Vale mencionar que si es cierto que LLaMa es *Open Source* y cualquiera puede utilizarla, se necesita una licencia aprobada del mismo Meta para poder descargar localmente y poder trabajar con esta para que realizar *fine-tuning* respectivo.

5.1. Utilización de Google Colab y Transformers

Transformers es una librería especial donde se puede instalar directamente a la computadora de varias maneras, en este caso siendo instalada en Python para poder trabajar de la siguiente manera.

Uno de los requisitos especiales de Transformers, es que se necesite una Unidad de Procesamiento Gráfico (GPU, por sus siglas en inglés) que pueda utilizar Arquitectura Unificada de Dispositivos de Cómputo (CUDA, por sus siglas en inglés). Como se ve en la Fig. 5.1, la habilidad de las GPUs de poder realizar trabajo en paralelo, hace que su mayor eficiencia sea fundamental para poder entrenar

redes neuronales complejas, debido a su capacidad de correo millones de hilos simultánea y en forma paralela, en el área especial de entrenamiento.

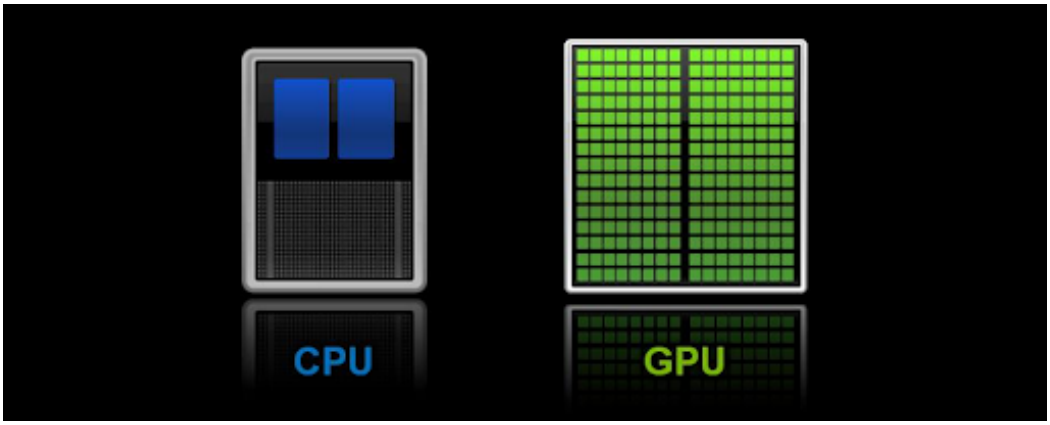

Figura 5.1: Composición de CPU y un GPU[5]

CUDA es una plataforma diseñada por NVIDIA para poder realizar este procesamiento en sus GPUs para poder maximizar el poder computacional para sus diferentes aplicaciones. Esta habilidad de poder trabajar simultáneamente con toda la información para informar al LLM lo hace de la forma más ideal para poder trabajar. [5]

En este caso, se utilizo Google Colab, un ambiente de la nube de Google para simular un GPU local. En Google Colab se tiene la capacidad de poder llamar un modelo de LLaMA 3.0 y poderlo entrenar.

5.2. LoRA (Low-Rank Adaptation)

LoRA es un método que fue desarrollado por Microsoft. Las LLMs tienen un alta cantidad de parámetros, que hacen que el entrenamiento completo sea costoso en temas de recursos e impráctico, debido a la actualización de todos los pesos respectivos.

El método de LoRA resuelve esta tribulación al aplicar matrices de bajo rango o un rango menor al rango completo del modelo, para ciertas capas de un modelo pre entrenado, como se mira en la Fig. 5.2. Esto permite una mayor facilidad de entrenamiento y eficiencia al mismo tiempo. Debidos a límites que se tienen por recursos, esta manera la mayor efectivo.

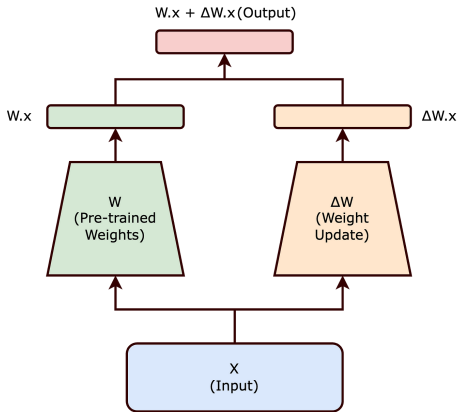

Figura 5.2: Diagrama del Entrenamiento con LoRA[8]

5.2.1. Hiperparametros

Rango

El rango es la dimensión del espacio de bajo en cual se hace la adaptación, donde se define el tamaño de las matrices en la adaptación para modificar el modelo. El rango especifica la cantidad de componentes que se utilizan para representar en los ajustes en los parámetros del modelo base. El rango es importante porque define la capacidad computacional de la máquina al tratar de adaptar el modelo lo mejor posible. Se necesitan más recursos para tener una mayor capacidad de adaptarse, con posibilidad de overfitting.

Alpha

El alpha es la magnitud de la adaptación que se realiza a los parámetros del modelo base. Lo que significa que actúa como un factor de escala para ver cuánto es la influencia en la adaptación del bajo rango del modelo. Un alpha más alto incrementa la magnitud de los ajustes. Alpha muy alto puede hacer sobreajuste mientras que una bajo indica que no es suficiente para capturar las características de los datos.

5.3. Formato Alpaca

Alpaca es una réplica de una LLM de la Universidad de Stanford tomando basde LLaMa, teniendo una forma especifica.

El modelo de Alpaca consiste del siguiente formato json.

ïnstruction": “”, ïnput": “", .ºutput":”

En el JSON, en *instruction*, se enfoca la instrucción que se desea realizar al LLM, o cual es la instrucción a tener que completar.

El *input*, es un parámetro opcional en donde se da un contexto de lo que necesita hacer la instrucción, o de que necesita trabajar basado en la instrucción respectiva.

Finalmente, está el *output*, donde da la respuesta deseada según la instrucción que se va a generar. En este caso, aquí ingresarán las palabras del lenguaje de señas respectivamente, en el orden dado, para después realizar una frase correcta. Aquí en el *output*, se ingresa la frase deseada y que a discreción del creador, va a ser reflejado en los resultados finales.

Esto alimenta al LLM y hace que aprenda y realice el *Fine Tuning* respectivo para poder trabajar en futuros prompts que sean pedidos.

De un set de datos, se va a realizar el *fine-tuning* para crear la LoRA respectiva en donde se analizó la calidad de los resultados. El set de datos fue creado por el mismo autor con una competencia del idioma español y conocimiento de la gramática y señas del lenguaje de señas. De aquí se realizará una separación de 70 %, 15 % y 15 % entre entrenamiento, validación y de prueba también.

Esta base de datos consiste de la siguiente estructura

ïnstruction": “”, ïnput": [SEÑAS REALIZADAS], .ºutput":[ORACIÓN TRADUCIDA]

5.4. Repetición y ajustes

Una vez establecidas las métricas de entrenamiento y rendimiento respectivo, basado en estos resultados, se vuelve a entrenar y hacer *Fine Tuning* al modelo para encontrar posibles mejoras del modelo para que sea más certero y preciso, y conseguir los mejores resultados basado en las modificaciones. Basado en todas las pruebas y diferentes condiciones que se hayan realizado, se decide cual es la mejor y explicar la situación y compararlos entre sí para determinar cual podría ser mejor.

5.5. Comparación con OpenAI

Al terminar y al elegir el mejor modelo, se va a realizar la comparación con OpenAI con su modelo LLM, GPT-3.5-Turbo ya también entrenado para hacer la interpretación de LENSEGUA. En ese modulo, se creo una base de datos de más de 4,000 entradas para entrenarlo y asegurarse que sean de alta calidad y que afecten de una manera significante el modelo. Con esta base de datos se va a volver a entrenar el modelo elegido de LLaMa, para realizar el análisis entre los dos. Basados en el análisis de LIME, y también los criterios de la distancia de Levenshetein, se va a mirar que modelo funciona mejor y cuales son las mayores diferencias entre ambos.

CAPÍTULO 6

Resultados

6.1. Resultados

En la sección de resultados, se van a demonstrar que se obtuvó al probar con los diferentes modelos con LLaMa 3.0.

Estos resultados, salieron durante en el entrenamiento de la LLM, donde al crear el modelo por sí, y poner todos los parámetros deseados, se presenta la pérdida en cada 10 ciclos, como se mira en la Figura 6.1, donde muestra la pérdida de entrenamiento y la pérdida de validación, para percibir el avance en el entrenamiento del LLM.

Después de realizar el entrenamiento, se muestra una predicción de las traducciones obtenidas para mostrar una posible interpretación de la información en formato de oración.

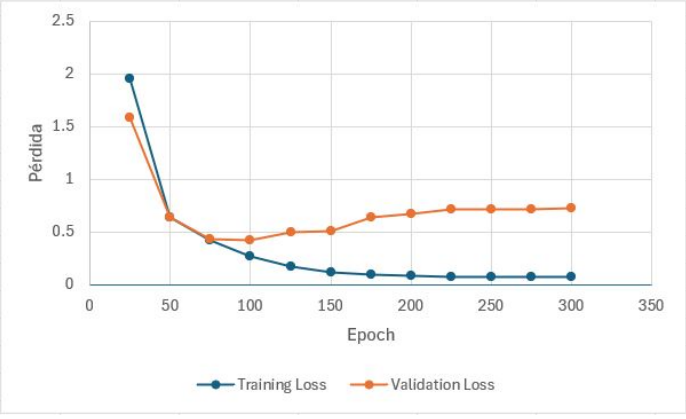

Figura 6.1: Pérdida del modelo preliminar

Como se mencionó en la metodología, se utilizó BLEU para mostrar los resultados. Al realizar la primera prueba solo para verificar que funciona el modelo, se puede encontrar el siguiente resultado. En la Figura 6.2, se puede determinar una constante similitud entre las pérdidas. En 5 horas, el modelo entrenado daba un resultado BLEU de 0.125.

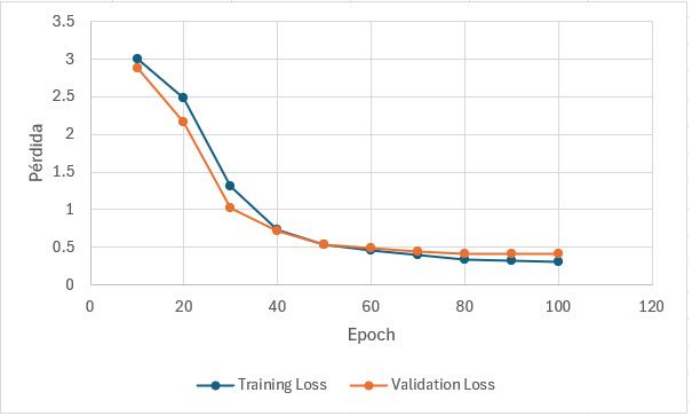

Figura 6.2: Pérdida del modelo aplicando Early Stopping

Al limitar la cantidad de ciclos, aplicando un *Early Stopping* y bando de 300 hasta solamente 100. En la métrica de BLEU, se obtuvo un resultado de 0.21.

Al tratar de mejorar el modelo, se decidio formar un estándar general para poder aplicar en todas las situaciones. El estándar menciona es respecto a los parámetros de LoRA, el rango y alpha, siendo de 8 y 16 respectivos. Este estandár de traducción de un idioma a otro fue utilizado para la Figura 6.3. Esto fue basado en el estudio de traducción bilingües. [16]

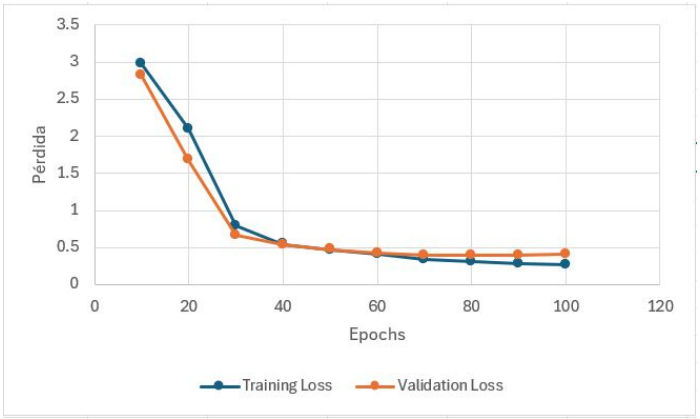

Figura 6.3: Pérdida del modelo aplicando *Early-Stopping* y estándar

El cambio después de entrenarlo cambio la métrica BLEU aumentado el valor 0.239.

Todos los modelos anteriores han sido entrenedos por un entrenador génerico, que este es más permisivo depeniendo a lo que se quiere realizar, entrenamiento supervisado y entrenamiento no supervisado. Tomando la decisión de entrenar el modelo bajo supervisión, se volvió a entrenar el

modelo también, y tener resultados de este caso (ver Figura 6.4).

Pero al quitar la flexbilidad y este completamentado enfocado a un *fine-tuning* supervisado, se pudo hacer un cambio en la métrica de BLEU, aumentado el valor a 0.262.

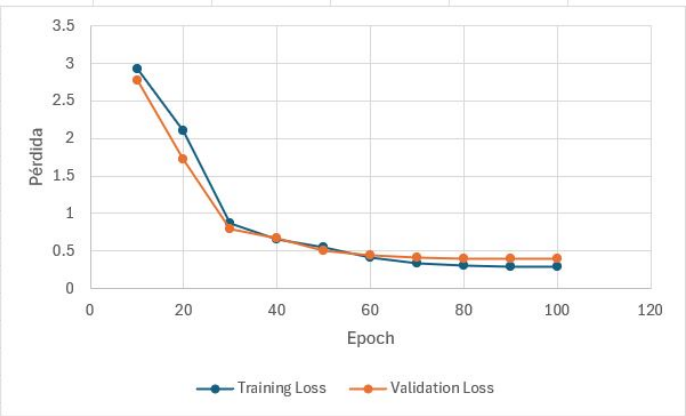

Figura 6.4: Pérdida del modelo aplicando *Early-Stopping* y *Fine-Tuning* Supervisado

Con todos estos modelos, los resultados finales de todos los modelos entrenados están en la siguiente tabla (Tabla 6.1).

Tabla 6.1: BLEU de oraciones

Modelo	BLEU
Modelo sin *Early Stopping* Génerico	1.25
Modelo aplicando *Early Stopping*	2.12
Modelo aplicando *Early Stopping* y estándar	2.39
Modelo aplicando *Early Stopping* y *Fine Tuning* supervisado	2.63

Sé decidió seguir trabajando con este modelo. Y se volvió a entrenar el modelo con más información para revisar si se tienen mejores resultados. Aumentado esto a una cantidad de 560, para obtener el BLEU hasta 0.2894, ya llegando a 0.3.

Con este modelo, se agarrón tres oraciones y se obtuvieron sus resultados BLEU, como se ve en la Tabla 6.2.

Tabla 6.2: BLEU de oraciones para modelo *Early-Stopping y Fine-Tuning* Supervisado

Oración Esperada	Oración Real	BLEU
'¿Como esta el día hoy?	'¿Como esta el día hoy?	1.0
'Miguel gana la carrera.'	'Miguel gano la carrera.'	0.0
'Pon la flor azul sobre la mesa.'	'Pon la flor azul en la mesa'	0.5
'Pon la flor azul sobre la mesa.'	'Pon la flor azul en la mesa'	0.0

Al tomar tres oraciones y observar su score de BLEU, se puede ver los diferentes resultados, variando entre 1.0, 0.0, y 0.5.

Una vez completado el *fine-tuning* y tomar la decisión de utilizar el último modelo, se empezo a utilizar la técnica de *Local Interpretable Model-agnostic Explanations* (LIME, por sus siglas en inglés) para evaluar el impacto de este proceso en las interpretaciones generadas por el modelo.

LIME tomo en cuenta los tokens y frases más importantes y que tomaba la mayor influencia al realizar la generación final. Para analizar el proceso de generación en el LLM, se decidieron utilizar 3 frases de las frases, y revisar su influencia en el modelo.

Debido a la utilización de la librería de transformadores para utilizar esto, se tuvieron que utilizar los valores logit y la atención del transformador, lo que explica los valores pequeños de la *importance score*, pero en escala, indica la misma cantidad de importancia que se tiene en la generación final. Los valores logit y la atención basados en los mismos transformadores, indican que tanto se enfoco en esa frase, y la importancia que realiza.

En la Figura 6.5 en el análisis de LIME para la frase 'COMO, CLIMA HOY', mira el impacto de las palabras y como influencia los valores logit al tratar de predecir los *tokens* a usar. Un valor positivo se realizan en todas, indicando que todas las palabras fueron entiddidas y causaron mucha ayuda al realizar la oración, y permitieron realizar interpretaciones más precisas y certeras.

En la misma figura, indica que la palabra 'COMO', fue que causo un cambio significativo, indicando que el modelo entendío que era una pregunta. Siguiendo el tiempo con hoy, y finalmente indicando que se preguntaba, sobre el clima. En la tabla 6.3, están los resultados números del análisis LIME.

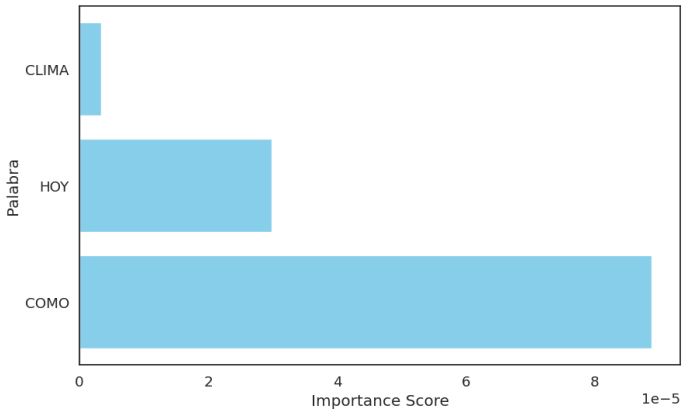

Figura 6.5: LIME del modelo para la oración 'COMO, CLIMA, HOY'

Frase Lensegua: 'COMO, CLIMA, HOY'	
Ínterpretación teoríca: '¿Comó está el clima hoy?'	
Palabra	Valores de Influencia
COMO	9.246e-05
HOY	3.395e-06
CLIMA	7.904e-07
Ínterpretación real: '¿Comó está el clima hoy?'	
Resultado BLEU	1

Tabla 6.3: Resultado LIME para la oración 'COMO, CLIMA, HOY'.

En la Figura 6.6 en el análisis de LIME para la frase 'AYER, VIERNES, LIMPIAR, CASA, TODO, DÍA', similarmente con la Figura anterior, todos los tokens son positivo, e indica mejor los tokens de día, todo y ayer, e tomando en cuenta mejor la frase de tiempo en este caso en específico, y dejando de último la acción de limpiar con un menos *importance score*. En la tabla 6.4, están los

valores del *importance score*

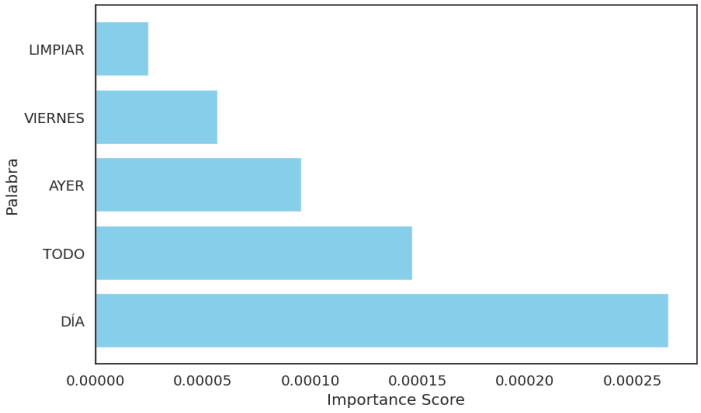

Figura 6.6: LIME del modelo para la oración 'AYER, VIERNES, LIMPIAR, CASA, TODO, DÍA'

Frase Lensegua: 'AYER, VIERNES, LIMPIAR, CASA, TODO, DÍA'	
Ínterpretación teoríca: 'AYER, VIERNES, LIMPIAR, CASA, TODO, DÍA'	
Palabra	Valores de Influencia
DÍA	2.666e-4
TODO	1.475e-4
AYER	9.583e-05
VIERNES	5.679e-05
LIMPIAR	2.4804e-05
Resultado LIME para la oración 'AYER, VIERNES, LIMPIAR, CASA, TODO, DÍA'	

Tabla 6.4: Resultado LIME para la oración 'AYER, VIERNES, LIMPIAR, CASA, TODO, DÍA'

Finalmente, en la Figura 6.7, para la frase 'MAR, GUSTAR, ÉL, MAR, GUSTAR, ELLA, NO'. En este caso, es una de las oraciones más largas, y también que tiene un token repetido. Y se muestra que este token no tiene tanta importacia, como el 'NO', que hace la diferencia entre las dos personas. Sin embargos, el modelo ya vió como confundió con la palabra 'ella' que indica que son diferentes entidades. Ya con varias entidades, el modelo tuvo problemas para poder entender que estaba sucediendo, o como ella lo condunfió. Se muestra los valores específicos en la tabla 6.5

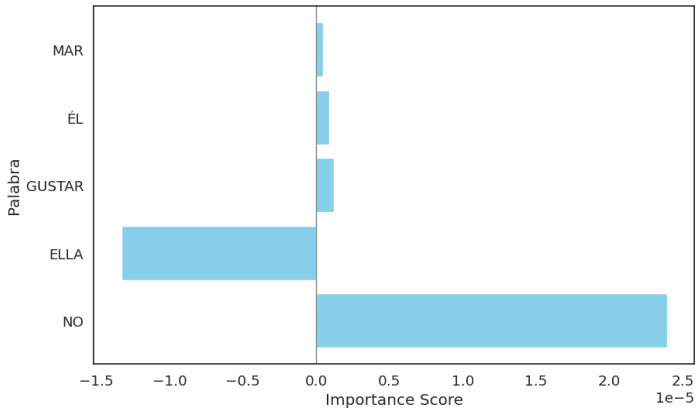

Figura 6.7: LIME del modelo para la oración 'MAR, GUSTAR, ÉL, MAR, GUSTAR, ELLA, NO'

Frase Lensegua: 'MAR, GUSTAR, ÉL, MAR, GUSTAR, ELLA, NO'	
Ínterpretación teoríca: 'A él le gusta el mar, pero a ella no	
Palabra	Valores de Influencia
NO	2.393e-05
ELLA	-1.329e-05
GUSTAR	1.181e-06
ÉL	8.746e-07
MAR	4.894e-08
Ínterpretación real: 'A él le gusta el mar, pero a ella no.	
Resultado BLEU	1

Tabla 6.5: Resultado LIME para la oración 'MAR, GUSTAR, ÉL, MAR, GUSTAR, ELLA, NO'

Al ver la calificación BLEU, de estas tres oraciones, todas tienen un resultado de al menos 0.8, indicando que el modelo a pesar de todo, pudo entender como funcionaba. Todas acertaron de manera correcta, inclusive con la confusión que se generó con la tercera oración.

Al ver las oraciones, y enteiendo el contexto, se puede ver como los valores de influencia con los logits afectan de una manera que se pueda entender los modelos de manera correcta.

6.2. Retroalimentación a expertos

Al final de todo el entrenamiento, se mostraron los resultados finales del *fine-tuning*, a una experta con concocimiento de LENSEGUA. Se mostraron las oraciones de cada uno de las traducciones finales, Esta entrevista esta includía en el Anexo B.

Al mostrar la información, la experta menciono varios comentarios con las salidas. Primero, menciona sobre ciertas situaciones, donde la Inteligencia Artifical, confunde entre una situación, si es una pregunta, o si es una declaración.

En la tabla 6.6, en la oración de la de querer comer pastel, la oración generada es una pregunta, mientras que la real era una delcración. Varias de estas suceden en el modelo, y ella recomienda agregar un signo de interrogación y otro símbolo para poder identificar si es una oración o es una pregunta. Según la experta, esto evitaría la confución que ocurren en las situaciones.

Otro dato que menciona en sobre ciertos token que aparecen. En el siguiente ejemplo, menciona sobre los tiempos.

En la misma tabla, la oración sobre las casas, la versión esperada era la versión en pretérito, mientras que la versión real esta en presenta. Hay que mencionar que hay que modificar la base de datos, para poder especificar en que tiempo esta, y poder generar respuestas correctas en el tiempo correcto.

También habian traducciones que no acertaban de ninguna manera. No podía realizar palabras como 'SACAPUNTAS' y utilizaba ciertas letras al tratar de adividar que utilizar. También habían oraciones a medias, y que faltaban, o no transmitían correctamente la información. A pesar de esto, la mayoría tenían una esencia miníma de lo que se quiere hablar.

El comentario más especfico y menciona en general, es que las traducciones estaban entendidas y se sabía lo que significaba. La mayor diferencia, es que mostrando los resultados reales y los esperados, es que ambas definiciones están bien, y que se podía entender, no eran afectadas. Resumiendo sus propias palabras, 'Están escritas en formas diferentes, pero al final, ambas dicen lo misma, solo que en manera diferentes.'

Al pedir una calificación de la escala del 1 a 10, la experta de una calificación de 9 con respecta a la traducción generales. Dio un comentario, explicando, sobre que el contenido respectivo, estaba de calidad, diciendo lo que era correcta con respecto a la lengua de señas.

Tabla 6.6: Comentarios de traducción

Oración Esperada	Oración Real	Comentario
'¿Ustedes comen pastel?'	'Ustedes comieron pastel	La oración en sí esta bien. Sin embargo, esta en forma de interrogación, lo que no es según los signos específicas. Se necesita especificar con signos de puntuación.
'El eligió entre dos casas.'	'El elige entre dos casas.'	Buy buena traducción. Se guarda la presencia de la información. Lo que sí, es que esta en pasado la oración tradución. En las frases, no hay ninga referencia de la acción esta en pasado o presente.
'A Juan no le gusta leer.'	'A Juan no le gusta leer.'	Una traducción bastante perfecta. Guarda la información y tiene sentido con respecto a las señas utilizadas.
'Pon la flor azul sobre la mesa.'	'Pon la flor azul en la mesa.'	La traducción esta muy buena y es de alta cálidad. Lo que hay que tener en mente es que en LENSEGUA, no existén los arítculos o prepocisiones, estos hay que inferirlos basados en las señas, y estas cambian basado en la interpretación de la persona.
'SACAGRAPAS'	'Sacagawea'	Esta traducción esta basante mal. Se tiene la idea, pero las letras no coinciden con lo que se quiere decir exactamente.
'La pizza de queso es mi favorita.'	'Pizzo es mi queso favorite'	Hay una idea general de lo que se quiera hablar, que es el queso favorito. Sin embargo, esta a media la oración, y a pesar que la esencia básica esta ahí, no es acceptable de la misma manera y hay que mejorar en la base de datos para poder mejorar el resultado final.

6.3. Comparación con OpenAI

Al tener el modelo final ya entrenado con los nuevos datos, se midió primero la métrica de BLEU para tomar en cuenta la calidad de los datos. En este caso, al aplicar el análisis de BLEU, dió el resultado siguiente de 0.632.

Entrenado con la nueva base de datos, primero se comparo entre los dos la distancia de Levenshtein y el largo de las oraciones también.

Tabla 6.7: Comparación de Levenshtein de los modelos

Modelo	Promedio de Largo de Oracion	Promedio de Distancia Levenshtein	Promedio de Porcentaje Diferencia
Modelo GPT-3.5	26.635	3.375	11.98 %
Modelo LLaMa 3	28.65	6.185	15.27 %

Se hicieron pruebas con LIME para analizar el funcionamiento del modelo LLaMa con la nueva base de datos. Basados en los resultados de las palabras que afectan de manera positiva el resultado en ambos modelos, el color naranja indica que ayudo al modelo a predecir, mientras que azul confundió al modelo a adivinar la oración correcta.

Figura 6.8: Resultado LIME para la oración .ºjalá hoy carro mucho noçon LLaMa

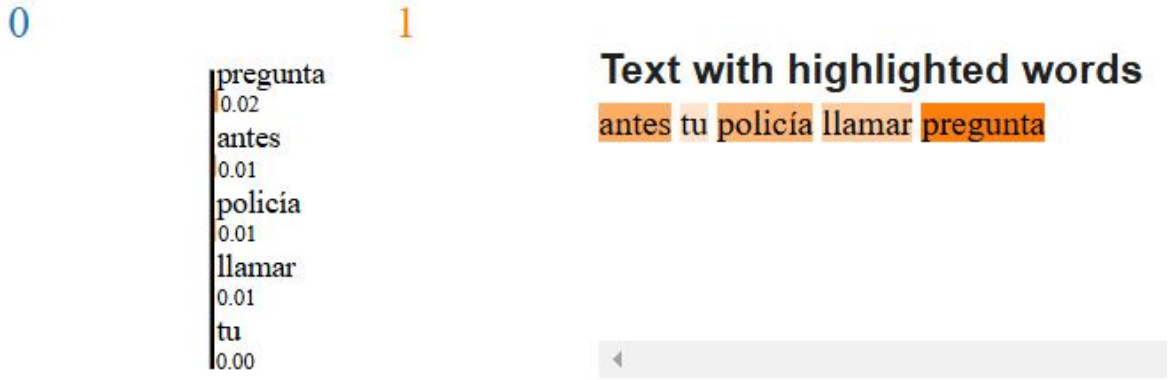

Figura 6.9: Resultado LIME para la oración .ªntes tu policía llamar preguntaçon LLaMa

Figura 6.10: Resultado LIME para la oración "pasado yo ir noçon LLaMa

CAPÍTULO 7

Discusión

La utilización de LLaMa para que fuera un interpreté de LENSEGUA funciona. La mayor ventaja que tiene LLaMa, a diferencia de algo más conocido, como GPT, es la mayot flexiblidad, de poder hacer *fine tuning* y poder crear un LLM más costumizable. Esta habilidad, hizo que sea posible y más sencillo de utilizar LoRA y el modelo Alpaca, basado en el mismo LLaMa, pero también que sea posible expandir de esto, y poder generar y mejorar en esta iteración de LLaMa. No solo Alpaca, pero otros modelos también, y también la posiblidad de poner diferentes prompts para poder realizar el mismo trabajo.

Y esta flexibilida ayudo que hiciera que el método LoRA fue de la manera más eficientes de poder entrenar un modelo de manera y eficiente de recursos, que al mismo tiempo, de resultados de alta calidad y que sean combatibles con lo que se necesita. La habilidad de poder solo modificar los pesos sin tener que entrenar el modelo desde cero hace que sea posible entrenar diferents modelos y decidir que funciona mejor, y también modificar los parametros para entrenar más modelos. También haciendo super flexible y eficaz para que funcione en un tema en especíco, en este caso, en caso de interpetre de LENSEGUA.

Como se mira a través de las diferentes gráficas y resultados de BLEU en las gráficas, se nota los cambios que fueron hechos en cada iteración para poder mejorar el modelo. En el primer modelo básico, se mira como entre más iteraciones que se realizan, hay un problema de overfitting, haciendo que el resultado de BLUE 0.125, indicando que el modelo era inefectivo, y solo entiendo a penas ciertas cosa. Un BLEU de esa cantidad indicaba que habían muchos errores, y no se entendía la esencia de la traducción. Esto indica que había que aplicar un *Early Stopping*, y evitar dejar entrenando el modelo por mucho tiempo, o llegaría más problemas con overfitting y perdiendose el significado.

Ya al aplicar un *Early-Stopping*, se pueden mirar mejores resultados, mejorando hasta 0.21. Y de ahí, se llego al alcance de este proyecto de al menos 0.2 de BLEU para que sea lo mínimo funcional para un traductor de un lenguage a otro según esta métrica.

Al querer mejorar el modelo, aplicando al estándar para la LoRA, utilizando un rango de 8 y un alpha de 16, indica que estos valores son apropiados para utilizar en la traducción de LENSEGUA. El rango de 8 es lo suficientemente bajo, para que se pueda enfocar en los temas de traducción, ya que rangos más áltos implica el tener que aprender temas que no conozca, o que tenga menos

información, siendo más útiles para traducción de lenguas en idiomas mayas, como Kak'chiquel. Pero al ser del idioma español, este permite que se más enfoca en español. Entre más paramétros necesito, más rango necesitaría.

El alpha de 16 hace que sea escalable también, y tenga los pesos correctos. Pueda saber cuales son los pesos para poder identificar que token usar y seguir, pero tampoco cambiar de manera drástica con tokens diferentes y únicos.[6]

Esto finalmente lleva al uso de *Supervised Fine-Tuning* que debido a que este enfoque permita al modelo ajustar sus parámetros de manera más precisa en función de un conjunto de datos específico y etiquetado. Teniendo un cuenta información apropiada y anterior, y no utilizar un modelo generico entre *fine-tuning*. Durante esta fase, el modelo no solo se beneficia de las representaciones generales aprendidas durante el preentrenamiento, sino que también se adapta a las particularidades del dominio en cuestión, optimizando su capacidad para generar traducciones más coherentes y contextualmente relevantes con la información dada.

Ya teniendo en cuenta sobre un set de párametros a utilizar, basado en la métrica, se pudo notar un cambio en la métrica de BLEU. Aquí, como se mira la efectividad del modelo, llegando como mínimo 0.263 de BLEU, e inclusive con un dataset más amplió para pruebas, llegando hasta 0.289 de BLEU, casi hasta 0.3 indicando un modelo de traducción comercialmente viable.

Es importante mencionar que la calidad del modelo no es parte del alcance del proyecto, lo más importante es que sea funcional y que pueda ser utilizado con la aplicación de Señas Chapinas.

Es importante tomar en cuenta que BLEU toma en cuenta principalmente el orden de las palabras, y ponde mucho pensos en ciertas palabras que se consideran sencillas. Es decir, que a pesar que de su alta calidad, hay ciertos casos donde penaliza una oración, a pesar que tenga un orden correcto.

En la Tabla 6.1, se puede mirar las limitaciones de BLEU. Al analizar los n-gramas, da al mismo valor todas las palabras, haciendo que ciertas palabras tengan más valor de lo que realmente tienen. En el ejemplo de de 'Yo la animo', se puede mirar que ambas oraciones tienen el mismo significado, y que a simple vista, se puede entender los mismo. Sin embargo, BLEU no toma en cuenta esto. Mira todas las palabras en la oración, y hace la mámetica para dar el resultado. Y este es lastimado por la falta de 'a' y 'ella' y al añadir, 'la'. Como dijo la experta, 'Esta bien traducido, pero esta diciendo lo mismo en forma diferente.'

Otro de los ejemplos en donde se ve más afectado, es en los tiempos verbales de las oraciones. En la misma tabla, esta la oración 'Miguel gano la carrera'. La oración esperada era, 'Miguel gana la carrera.' Aquí se mira sobre a pesar que todas las palabras están en orden, y entienda el sentido correcta, no esta compltamente correcto, debido que el resultado real es 'Miguel gana la carrera.' Esta diferente palabra es fuciente para que la calificación de BLEU sea de 0.0. Es una situación en donda se necesita especificar sobre cuales deben de ser los tiempos de los verbos, justo lo mismo que confirma la experta. Aun así, el contexto general de la oración y acción queda intacta.

Y tamnbién algo de los límites son ciertos tokens que aparecme, por ejemplo, si aparece un signo de puntuación como el punto o no en la traducción, o si hay un error de capitalización, y lo hace que a pesar que el contenido este correcto, no se vea reflejado por temas de los n-gramas.

Esto es para indicar, que BLEU es una buena métrica para analizar que la traducción tenga las mismas palbaras y tenga el contenido correcto, no es una métrica perfecta, ya que no toma en cuenta el orden, debido que hay varias formas de decir una oración, y todas son válidas para que sean entendibles y coherentes. Y también toma todos los tokens de igual importancia, y penaliza de una alta manera si falta un token y si una letra es diferente.

Esto no indica que BLEU no agarra errores. Ciertas situaciones, como palabras bastantes complicadas como 'SACAPUNTAS', y oraciones dejando a medias y faltando información correcta, indica que hay ciertas mejoras o el modelo de LLaMa o el LLM no puede agarrar la información de manera

correcta no importa lo que suceda. Este es el modelo de 7 mil millones de paramétros, y que el modelo de 60 mil millones de paramétros pueda capturar mejorar esta relación. Asimismo, a pesar que la oración no sea perfecta, si la esencia básica esta ahí, pero dice otra cosa diferente, BLEU da puntos por una mala traducción.

En el modelo, se puede mirar como los tokens afectan al modelo, y predice en que trabajar. Al utilizar LIME, mira el proceso de *importance score* utilizando logits. Debido que es un sitema de traducción de un idioma a otro, y debido a la complejidad del sistema de transformadores y *fine-tuning* con LoRA, hace que la atención y los *logits* sean más bajos. Sin embargo, estos valores se pueden seguir utilizando para analizar como funciona el modelo.

Y basado con los valores logits y LIME, mira como el modelo realiza las traducciones necesarias y de calidad para poder enteneder el contexto alrededor para dar traducciones correctas. Basado en los valores de logits y la atención respectiva del modelo de transformadores, toma en el contexto y se enfoca en lo más importante, y hace las traducciones especificas.

Esto mira como el procedimiento de *fine-tuning* hace un impacto positivo, y lleva por un camino correcto al proceso de generación de oraciones. En los ejemplos utilizados para mostrar los valores logits, indican como afectan los tokens para poder crear las oraciones, y mostrando su importancia. Y estos son los ejemplos con un BLEU de 1.0.

En las tres interpretaciones originales donde se miraba solamente el BLEU, hay una alta cantidad de información que permite entender, y que al menos dos personas, incluyendo una experta en LENSEGUA, pudiesen entender la oración y el contexto de la oración. Esto es suficiente para indicar que son buenas interpretaciones, debidoa una alta métrica de BLEU e interpertable por terceras partes expertas.

En conclusión, el análisís demuestra el *fine-tuning* y la utilización de LoRA para entrenar el modelo de una manera que . No es perfecta, debido que había situaciones que no eran completamentes correctas, más las limitaciones de BLEU afectando el rendimiento final. Pero las oraciones correctas, eran entendibles y coherentes, y mostraban de manera correcta la informacion que se quería transmitir. El modelo para la tarea siempre sería complicada, y no siempre daría resultados perfectos, pero al realizar esta tarea, dio resultados satisfactorios.

Finalmente, cuando se tomaron los dos modelos, y entrenarlos con el mismo dataset del modelo de OpenAI, basados en los resultados, hay una ligera mejora del modelo de OpenAI que con el modelo de LLaMa.

El modelo de LLaMa tiene una distancia de Levenshtein de 6.185, casi el doble que el modelo de OpenAI, de 3.375. Con sus distancias promedios, esto hace que el promedio de difernecia de LLaMa sea mucho mayor que el modelo de OpenAi por más de 3 %. Esto indica que el modelo GPT-3.5 tiene la ventaja cuando se trata de hacer las interpretaciones, y comete menos errores en generales. Y tambi

La gran diferencia entre estos dos modelos, son la facilidad de entender contexto y la complejidad, y eficiencia de recursos entre los dos. GPT-3.5 es el mejor modelo entre los dos, solo viendo a sus resultados. Al analizar ambos, se muesta la ventaja que se tiene GPT-3.5 con todos sus recursos que tiene para resultados más certeros. También al ver el análisis de LIME, se mira que hay una gran influencia en cada palabra con GPT-3.5 que con LLaMa.

Sin embargo, estos recursos son lo que más lo afecta, debido que GPT-3.5 necesita más tiempo para poder entrenar con una cantidad de datos. A pesar que se tuvo que mejorar la GPU para entrenar con LLaMa, LLaMa, no se necesita tantos recursos y su habilidad de ser open-source y entrenar con solo lo único que necesita hace que sea una versión viable a utilizar, si no se tienen los recursos correctos si ne necesita entrenar desde 0.

CAPÍTULO 8

Conclusiones

1. La observación de los resultados y la métrica de BLEU demuestra que es posible desarrollar un intérprete de IA capaz de generar oraciones a partir de traducciones de señas de LENSEGUA. Además, las LLM están capacitadas para traducir lenguas menos comunes, como la Lengua de Señas Guatemalteca, lo que sugiere que es viable establecer una conexión entre las oraciones y optimizar el proceso para lograr una interpretación más precisa.

2. Desarrollar una base de datos desde cero, junto con el modelo Alpaca permitió preprocesar los datos de forma correcta para poder usar el LLM como interprete. Este enfoque permitió que el LLM pudiera interpretar y utilizar la información de manera efectiva. La implementación de este sistema demuestra cómo la integración de modelos avanzados y bases de datos bien estructuradas puede mejorar significativamente la capacidad de procesamiento y comprensión de datos en aplicaciones de inteligencia artificial.

3. Se utilizó LLaMA 3.0 para desarrollar una herramienta capaz de integrar traducciones de LENSEGUA y generar oraciones gramaticalmente coherentes y comprensibles. Este avance demuestra la eficacia de los LLMs en la mejora de la calidad y precisión de las traducciones.

4. Se observaron resultados satisfactorias al validar el producto de los procedemientos de interpretación no solo por las metricas de BLEU, como contraparte tecnica, sino que también corroborando su validez con personas terceras externas al desarrollo tecnologico.

5. Se alojó la aplicación, y el modelo de procesamiento de interpretación, en un servidor en la nube para su facil acceso y tener caracteristicas remotas.

CAPÍTULO 9

Recomendaciones

1. Se dedicaron varios días completos para recopilar al menos 500 datos que pudieran dividirse entre el entrenamiento, la validación y las pruebas, con solo una persona trabajando en este proceso. Se podrían haber obtenido más datos si hubiera habido más personas colaborando en la recolección.

2. Es fundamental obtener más datos y transformarlos de manera que sean utilizables, lo que permitirá contar con una muestra más representativa. Esto mejorará el entrenamiento del modelo y le proporcionará más información para aprender, y que estos también tengán suficiente información para que puedan ser utilizables sin ambigüedad.

3. Encontrar una manera eficiente de poder realizar el entrenamiento en un mismo lugar, que pueda ser rápido, eficiente, escalable, y que den resultados de calidad sin perder tiempo o recursos.

Bibliografía

[1] Amazon Web Services: *What Is Transformers in Artificial Intelligence?*, 2023. https://aws.amazon.com/es/what-is/transformers-in-artificial-intelligence/, Accessed: 2024-11-02.

[2] Amazon Web Services, Inc.: *¿Qué es el procesamiento de lenguaje natural? - Explicación del procesamiento de lenguaje natural*, n.d. https://aws.amazon.com/es/what-is/nlp/.

[3] Arrieta, A. B., N. Díaz-Rodríguez, J. Del Ser, A. Bennetot, S. Tabik, A. Barbado, S. Garcia, S. Gil-Lopez, D. Molina, R. Benjamins, R. Chatila y F. Herrera: *Explainable Artificial Intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible AI.* Information Fusion, 58:82–115, 2019. https://doi.org/10.1016/j.inffus.2019.12.012.

[4] Banerjee, D., P. Singh, A. Avadhanam y S. Srivastava: *Benchmarking LLM powered Chatbots: Methods and Metrics.* arXiv, 2023. https://doi.org/10.48550/arxiv.2308.04624.

[5] Corporation, NVIDIA: *CUDA C++ Programming Guide (12.6)*, 2024. https://docs.nvidia.com/cuda/pdf/CUDA_C_Programming_Guide.pdf.

[6] Dettmers, Tim, Alberto Pagnoni, Ariana Holtzman y Luke Zettlemoyer: *QLoRA: Efficient Finetuning of Quantized LLMs (Version 1).* arXiv, 2023. https://doi.org/10.48550/ARXIV.2305.14314.

[7] Guatemala, Congreso de La Republica de: *Reglamento de la ley que reconoce y aprueba la lengua de señas de Guatemala - Lensegua*, 2021. Diario de Centro América, No.11 (7 de junio 2021).

[8] Jawade, B.: *Understanding LORA — low rank adaptation for finetuning large models.* Medium, December 2023. https://towardsdatascience.com/understanding-lora-low-rank-adaptation-for-finetuning-large-models-936bce1a07c6.

[9] LENSEGUA: *Ley que reconoce y aprueba la lengua de señas de Guatemala - Lensegua*, 2020. Diario de Centro América, No.11 (18 de febrero 2020).

[10] OpenAI: *Better Language Models*, 2019. https://openai.com/index/better-language-models/, 14 febrero.

[11] Papineni, K., Roukos S. Ward T. Zhu W.: *BLEU: a Method for Automatic Evaluation of Machine Translation.* Association for Computational Linguistics, 2002. https://doi.org/10.3115/1073083.1073135.

[12] togethercomputer: *RedPajama-Data.* https://github.com/togethercomputer/RedPajama-Data.

[13] Touvron, H., T. Lavril, G. Izacard, X. Martinet, M. Lachaux, T. Lacroix, B. Rozière, N. Goyal, E. Hambro, F. Azhar, A. Rodriguez, A. Joulin, E. Grave y G. Lample: *LLaMA: Open and Efficient Foundation Language Models*. arXiv, 2023. https://doi.org/10.48550/arxiv.2302.13971.

[14] Turner, R. E.: *An Introduction to Transformers*. arXiv, 2023. https://doi.org/10.48550/arxiv.2304.10557.

[15] Vaswani, A., N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser y I. Polosukhin: *Attention is all you need*. arXiv, 2017. https://arxiv.org/abs/1706.03762.

[16] Weller, O., M. Marone, V. Braverman, D. Lawrie y B. Van Durme: *Pretrained Models for Multilingual Federated Learning*. En *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics, 2022.

ANEXO A

Código

Repositorio de Github donde esta guardada la base de datos cual fue creada desde cero.

https://github.com/Roberto-VC/LLaMa-LENSEGUA

Google Colab en donde se trabajo el *Fine Tuning* de LLaMa para la interpretación de LENSEGUA.

https://colab.research.google.com/drive/1VbR9pUMkIEgJlFTuWFnjTiIqU9Pmf-HMscrollTo=61-vMAPh9UY

ANEXO B

Entrevista

Transcripción de la entrevista realizada con Lizbeth Lopez Casteñeda, y su opinión sobre las traducciones realizadas con el LLM.

Entrevista con Lizbeth Lopez Castañeda, conocimiento de LENSEGUA

Roberto: Hola, muy buenas noches.

Lizbeth Muy buenas noches. Espero que estes bien.

Roberto. Muchas gracias. Y espero que usted esté bien.

Lizbeth: Estoy bien gracias. Dime entonces, en que necesitas que te ayude.

R: Bueno, aquí le voy a enseñar. Aquí tengo mi proyecto. Lo que estoy haciendo es una inteligencia artificial, en donde utilizando LENSEGUA, traduce oraciones. Es para una aplicación que realiza traducciones de lengua de señas.

L: ¿Entonces estás haciendo la traducción de señas con todos los gestos?

R: Sí, pero no. Si es una traducción, pero lo que está haciendo, son las señas ya traducidas, pero ahora, lo que está haciendo, es con estas señas, formando oraciones basadas en estas señas.

L: Ah ya entiendo entonces.

R: Sí, y necesita su ayuda, para verificar la calidad de estas traducciones y ver qué es lo que usted piensa, y cree que son de calidad.

L: Muy bien. Está bueno.

R: Okay, aquí es lo que está pasando. ¿Qué piensa usted de estas traducciones?

L: Primero, lo que me doy cuenta es que las traducciones están tan bien hechas y me gustan como se miran la mayoría.

R: Muchas gracias, Aunque tengo una duda. Veamos esto, 'El eligió entre dos casas'. Esta es la

oración que salió, en presente., aunque yo estaba esperando que saliera, “El elige entre dos casas. Mi duda es como identificar que sucede y manejar tiempos.

L: Eso me dí cuenta. Lo que está sucediendo en LENSEGUA, es que al principio o final, hay que indicar el tiempo. Utiliza una seña para indicar que sucede. Solo indicando entre hoy y ayer, es suficiente para indicar tiempo, pero también hay otras maneras en que mejorarlo.

R: Entiendo. Muchas gracias. Otra cosa, aquí, se mira, la oración “¿Ustedes comen pastel?” y “Ustedes comieron pastel” que la traducción tiene sentido. Sin embargo, lo que sí son los signos de interrogación a la par.

L: También me di cuenta y tengo dudas en general sobre estos casos. Cómo identificar una pregunta.

R: Sí, es tratando de realizar que fueran en pasado o presente. Me puede explicar cómo podría mejorar esto.

L: Usualmente en LENSEGUA, si se quiere hablar una pregunta, se añade al final el signo de interrogación para evitar confusiones. Aquí esto se puede utilizar para hacer la diferencia entre pregunta o no.

R: Muchas gracias. Agradezco

L: Obviamente, no es completamente perfecto. Solo mire en la siguiente, para tratar de identificar “SACAGRAPAS”, salía, “Sacagawea”. Está la idea, pero la traducción está mal.

R: Si no lo atino en ciertas ocasiones.

L: Si jaja.

R: Pero sí, que piensa en general.

L: Pues todas me miran bien. Varias de las traducciones están muy buenas, como estas, “A Juan no le gusta leer,”, y “A Juan no le gusta leer.”, me gustan mucho.

R: Muchas gracias.

L: Y otras, en relación con la original, están bien, pero dicen lo mismo pero en forma diferente. Se tiene este ejemplo, ’Pon la flor azul sobre la mesa’, y ’Pon la flor azul en la mesa.’ Esto dice lo mismo pero de forma diferente. Debido a que en LENGUA no existen artículos ni preposiciones, hay que ser un poco creativo y flexible para indicar de lo que se está hablando. Finalmente, hay otras que se pudieran mejorar por ejemplo, “La pizza de queso es mi favorita.” y “Pizzo es mi queso favorite”, donde está la idea general, la pizza de queso siendo favorita pero también hay que poder mejorarla para que si se entiende que es pizza de queso, no solo otra cosa.

R:Si tuviera que dar una calificación de la escala del 1 al 10, qué calificación le daría.

L: Pues, yo le daría un 9. Sí, todo está bien. Como había mencionado, las traducciones son de alta calidad. Como había mencionado antes, varias están diciendo lo que quieren decir, pero están escritas de una manera diferente. Y aparte de ciertas que necesitan mejora, la mayoría están bien.

R: Muchísimas gracias. Se lo agradezco mucho. Muy amable, y que bueno que le hayan gustado.

L: Muchas gracias a usted por enseñarme esto también. Muy lindo e interesante quedo.