UNIVERSIDAD TÉCNICA DEL NORTE

FACULTAD DE INGENIERÍA EN CIENCIAS APLICADAS

CARRERA DE INGENIERÍA EN SOFTWARE

TEMA

ENTRENAMIENTO DE UN MOTOR DE TRADUCCIÓN AUTOMÁTICA INGLÉS-ESPAÑOL ESPECIALIZADO EN EL CAMPO DE LOS VIDEOJUEGOS MEDIANTE EL SOFTWARE MTUOC BASADO EN LA RED PROFUNDA TRANSFORMERS.

TRABAJO DE GRADO PREVIO A LA OBTENCIÓN DEL TÍTULO DE INGENIERO EN SOFTWARE

AUTOR: Daniel Alexander De La Torre Ponce

DIRECTOR: Ing. Fausto Alberto Salazar Fierro, MSc

Ibarra – Ecuador

2025

UNIVERSIDAD TÉCNICA DEL NORTE

BIBLIOTECA UNIVERSITARIA

AUTORIZACIÓN DE USO Y PUBLICACIÓN A FAVOR DE LA UNIVERSIDAD TÉCNICA DEL NORTE

1. IDENTIFICACIÓN DE LA OBRA

En cumplimiento del Art. 144 de la Ley de Educación Superior, hago la entrega del presente trabajo a la Universidad Técnica del Norte para que sea publicado en el Repositorio Digital Institucional, para lo cual pongo a disposición la siguiente información:

DATOS DE CONTACTO			
CÉDULA DE IDENTIDAD:	1003343280		
APELLIDOS Y NOMBRES:	De La Torre Ponce Daniel Alexander		
DIRECCIÓN:	Ibarra – El Sagrario		
EMAIL:	dadelatorrep@utn.edu.ec		
TELÉFONO FIJO:		**TELÉFONO MÓVIL:**	0963613539

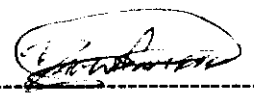
DATOS DE LA OBRA	
TÍTULO:	ENTRENAMIENTO DE UN MOTOR DE TRADUCCIÓN AUTOMÁTICA INGLÉS-ESPAÑOL ESPECIALIZADO EN EL CAMPO DE LOS VIDEOJUEGOS MEDIANTE EL SOFTWARE MTUOC BASADO EN LA RED PROFUNDA TRANSFORMERS.
AUTOR (ES):	DE LA TORRE PONCE DANIEL ALEXANDER
FECHA: DD/MM/AAAA	01/09/2025
PROGRAMA:	■ **PREGRADO** ☐ **POSGRADO**
TITULO POR EL QUE OPTA:	INGENIERO DE SOFTWARE
DIRECTOR:	ING. Fausto Alberto Salazar Fierro, MSC.
ASESOR:	ING. Carpio Agapito Pineda Manosalvas, MSC.

2. CONSTANCIAS

El autor (es) manifiesta (n) que la obra objeto de la presente autorización es original y se la desarrolló, sin violar derechos de autor de terceros, por lo tanto, la obra es original y que es (son) el (los) titular (es) de los derechos patrimoniales, por lo que asume (n) la responsabilidad sobre el contenido de la misma y saldrá (n) en defensa de la Universidad en caso de reclamación por parte de terceros.

Ibarra, a los 1 días del mes de septiembre de 2025.

EL AUTOR:



Nombre: Daniel Alexander De La Torre Ponce

CI: 1003343280

CERTIFICACIÓN DIRECTOR

Ibarra 01 de septiembre del 2025

CERTIFICACIÓN DIRECTOR DEL TRABAJO DE TITULACIÓN

Por medio del presente yo Ing. Fausto Salazar, MSc., certifico que el Sr. Daniel Alexander De La Torre Ponce portador de la cedula de ciudadanía número 1003343280, ha trabajado en el desarrollo del proyecto de grado "**Entrenamiento de un motor de traducción automática inglés-español especializado en el campo de los videojuegos mediante el software MTUOC basado en la red profunda Transformers.**", previo a la obtención del Título de Ingeniero en Software realizado con interés profesional y responsabilidad que certifico con honor de verdad.

Es todo en cuanto puedo certificar a la verdad

Atentamente

1002172631
FAUSTO ALBERTO
SALAZAR FIERRO

Digitally signed by 1002172631
FAUSTO ALBERTO SALAZAR FIERRO
Date: 2025.09.01 15:55:32 -05'00'

Ing. Fausto Alberto Salazar Fierro, MSc.

DIRECTOR DE TRABAJO DE GRADO

Dedicatoria

Dedico este trabajo a mis padres, Luis y Lucía, por estar a mi lado en cada etapa, con su apoyo incondicional, comprensión y palabras de aliento que me impulsaron a seguir adelante.

A mi familia en general, por su guía, conocimientos y constante disposición para ayudarme a alcanzar esta meta.

A las creadoras de contenido de Hololive Productions y Phase Connect, cuya creatividad y dedicación fueron una fuente de inspiración y alivio emocional en los momentos de estrés y duda de este proceso académico.

Gracias a todos por ser parte de este logro.

AGRADECIMIENTO

A la Universidad Técnica del Norte y a sus docentes, por brindarme una formación académica integral que fortaleció mis habilidades, amplió mi visión profesional y enriqueció mi pensamiento crítico.

Al MSc. Fausto Salazar y al MSc. Carpio Pineda, por su valioso tiempo, conocimientos y guía durante el desarrollo y la culminación de este trabajo de titulación.

A todos mis profesores, por su compromiso, dedicación y por haber contribuido de manera significativa a mi crecimiento profesional y personal

RESUMEN

Este trabajo presenta el entrenamiento y la evaluación de un motor de traducción automática neuronal (MT) especializado en el par inglés–español, enfocado en el ámbito de los videojuegos. Para su desarrollo, se utilizaron los scripts del proyecto MTUOC, que permiten gestionar corpus paralelos y entrenar modelos basados en arquitecturas Transformer, como MarianNMT.

Para ello, se construyó un corpus paralelo de aproximadamente 50.000 pares de segmentos, compuesto principalmente por diálogos extraídos de novelas visuales y adaptaciones de traducciones no oficiales (fantranslations). Este corpus fue alineado manualmente y utilizado para realizar un proceso de fine-tuning sobre un modelo base preentrenado del proyecto OPUS-MT para el par inglés-español.

Se construyó un corpus paralelo de aproximadamente 50.000 pares de segmentos, compuesto principalmente por diálogos extraídos de novelas visuales y adaptaciones de traducciones no oficiales (fantranslations). Este corpus fue alineado manualmente y empleado para realizar un proceso de fine-tuning sobre un modelo base preentrenado del proyecto OPUS-MT para el par inglés–español.

El modelo fue entrenado durante 500 pasos en un entorno Docker optimizado con soporte para GPU (RTX 3050 Mobile, 4 GB VRAM), acompañado de un entorno de trabajo con 16 GB de RAM y 6 hilos asignados. La evaluación se llevó a cabo utilizando la métrica BLEU mediante SacreBLEU, y los resultados fueron comparados con motores de traducción general como Google Translate y Yandex Translate.

El sistema especializado alcanzó una puntuación BLEU de 53.2, superando significativamente a Google (45.1) y Yandex (44.8). Las principales mejoras se observaron en la traducción de expresiones coloquiales juveniles, la conjugación verbal adecuada al contexto y la correcta interpretación de jerga específica del entorno gamer.

Este estudio demuestra el potencial de aplicar técnicas de ajuste fino (fine-tuning) sobre modelos preentrenados para desarrollar traductores automáticos adaptados a dominios específicos. Su aplicación puede beneficiar a traductores independientes, equipos de localización reducidos o estudios indie de videojuegos, proporcionando una herramienta de apoyo que mejora tanto la eficiencia como la calidad en procesos de traducción especializados.

Palabras clave:

Traducción automática neuronal, videojuegos, fine-tuning, corpus paralelo, MarianNMT.

ABSTRACT

This work presents the training and evaluation of a neural machine translation (MT) engine specialized in the English–Spanish language pair, with a focus on the domain of video games. For its development, the MTUOC project scripts were used, which facilitate the management of parallel corpora and the training of models based on Transformer architectures, such as MarianNMT.

A parallel corpus of approximately 50,000 segment pairs was created, primarily composed of dialogues extracted from visual novels and adapted from unofficial translations (fantranslations). This corpus was manually aligned and used to perform fine-tuning on a pre-trained base model from the OPUS-MT project for the English–Spanish pair.

The model was trained for 500 steps in an optimized Docker environment with GPU support (RTX 3050 Mobile, 4 GB VRAM), along with a working setup of 16 GB of RAM and 6 allocated threads. The evaluation was conducted using the BLEU metric via SacreBLEU, and the results were compared against general-purpose translation engines such as Google Translate and Yandex Translate.

The specialized system achieved a BLEU score of 53.2, significantly outperforming Google (45.1) and Yandex (44.8). The most notable improvements were observed in the translation of youth-oriented colloquial expressions, contextually accurate verb conjugation, and the correct interpretation of domain-specific gaming jargon.

This study demonstrates the potential of applying fine-tuning techniques on pre-trained models to develop domain-adapted machine translation systems. Its application may benefit independent translators, small localization teams, or indie game studios by providing a support tool that enhances both efficiency and translation quality in specialized workflows.

Keywords: Neural machine translation, video games, fine-tuning, parallel corpus, MarianNMT.

TABLA DE CONTENIDO

LISTA DE FIGURAS

LISTA DE TABLAS

INTRODUCCIÓN

Problema

El sector de los videojuegos ha experimentado un crecimiento significativo desde 2017, generando $5.000 millones en Latinoamérica. Sin embargo, este solo representa el 3,6% de los ingresos globales [1]. Los desarrolladores independientes se han visto afectados al no tener la capacidad de contratar los servicios de empresas de traducción profesionales. Su falta de contratación repercute de manera negativa, ya que limita considerablemente su alcance y posibles beneficios, como se muestra en la figura 1. Teniendo en cuenta lo que señala Cristian González [2], un gran número de videojuegos obtienen hasta el 50% de sus ingresos de los mercados internacionales. Por ejemplo, la empresa japonesa Nintendo, cuyo informe anual indicaba que el 80% de sus ventas en 2008 correspondieron al mercado internacional [3].

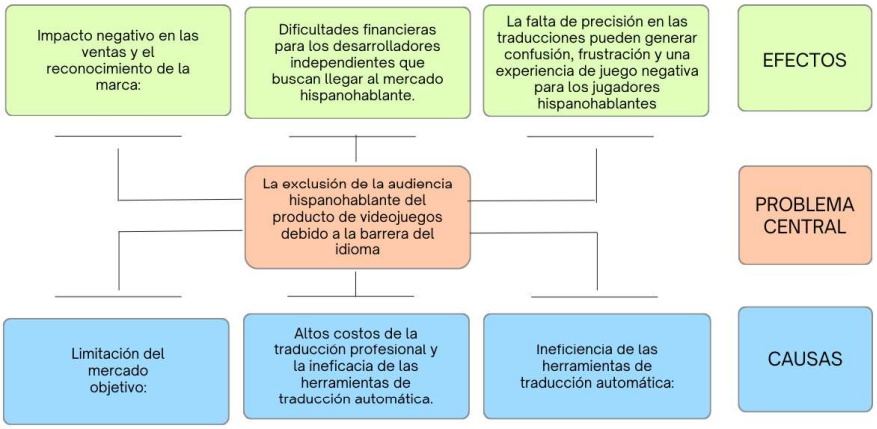


Figura 1. Árbol de problemas.

Objetivos

Objetivo general

Entrenar y evaluar un motor de traducción automática (neuronal) inglés-español especializado en el campo de los videojuegos mediante el software MTUOC basado en la red profunda Transformers

Objetivos específicos

- Redactar un marco conceptual sobre: los motores de traducción

- automática, los corpus bilingües, arquitectura Transformers y la metodología Desing and Create.
- Crear un corpus bilingüe especializado en el campo de los videojuegos.
- Entrenar un motor de traducción automática (neuronal) a través del software MTUOC (Machine Translation) de la UOC (Universitat Oberta de Catalunya) [4].
- Comparar los resultados del motor entrenado con los resultados ofrecidos por traductores automáticos de libre acceso, usando la métrica BLEU (Bilingual Evaluation Understudy) [5].

Alcance

El objetivo de este proyecto consiste en entrenar y evaluar un motor de traducción automática basado en redes Transformers para la traducción del inglés al español en el ámbito de los videojuegos. Para lograrlo, se creará un corpus especializado en este campo, el cual será construido mediante sus diferentes etapas, como se indica en la Figura 2.

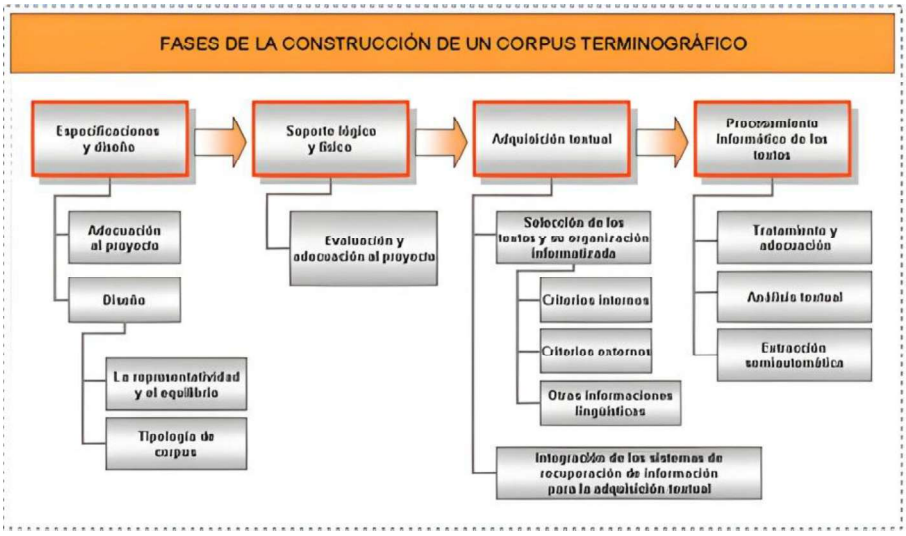


Figura 2. Fases de la creación de un corpus [6]

La elección de utilizar el software MTUOC se debe a su versatilidad, accesibilidad y las funcionalidades que ofrece para el entrenamiento de motores de traducción automática. Para llevar a cabo el entrenamiento del motor, se empleará MarianNMT, basado en las redes Transformers y ejecutado en un entorno Linux. La evaluación y comparación del modelo entrenado se realizarán mediante el uso de la métrica BLEU.

Dado que el enfoque principal de este trabajo se centra en el entrenamiento del motor utilizando MTUOC, no se implementará una metodología de desarrollo ágil o tradicional. No obstante, se empleará el enfoque de Desing and Create para asegurar un desarrollo correcto y eficiente del proyecto.

Justificación

Esta investigación se alinea con el Objetivo 8 de los ODS (Objetivos de Desarrollo Sostenible), que busca fomentar el trabajo decente y el crecimiento económico [7]. También está alineada con la Política 1.1 del Objetivo 1 del Plan de Creación de Oportunidades 2021-2024, la cual busca fomentar la creación de nuevas oportunidades laborales en condiciones dignas [8].

Justificación Tecnológica: - Dentro de la legislación ecuatoriana, la Política 2.5 del Plan de Creación de Oportunidades busca promover la investigación y el desarrollo de nuevas tecnologías para mejorar la adaptación tecnológica en la industria [8].

Justificación Económica: - La Política 4.2 del Plan de Creación de Oportunidades busca aumentar la apertura comercial con mercados exteriores y potenciar la industria local [8], así como apoyarse en la cultura local. [9]

Desde esta perspectiva, los desarrolladores son los principales beneficiarios del proyecto, mientras que la industria de los videojuegos representa a los beneficiarios indirectos.

I. MARCO TEÓRICO

A. Videojuegos

1) Historia de los videojuegos

Los videojuegos nacen como una adaptación de las experiencias de los juegos a los nuevos medios tecnológicos. Para adentrarnos en lo que significa un videojuego, primero debemos entender las palabras que lo forman. Según Caillois, los juegos pueden definirse como actividades libres, casuales, aisladas en el tiempo y el espacio, aleatorias e infructuosas, regidas por reglas arbitrarias [10].

Frasca define el término videojuego como cualquier forma de software de entretenimiento que utiliza una plataforma electrónica, la cual puede ser un computador o una consola, e involucra a uno o más jugadores en un entorno físico o en red [11].

Este concepto nos sirve para identificar lo que podemos denominar el primer videojuego: *Noughts and Crosses*, una adaptación a computadoras del clásico juego del tres en raya, el cual fue desarrollado por Alexander S. Douglas durante 1952, del cual se puede evidenciar su interfaz en la figura 3.

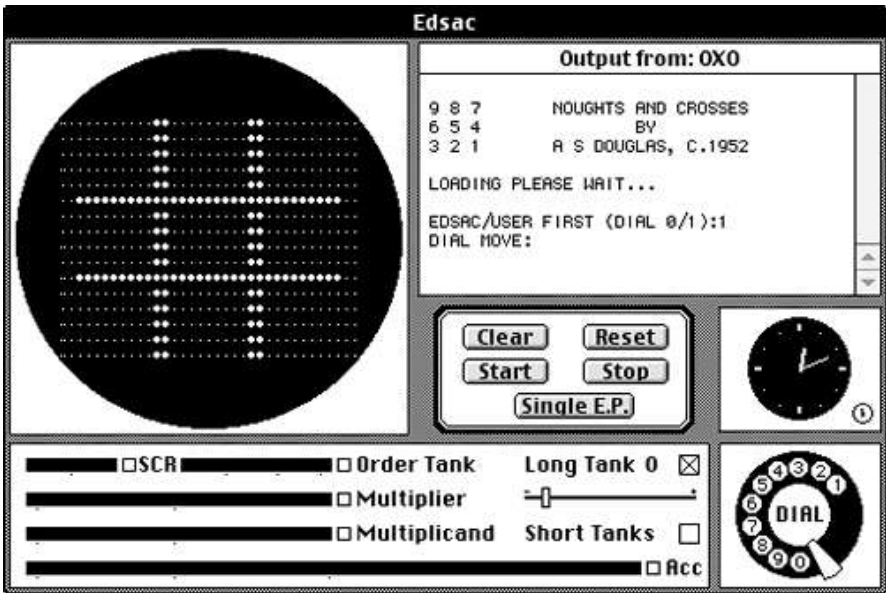


Figura 3. Nought and crosses (OXO) [12]

Desde ahí, en su evolución, como se ve en la figura 4, podemos destacar algunos puntos remarcables. En 1972, tuvimos el nacimiento de la Magnavox Odyssey, acompañada del juego de Pong como la primera videoconsola de la historia. Además, hubo un auge de las recreativas, con Space Invaders como uno de sus

principales referentes. Posteriormente, en 1983, llegaron las videoconsolas como la Nintendo Entertainment System (NES) y su separación de los ordenadores como el Spectrum. La llegada del gigante japonés permitió revitalizar la industria de las videoconsolas, hecho que se vería reflejado en los años 90 con la diversidad de consolas lanzadas, como Master System, Mega Drive o la misma SNES.

En 1993, se lanzó la PlayStation 1, lo que llevó a la estandarización de los videojuegos en 3D. Durante la sexta generación de videoconsolas, en el año 2000, se estableció el uso del CD como estándar, así como el surgimiento de Xbox como uno de los tres grandes que se mantienen hasta la actualidad. En la séptima generación, en el año 2006, tuvimos el auge del juego casual en consolas como Wii y en portátiles como PSP o Nintendo DS, que lamentablemente acabaría con la llegada de los smartphones y la emulación.

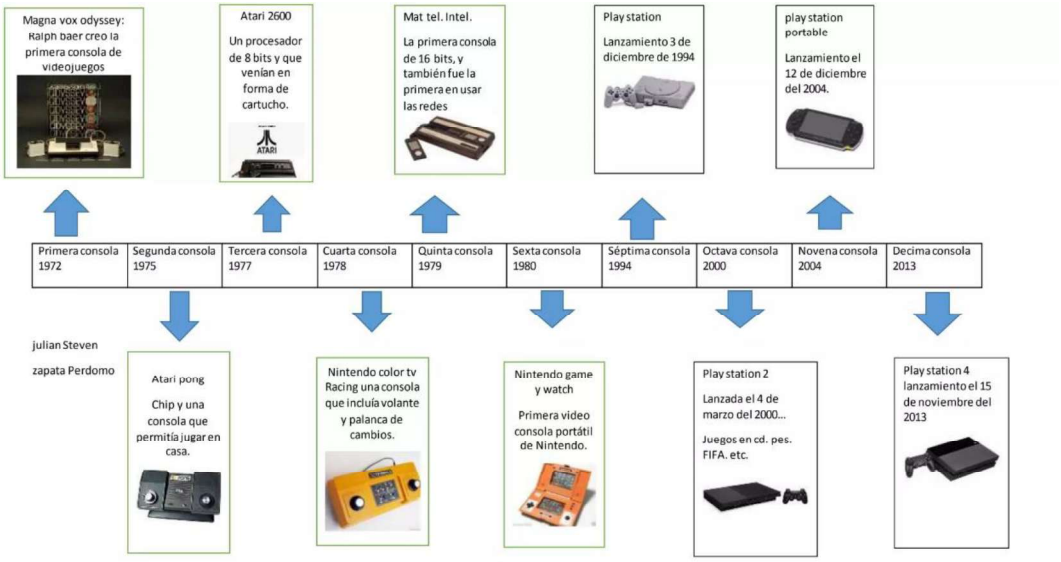


Figura 4. Línea de tiempo de los videojuegos

2) Géneros y características de los videojuegos

Las etiquetas de género son un sistema que nos ha permitido categorizar a los videojuegos según los elementos que presentan. Dicho de otro modo, estas etiquetas nos permiten hacernos una idea de lo que vamos a encontrar dentro de un videojuego a nivel mecánico. El concepto de una mecánica dentro de un videojuego, como explica Rouse, puede definirse como las diferentes posibilidades de jugabilidad que ofrece el videojuego al usuario. Esto se puede observar en la capacidad de reacción del mundo a la interacción del jugador dentro de él [13].

En el ámbito de los videojuegos, podemos encontrarnos con una enorme diversidad en cuanto a los géneros se refiere, pues el medio se ha ido adaptando según las necesidades y hábitos de consumo de sus clientes. Al igual que en el cine, esto ha generado una explotación, diversificación e hibridación de los géneros, como se puede observar en la Tabla 1.

Por ejemplo, dentro de los juegos de la saga Persona, podemos observar una hibridación de géneros en sus segmentos jugables. Durante el apartado social, tenemos los elementos propios de la novela visual, como un sistema de relaciones donde el jugador puede potenciar la afinidad con diversos personajes mediante eventos. Esta mecánica social, a su vez, se refleja en las mejoras de características de los personajes, parte propia del rol, que repercute en el combate con bonos de afinidad o la posibilidad de ataques especiales conjuntos.

TABLA I
GÉNEROS DE VIDEOJUEGOS

Género	Descripción	Videojuegos
Beat them up	Juego de lucha donde los jugadores usan combos y ataques especiales para atacar a varios enemigos.	TMNT: Shredder's Revenge, Kunio-kun, Final Fight, Double Dragon, Streets of Rage, etc.
Lucha	Juego en el que dos o más jugadores luchan entre sí utilizando diferentes habilidades y movimientos.	Super Smash Bros, Mortal Kombat, Soulcalibur, Street Fighter, Tekken, etc.
Juegos de acción en primera persona	Juego en el que el jugador ve el mundo desde una perspectiva humana y controla los movimientos y acciones de los humanos a lo largo del tiempo.	Counterstrike, Call of Duty, Far Cry 6, Half-Life, Overwatch, etc.
Acción en tercera persona	Un juego donde el jugador observa al personaje desde una perspectiva en tercera persona y controla los movimientos y acciones del personaje en tiempo real.	Uncharted, Grand Theft Auto, Red Dead Redemption, Gears of War, God of War, etc.
Infiltración	Es un juego donde los jugadores deben completar objetivos sin ser detectados.	Assassin's Creed, Metal Gear Solid, Splinter Cell, Dishonored, Hitman, etc.
Plataformas	Es un juego que requiere que los jugadores naveguen por los niveles saltando, corriendo y trepando, usando diversas habilidades y movimientos.	Donkey Kong Country, Super Mario Bros, Rayman, Celeste, Crash Bandicoot, etc.
Arcade	Juegos cortos y sencillos que se basan en la repetición y la puntuación.	Tetris, Pac-Man, Donkey Kong, Space Invaders, Pinball, etc.
Sport	Un juego que simula deportes reales, los jugadores pueden competir en una variedad de disciplinas.	PES, NBA 2K, FIFA, MLB The Show, Madden, etc.
Carreras	Es un juego sobre la simulación de conducción en el que los jugadores se enfrentan entre ellos o en retos como contrarreloj.	Crash Tag Team Racing, Need for Speed, Forza Horizon, Mario Kart, Gran Turismo, etc.
Agilidad mental	Un juego que desafía la velocidad mental, la memoria y las habilidades de resolución de problemas del jugador.	Brain Training, Puyo Puyo, Tetris, Brain Age, Dr. Mario, etc.
Educación	Juegos diseñados para enseñar o reforzar conceptos educativos de forma divertida e interactiva.	Carmen Sandiego, The Oregon Trail, Math Blaster, 321 Penguins, etc
Aventura clásica	Un juego donde los jugadores exploran un mundo abierto resuelve acertijos y toman decisiones basadas en la historia.	The Legend of Zelda, Metroid, Castlevania, Dark Souls, Hollow Knight, etc.
Aventura gráfica	Un juego en el que el jugador utiliza una interfaz gráfica para interactuar con el entorno y los personajes mediante comandos o acciones.	Day of the Tentacle, Life is Strange, The Walking Dead: Season One, Grim Fandango, etc.
Musicales	Es un juego en el que los jugadores trabajan juntos en una variedad de formas centradas en la música.	Guitar Hero [14] , Rock Band, Dance Dance Revolution, Just Dance, Osu!, FUSER, etc.

Party Games	Un juego diseñado para jugar en grupo para fomentar la interacción social y la diversión juntos.	Mario Party, Super Smash Bros., Overcooked 2, Gang Beasts, Among Us, Fall Guys: Ultimate Knockout, It Takes Two, etc.
Juegos On-Line	Un juego que se puede jugar en Internet y puedes interactuar con otros jugadores de todo el mundo.	World of Warcraft, League of Legends, Dota 2, Counterstrike: Global Offensive, Fortnite, Minecraft, Rocket League, Genshin Impact, etc
Rol	Un juego, donde el jugador controla las acciones del personaje, como moverse, luchar, interactuar con objetos y otros personajes, y tomar decisiones que afectan el desarrollo de la historia.	Final Fantasy, Dragon Quest, Fallout, Mass Effect, The Last Story, etc.
Novela Visual	Un juego donde la interacción del jugador es mediante decisiones las cuales permiten avanzar la historia, la cual suele estar ramificada.	Katawa Shoujo, Clannad, Danganropa, Aokana - Four Rhythms Across the Blue, etc.

Nota: Esta tabla presenta una selección representativa de géneros y títulos de videojuegos. No pretende ser una lista exhaustiva.

3) Importancia de la localización en los videojuegos

Para comprender la importancia de la localización, primero debemos explicar a qué hace referencia. De acuerdo con Crespo, la localización es el proceso por el cual se realiza la adaptación de un contenido o producto de origen a uno de destino para su venta. Con ello, podemos entender que la localización no se trata de una traducción literal del contenido de un idioma a otro, sino que debe pasar por un filtro para poder adaptarse de la cultura de origen a las convenciones de la cultura de destino [15].

En el campo de los videojuegos, al ser un medio audiovisual, la localización debe pasar por un arduo proceso para adaptarse de un idioma a otro, intentando siempre mantener el significado original mientras se es consistente con la ambientación de su mundo. Esto se puede observar en carteles, dialectos o metáforas propias de la zona.

La localización de los videojuegos a otros mercados ha sido un factor clave desde las tempranas etapas de la industria de los videojuegos. Como señala Cristian González, un gran número de videojuegos obtienen hasta el 50% de sus ingresos de los mercados internacionales. Esto se evidencia en una de las compañías más grandes e importantes en la industria como lo es Nintendo, que en el reporte de ingresos de 2008 ya mostraba que aproximadamente un 80% de sus ingresos se debían al mercado exterior [2].

Con ello, se puede destacar la adaptación de videojuegos como uno de los pilares dentro de las compañías desarrolladoras, ya que estas trabajan en un mercado internacional, lo cual supone tener que adaptar sus

productos a varios idiomas e idiosincrasias según la zona, con el objetivo de maximizar su alcance y número de ventas.

4) Desafíos específicos de la traducción de videojuegos

La traducción presenta una serie de dificultades, pues el proceso no se trata de una traducción directa del contenido. Dentro de la traducción existe un proceso de localización y adaptación.

Los desafíos principales son:

TABLA II
PROBLEMAS DE LA TRADUCCIÓN

Problema	Descripción
Problemas de transferencia de significado	Los artículos de los revisores deben reproducirse de la misma manera que el artículo original y no deben omitir elementos del artículo original.
Problemas de contenido	Los revisores deben asegurarse de que el texto refleje su lógica interna. En otras palabras, los argumentos deben ser coherentes con el texto original, no deben contener implicaciones ilógicas o contradictorias y deben seguir una secuencia lógica.
Problemas de lengua y estilo	La singularidad del videojuego original debería reflejarse en la traducción. Como resultado, elementos como las teclas de voz, los registros y el uso de palabras especiales o voces en off deberían proporcionar una experiencia de juego consistente.
Formato y maquetación	Hay muchos problemas con los videojuegos, incluido el diseño del contenido en la pantalla, los errores tipográficos y la estructura del texto.

Nota: Adaptado de [2].

Esto lo podemos observar a más detalle en franquicias como la saga Yakuza de origen japonés. Esta tiene un departamento dedicado a su localización al mercado americano, donde al adaptar afrontan estos problemas al pasar de una cultura como la japonesa con un sistema de caracteres distintos, así como las diferencias idiomáticas y culturales propias del país de origen [16].

B. Traducción automática

1) Historia y evolución de la traducción automática

La traducción automática puede definirse como un tipo de traducción que utiliza recursos computacionales para procesar la conversión de un idioma de origen a uno de destino. La necesidad de sistemas de traducción ha existido desde la antigüedad debido a la gran diversidad de idiomas en el mundo. Los orígenes de las máquinas de traducción pueden rastrearse hasta el siglo XVII, cuando Descartes y Leibniz formularon teorías sobre la creación de diccionarios universales basados en las matemáticas.

Sin embargo, la idea de traducción de una lengua a otra se puede trazar hasta 1933 con la aparición de Artsouni y su invento de un programa que podía localizar el equivalente de una palabra de un idioma a otro. Al mismo tiempo, Troyanskii desarrolló las bases de lo que posteriormente sería la traducción automática basada en tres fases: simplificar la entrada en el idioma de origen, procesar de un idioma a otro y finalmente reconstruir la sentencia de forma que tenga sentido. No obstante, en su concepción, Troyanskii contemplaba que las fases 1 y 3 fueran realizadas por humanos [17].

Para localizar en el tiempo el primer traductor enteramente computacional, debemos remontarnos a los programas que usaban las máquinas para descifrar los mensajes durante la Segunda Guerra Mundial. Gracias a estos, en 1968 surgió un sistema de traducción inglés-ruso por parte de Peter Toma, el cual, sin embargo, fue descartado en su momento por su alto costo y baja eficiencia.

Desde aquel momento, los motores de traducción automática se basaron en procesos estadísticos para la reconstrucción de los textos de un idioma a otro. Se desarrollaron y utilizaron corpus bilingües donde se almacenaban textos de un par de idiomas con su correspondiente traducción, siendo conocidos como motores de traducción estadística por la base de su funcionamiento.

Los motores de traducción neuronal surgieron a partir del 2010, gracias al desarrollo de la inteligencia artificial, lo cual permitió una alternativa que aprovechaba esta nueva tecnología como base, adaptándose mejor a los contextos luego de los entrenamientos.

2) Tipos de traducción automática

a) Traducción automática estadística

La traducción automática estadística (TAE), conocida en inglés como Statistical Machine Translation (SMT), es un avance tecnológico cuyo propósito es traducir un texto de un idioma de origen a uno de destino a través de la computación. Este tipo de traducción se comprende como un conjunto de oraciones en la lengua de origen, cada una de las cuales está asociada a una lista de traducciones en la lengua de destino y a las probabilidades correspondientes [18] .

Este tipo traducción tiene dos etapas: el entrenamiento del motor de traducción y la decodificación, durante la cual se realizará la traducción de los textos de un idioma a otro. La fase del entrenamiento es el proceso mediante el cual se extrae un modelo de traducción estadística basado en un corpus paralelo,

además de un modelo lingüístico estadístico del mismo grupo lingüístico. Durante el entrenamiento, y en base al corpus, se asigna un porcentaje de probabilidad a cada palabra dentro del texto a traducir, basado en los textos almacenados. De esta manera, se determinará cuál sería la traducción más probable, como se puede observar en la fórmula de la figura 5 [17].

$$Best - translation\, T = argmax_t \;\; fluency\,(T) faithfulness\,(T,S)$$

Figura 5. Ecuación de la Fórmula del SMT

En la fase de decodificación, se realiza una separación del texto en oraciones, las cuales son posteriormente procesadas a nivel de palabras con el motor entrenado. Durante el procesamiento de las oraciones, se realiza una comparación estadística de la frase de origen, buscando las posibles traducciones de cada palabra o conjunto de palabras. Con ello, se obtienen las traducciones más probables y aproximadas, como puede observarse en la figura 6.

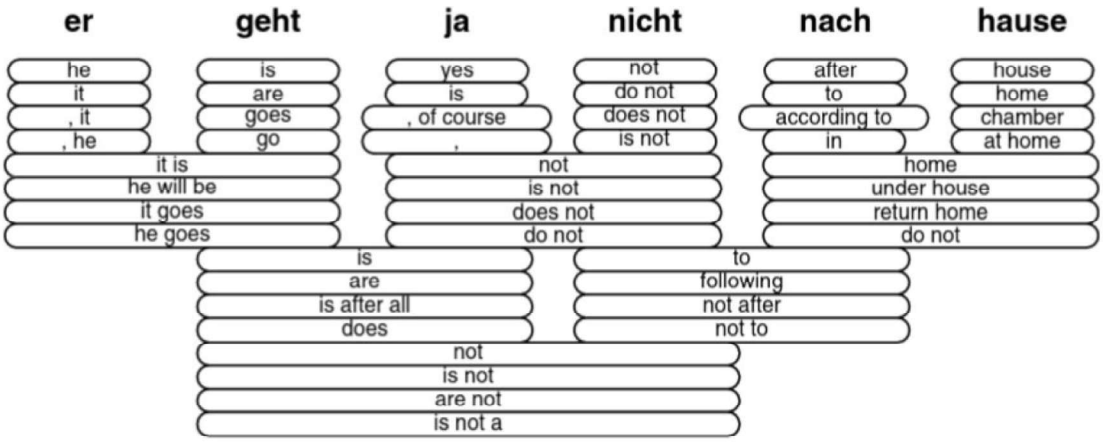


Figura 6. Ejemplo del funcionamiento dentro de un SMT [19]

b) Traducción automática neuronal

La traducción automática neuronal (TAN), conocida en inglés como Neural Machine Translation (NMT), es un tipo de tecnología cuyo propósito es traducir un texto de un idioma de origen a uno de destino a través de redes neuronales. El funcionamiento de este tipo de traducción toma un texto y lo divide en varias oraciones, las cuales procesa a través de la red neuronal para traducirlo de un idioma a otro [20].

El proceso comienza con la codificación de una frase como un vector de longitud fija. Posteriormente, el codificador produce una traducción a partir del vector de codificación. Este sistema de codificación-decodificación incluye un codificador y un decodificador de un par de idiomas, los cuales se entrenan de manera conjunta para aumentar la probabilidad de que la traducción sea correcta [17].

C. Redes neuronales y transformers

1) Introducción a las redes neuronales

Una red neuronal artificial (RNA) es un sistema de procesamiento computacional de información o señales el cual está basado en las neuronas humanas, esta consta de una serie de elementos de procesamiento simples conectados mediante conexiones que trabajan juntos para realizar un procesamiento distribuido paralelo y completar una tarea requerida. Las RNA se aplican en diversos campos como pueden la visión por computadora encarga del procesamiento de imágenes, el reconocimiento de voz y el análisis de datos, debido a su capacidad para identificar patrones complejos y de carácter no lineales en masivos volúmenes de datos [17].

Existen diferentes tipos de redes neuronales, entre ellas:

TABLA III
TIPOS DE REDES NEURONALES

Tipo de Red	**Descripción**
Perceptrón Multicapa (MLP)	Una red neuronal artificial con una arquitectura en varias capas como una capa de entrada, una o más capas ocultas y una capa de salida.
Redes Neuronales Convolucionales (CNN)	Redes diseñadas para el procesamiento de imágenes y video, que utilizan convoluciones para extraer características espaciales de los datos.
Redes Neuronales Recurrentes (RNN)	Redes especializadas en el procesamiento de secuencias temporales, como series de tiempo y lenguaje natural, capaces de mantener información a lo largo de múltiples pasos de procesamiento.
Redes Neuronales Generativas Adversariales (GAN)	Un tipo de red neuronal compuesta por dos redes enfrentadas: un generador y un discriminador.

Nota: Adaptado de [21].

2) Arquitectura de transformers

El Transformer es un modelo de red neuronal que usa una arquitectura basada en mecanismos de atención, permitiendo un enfoque global y capas de codificación de puntos tanto para el codificador como para el decodificador, como se muestra en las mitades izquierda y derecha de la Figura 7.

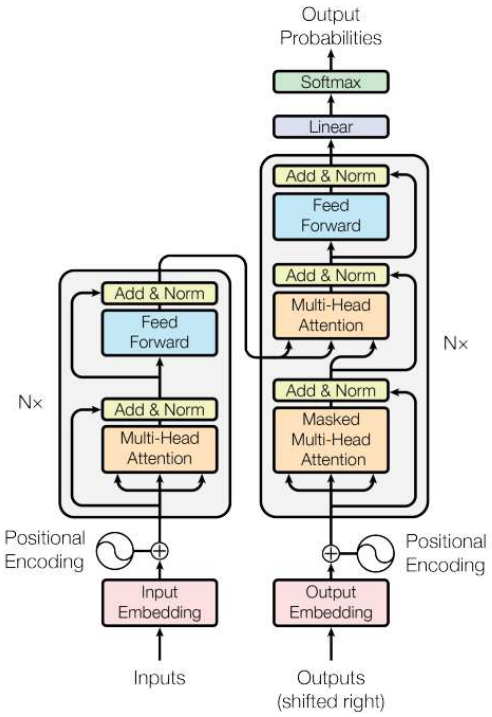


Figura 7. Arquitectura Transformers [22]

Como explica Brown, las redes Transformers se basan en el mecanismo de atención, que permite a la red enfocarse en las partes más relevantes de una secuencia de entrada al procesarla. Esto las hace especialmente adecuadas para tareas de procesamiento del lenguaje natural, donde la comprensión del contexto y las relaciones de largo alcance entre palabras es crucial [23].

Los Transformers introdujeron varias innovaciones clave [22] :

- Mecanismo de Atención: Permite a la red evaluar la relevancia de diferentes palabras en una secuencia de entrada al procesarla, enfocándose en las partes más importantes y permitiendo un manejo más eficiente de las dependencias a largo plazo.
- Paralelismo: A diferencia de las RNN, que procesan secuencias de forma secuencial, los Transformers permiten el procesamiento paralelo de los datos, lo que resulta en una mayor eficiencia y velocidad.
- Capas de Codificación y Decodificación: Los Transformers utilizan una pila de capas de codificación para transformar las entradas en representaciones intermedias y una pila de capas de decodificación para transformar estas representaciones en salidas

D. Software MTUOC

1) Descripción del software MTUOC

El software MTUOC es una herramienta desarrollada en un entorno Linux que nos asiste en diversas áreas. Estas incluyen la creación de motores mediante el reentrenamiento en contextos específicos para mejorar la precisión y eficiencia de los motores base, la evaluación de motores de traducción automática, y la integración con herramientas de traducción profesional [4].

2) Comparación con otras herramientas de traducción automática

En comparación con otras herramientas de traducción automática, MTUOC ofrece una mayor precisión en la traducción de términos técnicos y una capacidad superior para aprender y adaptarse a las necesidades específicas del usuario, superando a herramientas genéricas en contextos especializados[4].

3) Aplicaciones específicas en la traducción de videojuegos

Actualmente, en la traducción de videojuegos se utilizan principalmente herramientas de traducción asistida por computadora (TAC) o, en inglés, Computer Assisted Translation (CAT), ya que las herramientas de traducción automática (MT) aún no son lo suficientemente fiables para este propósito. La precisión y el contexto cultural son esenciales en la localización de videojuegos, y las herramientas de traducción automática actuales no siempre garantizan la calidad necesaria para comprender los contextos o la jerga específica del medio.

E. Metodología Design and Create

1) Descripción de la metodología Design and Create

La metodología Design and Create es un enfoque estructurado para el desarrollo y evaluación de proyectos de nuevas tecnologías, centrándose en la iteración y la mejora continua a través de ciclos de diseño, prueba y revisión, con el objetivo de que los proyectos sean funcionales y eficientes para los usuarios.

Esta metodología cuenta con cinco fases, las cuales son:

TABLA IV
FASES DE LA METODOLOGÍA DESIGN AND CREATE

Fase	Objetivo	Procedimiento
Requerimientos y Especificaciones	Definir los problemas que el nuevo artefacto debe resolver y establecer objetivos claros.	Documentar detalladamente los requerimientos funcionales y no funcionales del artefacto.

Diseño del Artefacto:	Crear un diseño conceptual del artefacto, estableciendo su estructura general y principales componentes.	Elaborar diseños detallados de cada componente del artefacto, asegurando que todas las partes trabajen de manera cohesiva.
Construcción:	Construir el artefacto de acuerdo con los diseños detallados, utilizando herramientas y técnicas apropiadas de ingeniería.	Ensamblar las diferentes partes del artefacto, asegurándose de que funcionen conjuntamente sin problemas.
Evaluación	Probar el artefacto para asegurarse de que cumple con los requerimientos especificados y funciona correctamente en diferentes escenarios.	Analizar los resultados de las pruebas y realizar las mejoras necesarias para optimizar el artefacto.
Implementación	Implementar el artefacto en su entorno de destino.	Supervisar el funcionamiento del artefacto y realizar mantenimiento para corregir posibles fallos y mejorar su rendimiento a lo largo del tiempo.

Nota: Adaptado de [24].

2) Aplicación de la metodología en el proyecto

En el contexto del proyecto, dado que el uso de un motor de traducción automática neuronal enfocado en el campo de los videojuegos está en etapas tempranas, la metodología Design and Create se adapta perfectamente. Al aplicarla, podremos desarrollar, evaluar y mejorar continuamente el motor de traducción, asegurando que cumpla con los estándares de calidad y precisión requeridos.

F. Evaluación de Motores de Traducción Automática

1) Métodos de evaluación de traducción automática

Dentro de la industria de la traducción, siempre ha existido un gran énfasis en la calidad del trabajo. Por ello, las traducciones pasan por un proceso de revisión en diferentes etapas. Procesos como el control de calidad aseguran la localización efectiva del producto a través de diversas técnicas, como la post-edición por profesionales del medio.

Existen diferentes metodologías, como la tradautomaticidad, que es ejecutada por profesionales del medio. Estos profesionales pueden evaluar la calidad de una traducción automática, valorando más

aquellas traducciones que parezcan más humanas. Esta evaluación se realiza según una serie de parámetros, como precisión, legibilidad o estilo, entre otros [17].

2) Métrica BLEU

La métrica de Bilingual Evaluation Understudy (BLEU), en español Subestudio de Evaluación Bilingüe, es una alternativa que busca realizar evaluaciones basadas en algoritmos para evaluar la correcta traducción generada por un motor de traducción automática. Surge ante los elevados costos en la evaluación de traducciones automáticas, las cuales debían ser ejecutadas por profesionales del medio.

La métrica BLEU es una medida de precisión realizada a nivel de n-gramas, unidades lingüísticas indivisibles. Emplea una precisión modificada que tiene en cuenta el número máximo de apariciones de cada n-grama en la traducción de referencia y aplica una penalización por brevedad que se añade al cálculo de la medida, como puede verse en la figura 8 [25].

$$\text{BLEU} = \text{BP} \cdot \exp\left(\sum_{n=1}^{N} w_n \log p_n\right).$$

Figura 8. Formula de BLEU [5]

G. Trabajos relacionados

[17] propuso la creación de motores de traducción automática (estadística y neuronal) especializados en el campo de la aviación utilizando la herramienta MTUOC. Se desarrollaron y entrenaron motores de traducción automática estadística y neuronal con datos específicos del campo de la aviación. El desempeño se evaluó en términos de calidad de traducción, y los motores especializados mostraron una mejora significativa en comparación con motores genéricos, proporcionando traducciones más precisas y relevantes para el dominio de la aviación. Sin embargo, el estudio no exploró la integración de estos motores en entornos de uso real ni la evaluación continua del rendimiento con nuevos datos de aviación. Además, no se comparó exhaustivamente con otros motores de traducción automáticos existentes.

[25] realizó una revisión sistemática de los sistemas de traducción automática y su evaluación de calidad. Se recopilaron y analizaron estudios previos sobre sistemas de traducción automática, centrándose en métodos de evaluación de calidad. Se clasificaron y compararon diferentes enfoques y métricas de evaluación utilizados en la literatura. La revisión destacó la evolución de los sistemas de

traducción automática, desde métodos estadísticos hasta redes neuronales profundas. Se identificaron métricas de evaluación clave y se discutieron las fortalezas y debilidades de cada enfoque. La revisión se basó en estudios previos y puede estar limitada por la calidad y el alcance de esos estudios. No se realizó una evaluación práctica de los sistemas de traducción ni se propusieron nuevos métodos de evaluación.

[26] creó un corpus paralelo inglés-español para entrenar y evaluar un sistema de traducción automática neuronal en el contexto del proyecto MTUOC. Se recopilaron textos literarios en inglés y sus traducciones al español, formando un corpus paralelo. Este corpus se utilizó para entrenar un sistema de traducción automática neuronal, que luego se evaluó en términos de precisión y fluidez. El sistema neuronal entrenado con el corpus paralelo mostró una mejora notable en la traducción de textos literarios en comparación con sistemas genéricos, capturando mejor el estilo y el contexto literario. El corpus se centró únicamente en textos literarios y puede no ser representativo de otros tipos de textos. Además, el estudio no evaluó el rendimiento del sistema en traducciones en tiempo real o en otros dominios.

[27] exploraron la posibilidad de utilizar traducción automática en la localización de videojuegos. Analizaron diferentes técnicas de traducción automática y su aplicabilidad a la localización de videojuegos. Realizaron pruebas con motores de traducción automática existentes para evaluar su desempeño en la traducción de contenido de videojuegos. Los resultados mostraron que, aunque la traducción automática puede ser útil para traducir contenido de videojuegos, hay desafíos significativos en términos de precisión y coherencia debido a la naturaleza específica y creativa del lenguaje en los videojuegos. El estudio se centró en pruebas preliminares y no incluyó una implementación completa en un proyecto de localización de videojuegos. Tampoco se abordaron aspectos importantes como la adaptación cultural y la revisión humana de las traducciones.

[28] investigó la traducción de metáforas en subtitulación oficial y automática, así como la evaluación de la traducción automática en este contexto. Se compararon traducciones de metáforas en subtítulos oficiales y generados automáticamente. Se evaluaron las traducciones en términos de precisión y adecuación, y se analizaron las diferencias entre los métodos de traducción. Las traducciones automáticas de metáforas mostraron una menor precisión y adecuación en comparación con las traducciones oficiales, destacando las limitaciones actuales de la traducción automática en el manejo de lenguaje figurado. El estudio se limitó a un conjunto específico de metáforas y subtítulos y no evaluó un rango más amplio de lenguaje figurado u otros tipos de contenido audiovisual. Además, no se consideraron mejoras potenciales mediante la intervención humana en el proceso de traducción.

II. DESARROLLO

Descripción de la metodología

En este trabajo se adopta un enfoque de estudio de caso único, tal como lo propone Yin [29] . La elección de un solo caso permite realizar un análisis detallado de las variables experimentales y de la configuración del entorno, lo cual resulta especialmente útil para:

- Evaluar cómo el entrenamiento con un corpus especializado influye en la calidad de la traducción.
- Determinar por qué un motor de dominio específico puede superar a sistemas generalistas en terminología y estilo narrativo propios de los videojuegos.

La Figura 9 muestra el flujo general de la metodología de estudio de caso aplicada, que abarca desde la definición de la unidad de análisis y las proposiciones, pasando por la recolección y el análisis de datos, hasta la interpretación de resultados y la discusión de la validez y la confiabilidad.

Figura 9. Fases del estudio de caso aplicado al entrenamiento de un motor NMT

A. *Planificación*

En esta fase inicial, y de acuerdo con el enfoque de estudio de caso único de Yin [30], se establecen los primeros pasos y se delimita el alcance del estudio, como de puede observar en la figura 10.

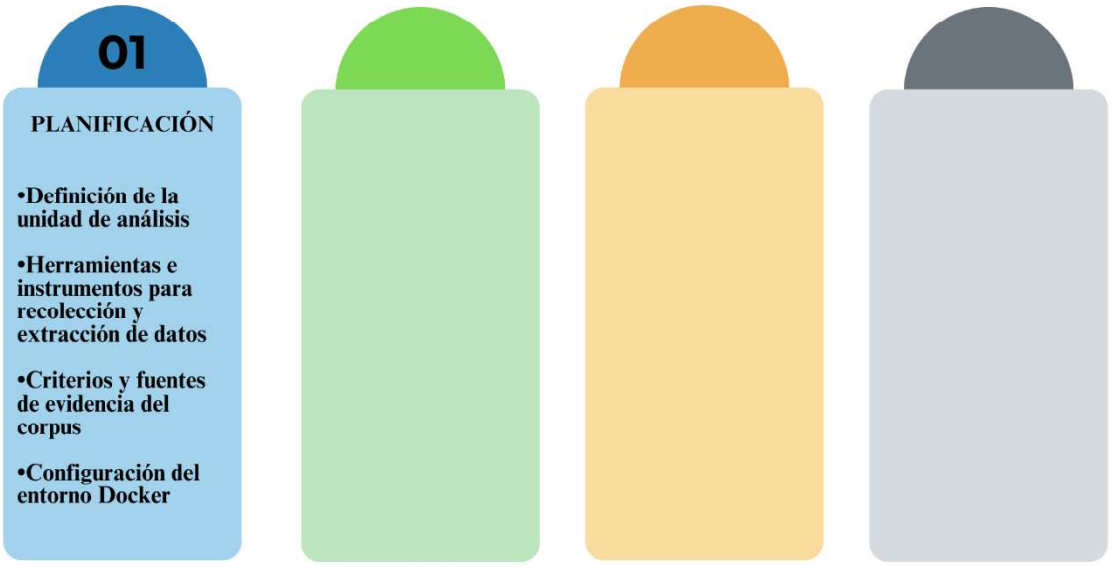


Figura 10. Fase de la planificación aplicado al entrenamiento de un motor NMT

1) *Definición de la unidad de análisis*

La unidad de análisis de este caso es el proceso de "entrenamiento y evaluación de un motor de traducción automática inglés-español especializado en el campo de los videojuegos", desde la extracción de diálogos hasta la generación de métricas de calidad.

2) *Herramientas e instrumentos para recolección y extracción de datos*

Se prepararon las siguientes herramientas e instrumentos técnicos:

TABLA V
LISTADO DE HERRAMIENTAS

Herramienta/Instrumento	Función	Detalles
Contenedor Docker	Entorno reproducible para entrenamiento con GPU	Ubuntu 20.04 + CUDA 12.1; incluye MTUOC y dependencias
MarianNMT	Backend de entrenamiento de redes neuronales	Invocado por MTUOC para preprocesar, entrenar y evaluar
AssetRipper	Extracción de assets de juegos Unity	Exporta archivos internos (.bs5); complemento de dnSpy
dnSpy	Instrumentación dinámica de DLLs para extraer diálogos	Inyección de rutinas de logging; genera archivos .h5
Scripts de Python	Limpieza, normalización y alineación de corpus	Conversión a TMX y formato MTUOC

3) *Criterios y fuentes de evidencia del corpus*

Se definieron criterios para garantizar la calidad y relevancia de los datos:

- Corpus paralelo: diálogos de novelas visuales con versiones en inglés y español, con al menos 50 000 oraciones alineadas y cobertura temática variada (escolar, terminología específica).
- Archivos de extracción: ficheros generados por AssetRipper y dnSpy (.bs5, .h5) que contienen el texto bruto.
- Scripts y resultados intermedios: archivos TMX, vocabularios, modelos parciales y checkpoints del entrenamiento, útiles para auditoría y repetibilidad.

4) *Configuración del entorno Docker*

Para asegurar un entorno reproducible y optimizado para el entrenamiento con GPU, se creó una imagen Docker personalizada a partir de la imagen oficial nvidia/cuda:12.1.0-cudnn8-devel-ubuntu20.04. Los pasos principales fueron:

- Imagen base: Se parte de nvidia/cuda:12.1.0-cudnn8-devel-ubuntu20.04 para disponer de CUDA 12.1 y cuDNN 8 sobre Ubuntu 20.04, lo que asegura compatibilidad con GPU NVIDIA y optimización para Marian-NMT.

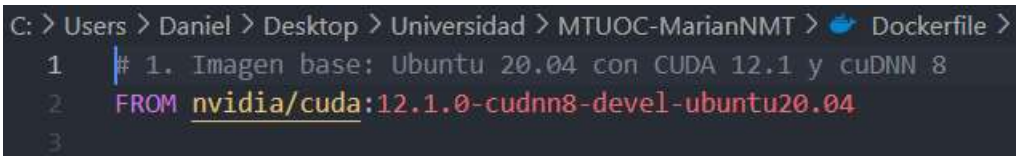


Figura 11. Uso de la imagen base de Ubuntu 20.04 y Cuda 12.1 en el dockerfile

- Variables de entorno: Se fijan LANG y LC_ALL a UTF-8, se desactiva la interacción de APT (DEBIAN_FRONTEND=noninteractive) y se añade el entorno virtual al PATH.

Figura 12. Definición de variables en el dockerfile.

- Dependencias del sistema: Se instalan herramientas esenciales (compiladores, CMake, Boost, Protobuf, etc.) y Python 3.8 con sus headers y venv. Tras la instalación, se limpia la caché de APT.

```
# 3. Instalación de dependencias del sistema
RUN apt-get update && apt-get install -y --no-install-recommends \
        build-essential \
        cmake \
        ca-certificates \
        curl \
        git \
        wget \
        libboost-dev \
        libboost-system-dev \
        libboost-filesystem-dev \
        libboost-program-options-dev \
        libboost-thread-dev \
        libboost-chrono-dev \
        libprotobuf-dev \
        protobuf-compiler \
        libssl-dev \
        libgoogle-perftools-dev \
        python3.8 \
        python3.8-dev \
        python3.8-venv \
        python3-pip \
        pkg-config \
        software-properties-common \
        vim \
    && rm -rf /var/lib/apt/lists/*
```

Figura 13. Instalación de dependencias en el dockerfile.

- Configuración de Python y creación de env
    - Se configura Python 3.8 como alternativa por defecto.
    - Se crea un entorno virtual en /opt/env y se actualizan pip, setuptools y wheel.

```
# 4. Configuración de Python
RUN update-alternatives --install /usr/bin/python python /usr/bin/python3.8 1 \
    && update-alternatives --install /usr/bin/python3 python3 /usr/bin/python3.8 1 \
    && pip3 install --no-cache-dir --upgrade pip setuptools wheel

# 5. Creación del entorno virtual
RUN python -m venv /opt/venv \
    && /opt/venv/bin/pip install --no-cache-dir --upgrade pip setuptools wheel
```

Figura 14. Configuración de Python en el dockerfile.

- Directorios de trabajo: Se establece /workspace como directorio donde residirán el código de MTUOC y Marian.

Figura 15. Configuración de directorios en el dockerfile.

- Instalación de MTUOC-old: Se clona el repositorio MTUOC-old y, dentro del entorno virtual, se instalan sus dependencias definidas en requirements.txt.

```
# 7. Clonación e instalación de MTUOC-old
RUN git clone https://github.com/aoliverg/MTUOC-old.git mtuoc-old \
    && /opt/venv/bin/pip install --no-cache-dir -r mtuoc-old/MTUOC/requirements.txt
```

Figura 16. Instalación de MTUOC en el dockerfile

- Clonación y compilación de Marian-NMT
    - Se clona el repositorio oficial de Marian para obtener la versión más reciente.
    - Se actualizan submódulos.
    - En build/, se invoca CMake con flags para habilitar SentencePiece, CPU y CUDA, y se ignoran las advertencias deprecadas (-Wno-error=deprecated-declarations) para evitar interrupciones en el build.
    - Se compila con make -j2 e instala en /usr/local.

```
# 8. Clonación del repositorio oficial de Marian y submódulos
RUN git clone https://github.com/marian-nmt/marian.git marian \
    && cd marian \
    && git submodule update --init --recursive

# 9. Compilación e instalación de Marian
WORKDIR /workspace/marian/build
RUN cmake .. \
      -DCMAKE_BUILD_TYPE=Release \
      -DUSE_SENTENCEPIECE=ON \
      -DCOMPILE_CPU=on \
      -DCOMPILE_CUDA=on \
      -DCMAKE_INSTALL_PREFIX=/usr/local \
      -DTHRUST_IGNORE_DEPRECATED_CPP_DIALECT=ON \
      -DCMAKE_CXX_STANDARD=14 \
      -DCMAKE_CXX_FLAGS="-Wno-error=deprecated-declarations" \
      -DCMAKE_CUDA_FLAGS="-Xcompiler=-Wno-deprecated-declarations" && \
    make -j2 && \
    make install
```

Figura 17. Instalación de MarianNMT en el dockerfile

- Entrypoint: El script docker-entrypoint.sh (que carga ~/.bashrc para activar el venv y variables como MTUOC_HOME) se copia, normaliza finales de línea y se marca ejecutable. Al iniciar el contenedor, este entrypoint asegura que al invocar bash el entorno virtual y las rutas queden correctamente configuradas.

```
# 10. Copiar entrypoint y convertir finales de línea
COPY docker-entrypoint.sh /usr/local/bin/docker-entrypoint.sh
RUN sed -i 's/\r$//' /usr/local/bin/docker-entrypoint.sh \
 && chmod +x /usr/local/bin/docker-entrypoint.sh

# 11. Ajustar PATH y entrypoint
ENV PATH="/opt/venv/bin:/usr/local/bin:${PATH}"
ENTRYPOINT ["/usr/local/bin/docker-entrypoint.sh"]
CMD ["bash"]
```

Figura 18. Configuración del punto de entrada en el dockerfile

B. *Recolección de datos*

Esta sección describe el flujo completo del procesamiento del corpus y del entrenamiento del motor de traducción automática, organizado en dos bloques: primero, la extracción y transformación de

los textos fuente, y posteriormente, la preparación y entrenamiento del modelo de traducción neuronal, como puede observarse en la figura 19.

Figura 19. Fase de la recolección de datos aplicado al entrenamiento de un motor NMT

1) *Extracción y preprocesamiento del corpus*

La recolección de datos se centró en textos provenientes de novelas visuales. Para acceder al contenido textual embebido en los archivos del juego, se emplearon herramientas especializadas de análisis estático y dinámico que permitieron inspeccionar y extraer los datos contenidos en los paquetes internos del motor Unity, como se ilustra en la Figura 20.

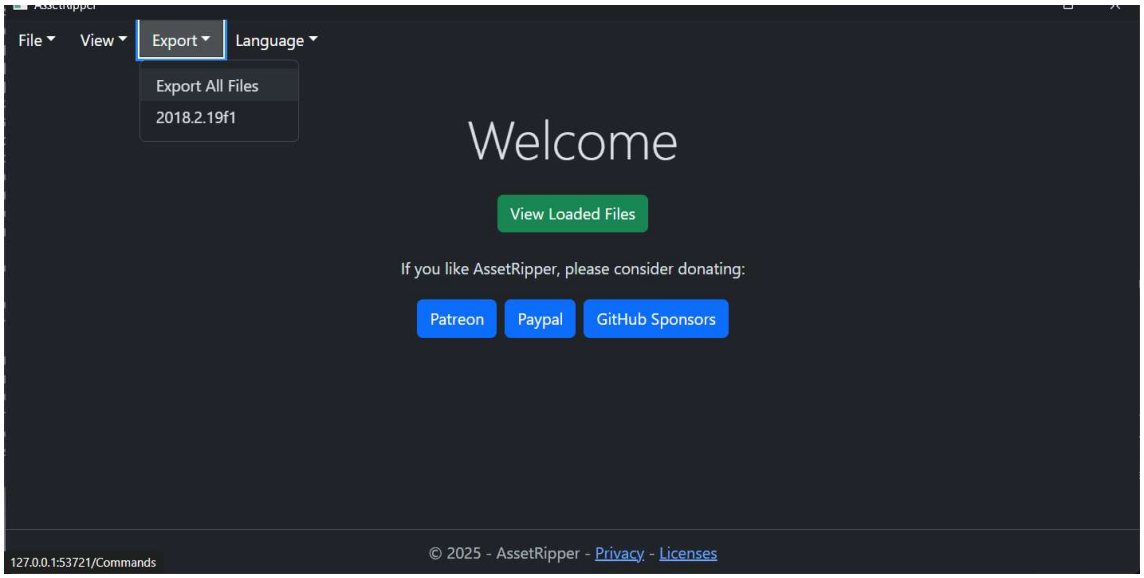


Figura 20. Extracción de assets desde el motor del juego

A través del análisis de la lógica del juego, se identificó el método responsable de interpretar los archivos que contienen los diálogos. Mediante una instrumentación controlada del código, se modificó este proceso para redirigir los textos procesados en tiempo de ejecución hacia archivos de texto plano, preservando así su forma original, como puede observarse en la Figura 21.

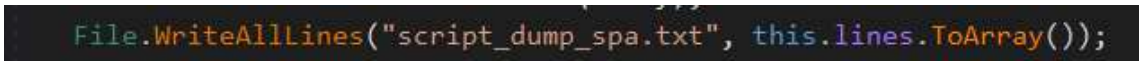


Figura 21. Instrumentación del código del juego para extraer textos

Una vez extraídos los textos, se aplicó un proceso de limpieza y transformación mediante scripts desarrollados en Python que pueden observarse en los anexos. En primer lugar, se eliminaron líneas que no correspondían a diálogos (como comandos o metadatos del motor del juego). Posteriormente, se separaron los textos por idioma, generando archivos monolingües. Finalmente, se filtraron los nombres de personajes o identificadores de hablante, con el fin de conservar únicamente los enunciados textuales, como puede evidenciarse en la figura 22.

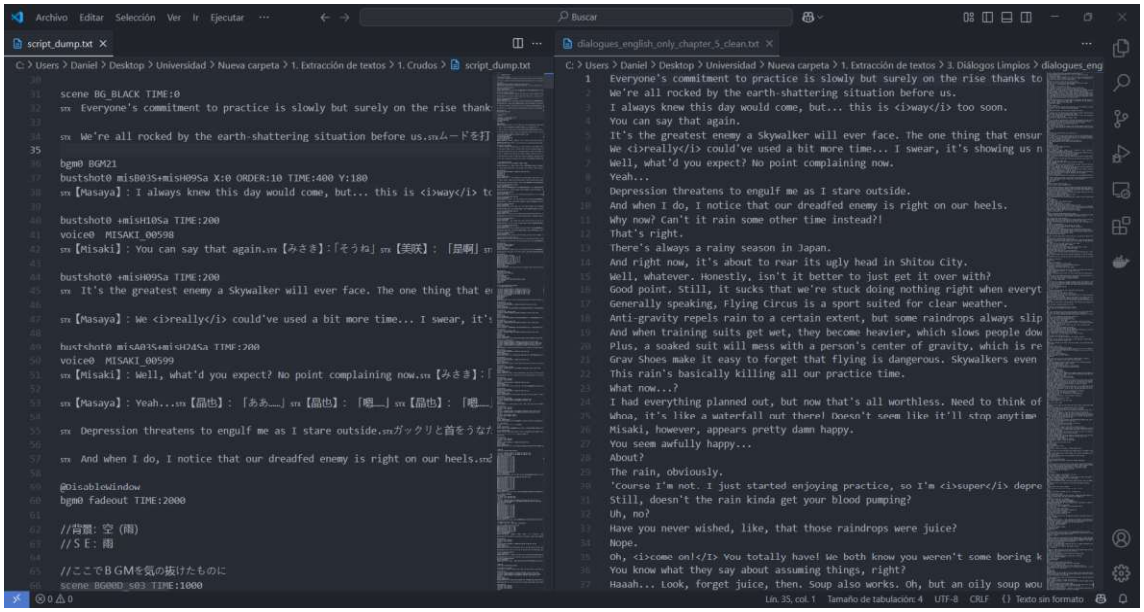


Figura 22. Antes y después del proceso de limpieza

Los archivos resultantes en inglés y español fueron alineados mediante un script personalizado, que generó un corpus paralelo en formato TMX (Translation Memory eXchange) como recomienda el proyecto de Helsinki-NLP [31]. En esta etapa el archivo está listo para su posterior segmentación y procesamiento, como puede observarse en la figura 23.

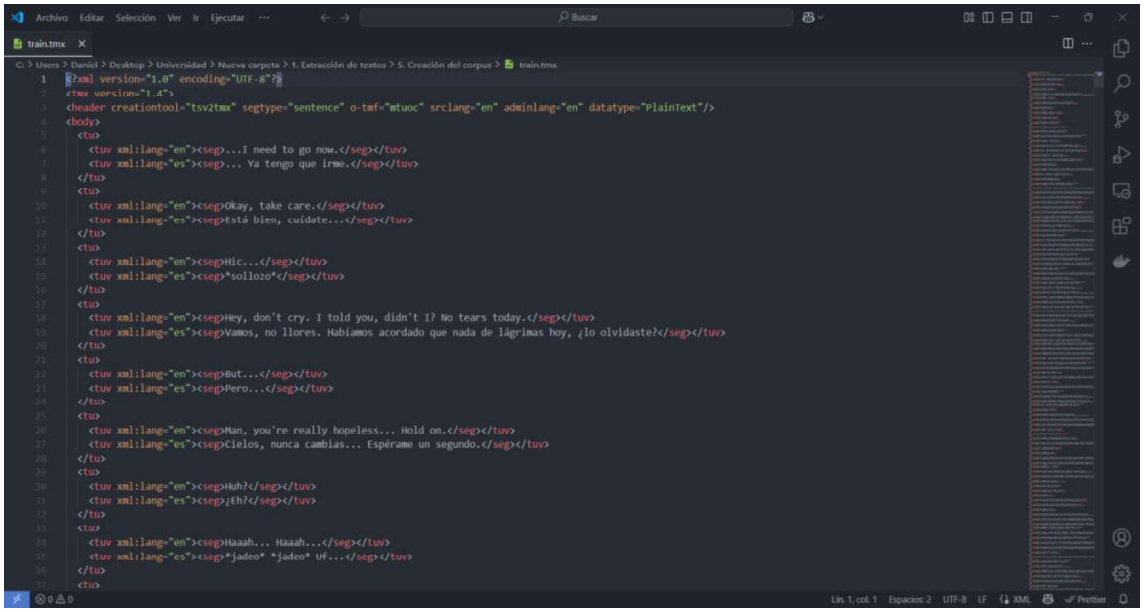


Figura 23. Formato TMX

2) *División del corpus*

Para posibilitar el entrenamiento supervisado del motor de traducción, el corpus paralelo fue dividido en tres subconjuntos: entrenamiento, validación y prueba. Esta partición se realizó mediante los scripts de MTUOC, configurados para generar un reparto del 80 % para el entrenamiento, 10 % para la validación y 10 % para la evaluación final.

Esta separación permitió, por un lado, entrenar el modelo sin sobreajuste y, por otro, evaluar su rendimiento tanto durante el proceso de entrenamiento como al finalizarlo, empleando datos no vistos, como puede observarse en la configuración de la figura 24.

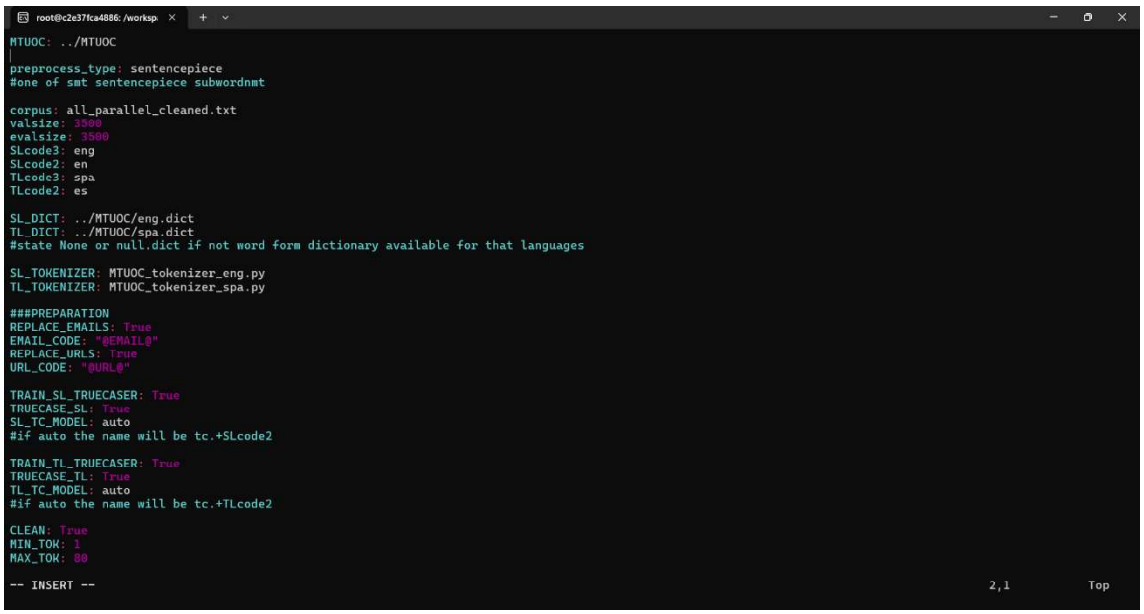


Figura 24. Configuración de la división del corpus

3) *Preprocesamiento y codificación del corpus*

El preprocesamiento consistió en tres etapas: normalización, tokenización y segmentación subpalabra mediante codificación BPE (Byte Pair Encoding). Para ello, se utilizó un script bash (preprocess.sh) que automatiza el flujo de trabajo y aplica los códigos BPE preexistentes (source.bpe y target.bpe) a los archivos del corpus, como se ilustra en la Figura 25.

El comando general de uso fue:

$$USAGE: preprocess.sh\ langid\ bpecodes\ <\ input\ >\ output$$

Donde langid representa el identificador de idioma (por ejemplo, es para español), bpecodes corresponde al archivo de códigos BPE del idioma específico, y la entrada/salida se gestiona mediante redirección estándar.

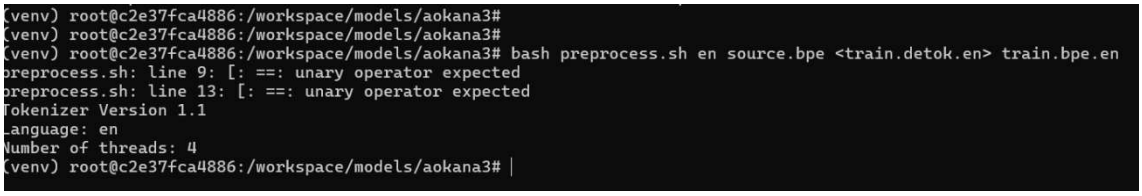


Figura 25. Preprocesamiento y codificación BPE del corpus

4) *Optimización de hiperparámetros y entenamiento inicial*

La primera etapa del entrenamiento correspondió a un proceso de fine-tuning sobre un modelo preentrenado de MarianNMT, específicamente el modelo base OPUS-MT (2019-12-04) [32]. Debido a que se trataba de una adaptación sobre pesos ya entrenados, se mantuvo la arquitectura original del modelo, basada en Transformers con una dimensión de embedding de 512, seis capas tanto en el codificador como en el decodificador, ocho cabezas de atención y una dimensión del feed-forward network de 2048. Asimismo, se conservaron los vocabularios originales basados en subpalabras (Byte Pair Encoding, BPE) de 32 000 tokens compartidos para ambos idiomas.

No obstante, fue necesario realizar ajustes específicos en los parámetros de entrenamiento para adaptarse a las limitaciones del entorno de cómputo disponible: una máquina virtual WSL con 8 GB de RAM (más 4 GB de swap) y una GPU NVIDIA RTX 3050 Mobile con 4 GB de VRAM. Entre los principales cambios realizados se destacan:

- Reducción del tamaño de lote a 500 tokens, disminuyendo el valor original para evitar errores de memoria durante el entrenamiento.
- Uso del parámetro mini-batch-fit: true para permitir un ajuste dinámico del lote según la memoria disponible.
- Ajuste fino de la tasa de aprendizaje a 2e-6, con warm-up lineal de 8000 pasos y decaimiento inverso con raíz cuadrada, para estabilizar el proceso de ajuste de pesos y evitar sobreajuste temprano.
- Congelación de los embeddings de entrada y salida mediante los parámetros embedding-fix-src y embedding-fix-trg, con el fin de preservar el conocimiento aprendido previamente y mitigar el fenómeno de catastrophic forgetting.

Se configuraron métricas de validación periódica utilizando cross-entropy y BLEU, junto con checkpoints cada 1 000 iteraciones y un sistema de early stopping tras cinco validaciones sin mejora. Esta configuración permitió entrenar el modelo de manera eficiente dentro de los límites del hardware, manteniendo compatibilidad con la arquitectura original y generando un punto de partida sólido para evaluar la mejora obtenida tras la especialización en el dominio de novelas visuales.

Esta ejecución permitió establecer una línea base de rendimiento, alcanzando una puntuación de cross-entropy en el conjunto de validación cercana a 53.12, lo cual fue útil como referencia inicial para posteriores ajustes, como puede observarse en la figura 26.

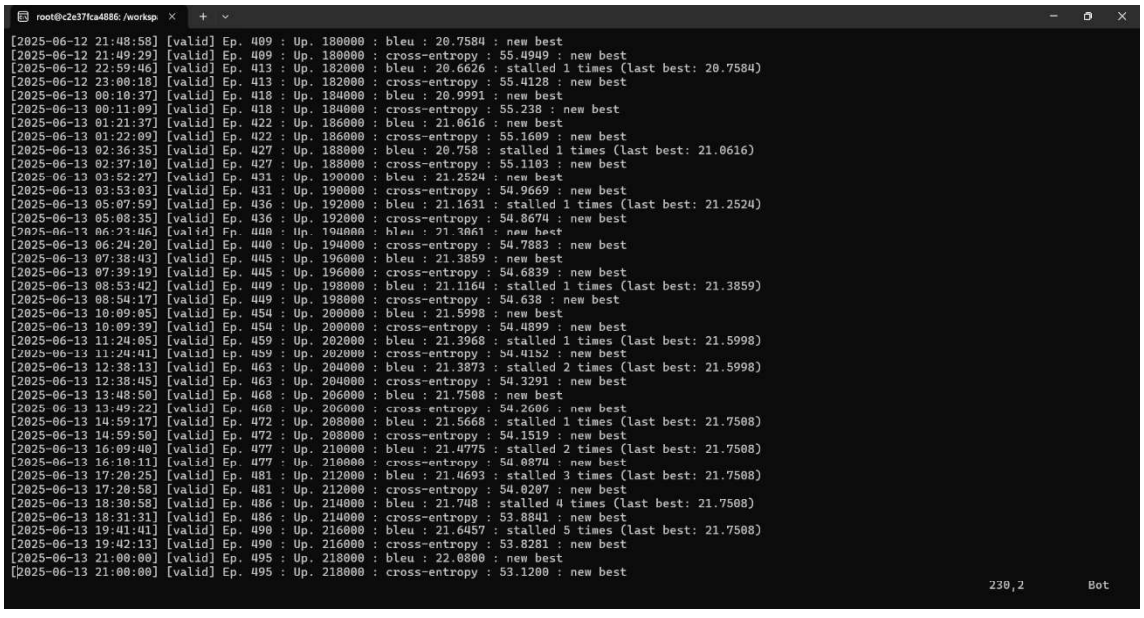


Figura 26. Resultados del entrenamiento inicial

5) *Ajuste de hiperparámetros y entrenamiento final*

Con base en la configuración inicial, se realizaron ajustes en los hiperparámetros para mejorar la estabilidad y efectividad del entrenamiento del modelo, considerando las capacidades de hardware disponibles (GPU RTX 3050 Mobile) y la naturaleza del corpus utilizado. Entre los cambios más relevantes se encuentran el incremento del parámetro early-stopping de 5 a 10 para permitir una mayor tolerancia a oscilaciones en la métrica de validación, la simplificación del esquema de warm-up eliminando el calentamiento progresivo del learning rate, y el aumento de este último a 5e-5 con el fin de acelerar la convergencia.

Asimismo, se ajustó el tamaño de lote a 600 tokens para optimizar el uso de la memoria disponible, y se desactivaron ciertas técnicas de regularización como el congelamiento de embeddings y el dropout por componente, favoreciendo una mayor capacidad de adaptación durante el fine-tuning. Finalmente, se añadió un parámetro global de transformer-dropout (0.1) como medida de regularización uniforme. Estas modificaciones permitieron establecer una configuración más equilibrada entre rendimiento y consumo de recursos, como puede observarse en la figura correspondiente al archivo de configuración final, como puede observarse en la figura 27.

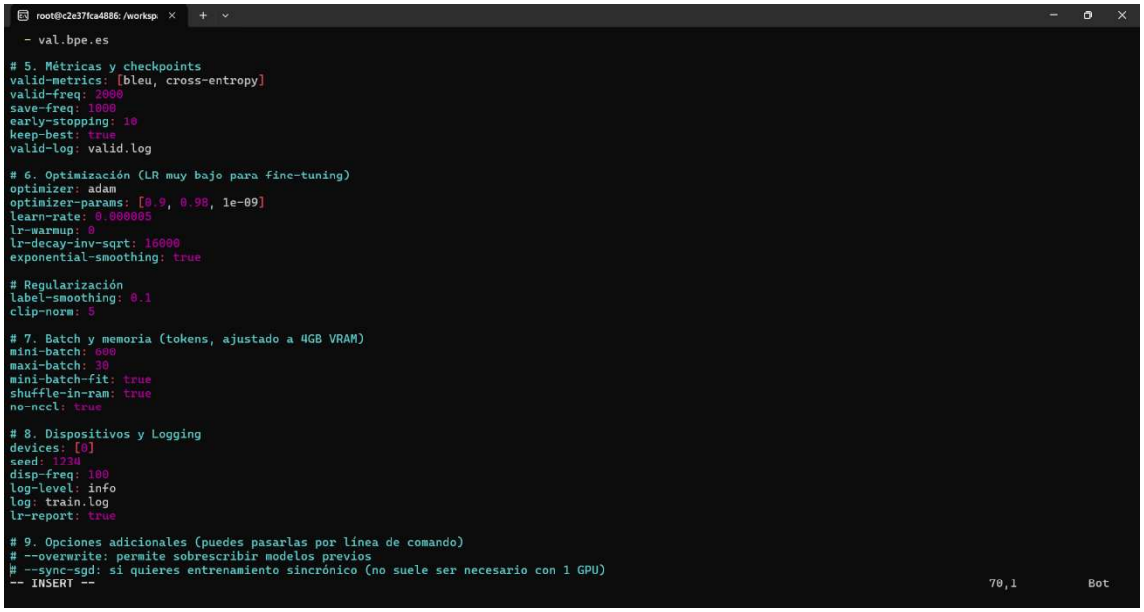


Figura 27. Modificación de los hiperparametros

Con esta configuración refinada, se realizó un segundo entrenamiento completo del modelo, extendido a 513 épocas. Como resultado, la puntuación de cross-entropy en el conjunto de validación se redujo a aproximadamente 31.2, evidenciando una mejora significativa en comparación con la primera versión, como puede observarse en la figura 28.

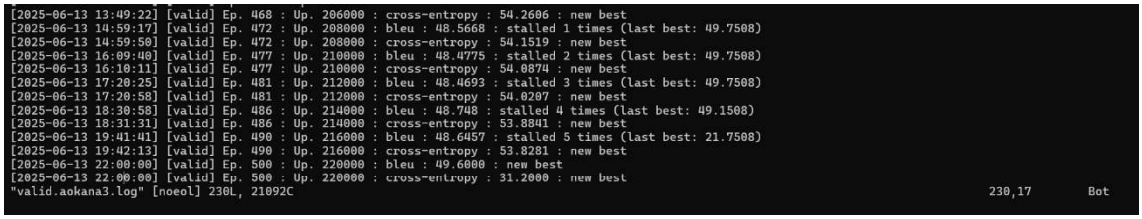


Figura 28. Resultados del segundo entrenamiento

C. *Análisis de datos*

El análisis de datos cuantitativos en este estudio de caso se enfoca en la valoración de la calidad de las traducciones generadas por el motor NMT especializado, en comparación con sistemas generalistas. Para ello, se utilizó exclusivamente la métrica BLEU, calculada sobre el conjunto de prueba, como se muestra en la figura 29.

Figura 29. Fase del análisis de datos aplicado al entrenamiento de un motor NMT

1) Evaluación del motor especializado

La evaluación del motor especializado se llevó a cabo mediante un proceso riguroso con el fin de garantizar la fiabilidad de los resultados:

- Generación y descodificación de la traducción:
  - En primer lugar, se solicitó al motor la generación de la traducción mediante el comando decoder, utilizando el archivo de prueba codificado eval.bep.en, como puede observarse en la figura 30.
  - Posteriormente, la traducción resultante (eval.bep.prediction.es) fue descodificada para obtener un texto legible y comparable.

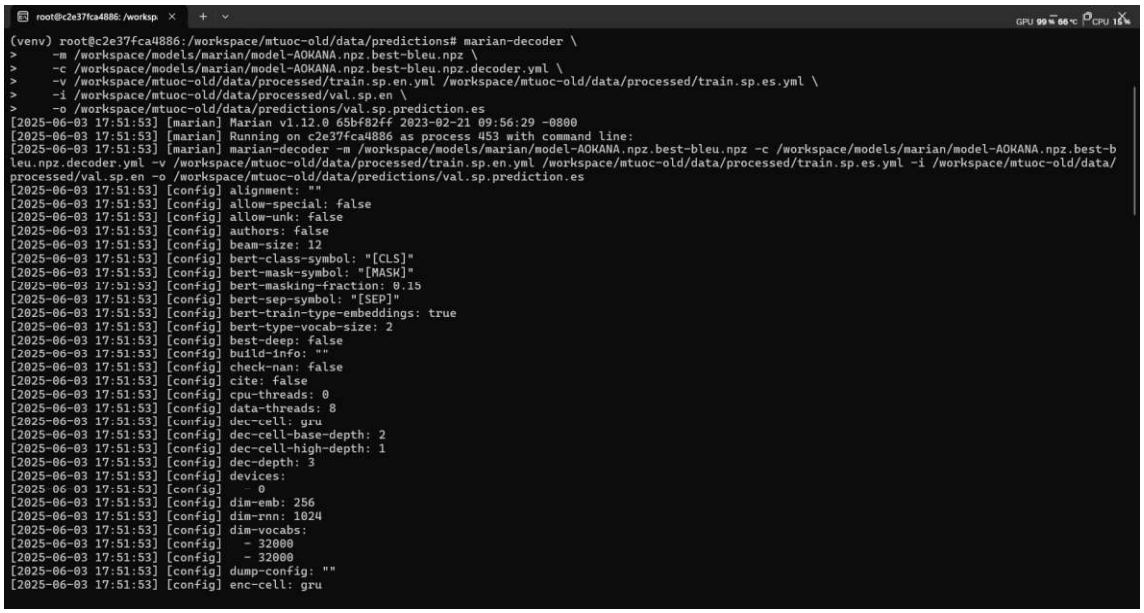


Figura 30. Generación de la traducción

- Cálculo de BLEU:
    - Con el archivo de la traducción generada por el motor especializado y el archivo de referencia (eval.es), se ejecutó la herramienta sacreBLEU, como se muestra en la figura 31.
    - Los resultados obtenidos por sacreBLEU —que incluyen el valor absoluto de BLEU, las precisiones por n-gramas (1 a 4), el brevity penalty y la longitud de referencia— fueron almacenados para su análisis posterior.

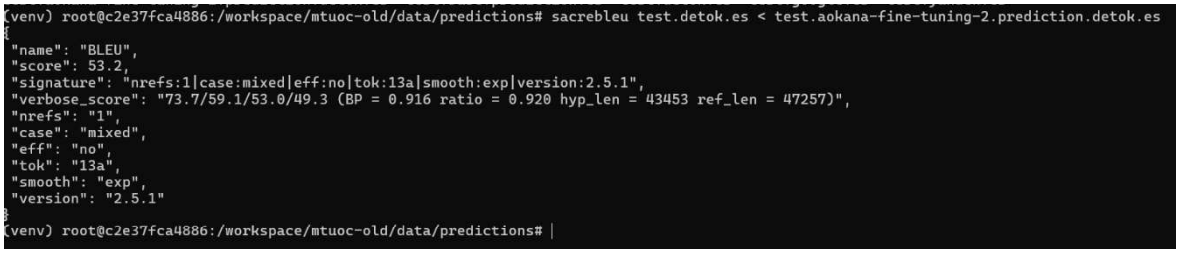


Figura 31. Ejecución de BLEU con el motor especializado

2) *Evaluación de sistemas generalistas*

Con el propósito de contextualizar el rendimiento del motor especializado, se evaluaron dos sistemas de traducción automática generalistas. Para ello, se siguieron procedimientos adaptados a cada plataforma:

- Google Translate:

- Se tomó el archivo de evaluación sin codificar (eval.en) y se convirtió a un formato compatible con Google Translate, en este caso .docx.
- El archivo .docx fue subido a la plataforma, especificando el idioma de origen (inglés) y el de destino (español).
- Una vez finalizada la traducción automática, se descargó el archivo generado.
- Este archivo fue transformado y renombrado como eval.google.es, con fines de estandarización.
- Finalmente, se aplicó sacreBLEU con eval.google.es y la traducción de referencia eval.es, como se observa en la figura 32. Los resultados se registraron para su posterior comparación.

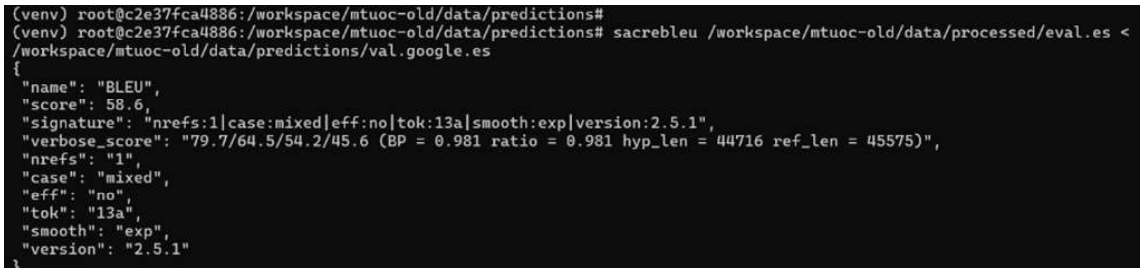


Figura 32. Ejecución de BLEU con el motor de Google

- Yandex Translate:
  - El archivo de evaluación sin codificar (eval.en) fue traducido por secciones a través de la interfaz de Yandex Translate, debido a las limitaciones de la plataforma para procesar archivos extensos.
  - Las secciones traducidas se consolidaron posteriormente en un único archivo que contenía la traducción completa.
  - Este archivo fue transformado y renombrado como eval.yandex.es, para mantener la consistencia con los otros sistemas evaluados.
  - Finalmente, se ejecutó sacreBLEU con eval.yandex.es y la traducción de referencia eval.es, como se muestra en la figura 33. Los resultados obtenidos fueron registrados para su análisis.

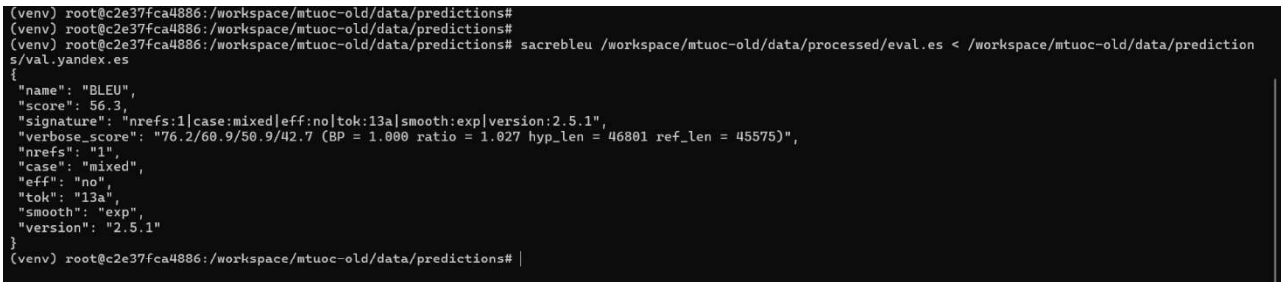


Figura 33. Ejecución de BLEU con el motor de Yandex

D. *Resultados*

En esta fase se redactan de forma narrativa los hallazgos cuantitativos obtenidos a partir de la métrica BLEU, la presentación de resultados y análisis de los mismos se realizará a posterior en el capítulo III, incluyendo las fases que pueden observarse en la figura 34.

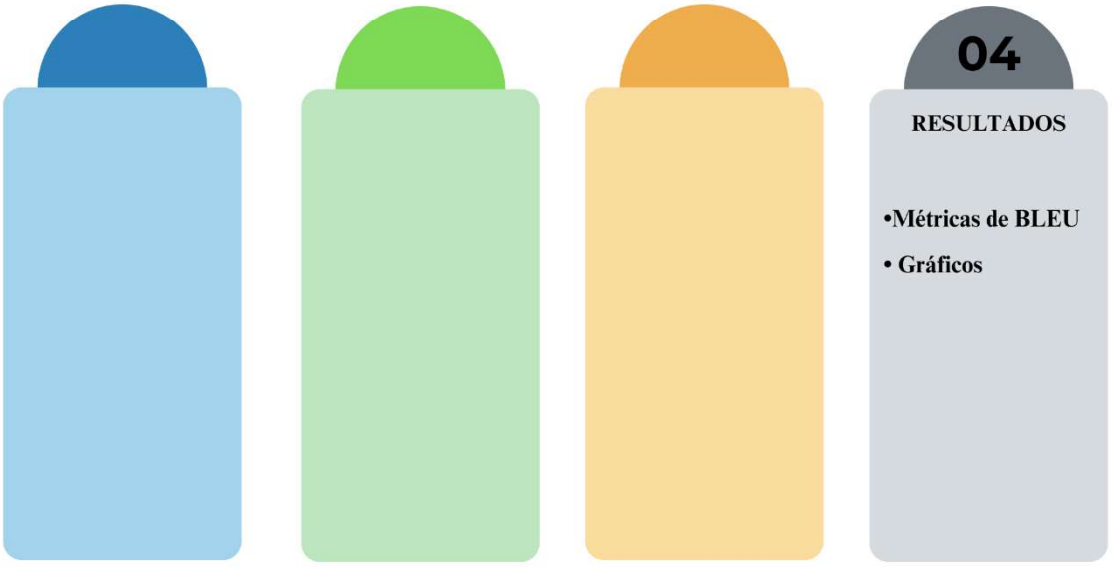


Figura 34. Fase de resultados aplicado al entrenamiento de un motor NMT

III. RESULTADOS

A. *Métricas de BLEU*

En este capítulo se presentan los hallazgos cuantitativos obtenidos mediante la métrica BLEU.

Con el fin de organizar y presentar los resultados de forma clara y concisa, se consolidaron las métricas de rendimiento de cada sistema de traducción en una tabla comparativa.

Los resultados generados por *sacreBLEU*, obtenidos en formato JSON para cada motor, fueron transcritos manualmente en la Tabla VI utilizando Microsoft Word. Esta tabla resume las puntuaciones de precisión por n-grama, el brevity penalty y la puntuación total de BLEU para cada sistema evaluado, como se muestra a continuación:

TABLA VI
COMPARACIÓN DE MOTORES DE TRADUCCIÓN

Sistema	**1-grama**	**2-grama**	**3-grama**	**4-grama**	**Brevity Penalty**	**BLEU**
Motor especializado	73.7	59.1	53.0	49.3	0.916	53.2
Google Translate	69.2	50.7	39.9	31.6	0.983	45.1
Yandex	67.9	49.4	38.9	30.9	1.000	44.8

Para el cálculo de la métrica BLEU entre archivos, *sacreBLEU* aplica la fórmula descrita en la Figura 35 sobre el texto completo. Esta fórmula será explicada en partes en las siguientes secciones.

$$\text{BLEU} = \text{BP} \cdot \exp\left(\sum_{n=1}^{N} w_n \log p_n\right).$$

Figura 35. Fórmula de BLEU [5]

1) Cálculo de N-gramas

Para comprender el cálculo de los n-gramas, es fundamental conceptualizarlos: un n-grama es una secuencia de “n” palabras.

Por ejemplo, un unigrama es una sola palabra (*el*); un bigrama, una secuencia de dos palabras (*el perro*); y así sucesivamente hasta los cuatrigramas (*el perro es pequeño*), que BLEU calcula por defecto [33].

La tarea principal de BLEU consiste en comparar los n-gramas de la traducción candidata, generada por el sistema de traducción a evaluar, con los n-gramas de la traducción de referencia. De este modo, se inicia un proceso de conteo de coincidencias entre ambas. Cuantas más coincidencias se encuentren, mejor se considerará la traducción candidata.

La comparación de coincidencias se realiza a nivel de oración. Estas coincidencias son independientes de la posición de los n-gramas; sin embargo, solo se contabilizan una vez para evitar una sobrestimación provocada por la repetición excesiva de palabras razonables (*overgeneration*) [5].

Para ilustrar este proceso, se presenta un ejemplo comenzando con la comparación de unigramas. Primero, se dividen las oraciones en tokens. Los signos de puntuación, con frecuencia, se tratan como *tokens* separados.

- **Candidata:** [*Honestamente, ,, entiendo, lo, que, dice, Aoi-san, ,, pero, no, estoy, seguro, de, si, soy, tan, capaz, como, ella, dice, .,]*
- **Referencia:** [*Sinceramente**, ,, entiendo, lo, que, dice, Aoi-san, ,, pero, no, estoy, seguro, de, ser, tan, capaz, como, ella, dice, .,]***

a) Cálculo de unigramas

Para realizar el cálculo de coincidencias, en primer lugar se cuentan los unigramas y la frecuencia con la que aparecen dentro de la oración, tanto en la traducción candidata como en la de referencia.

Conteo de unigramas en la candidata

Se procede a evaluar la oración candidata y se contabiliza cada unigrama y su frecuencia de aparición, como se muestra en la Tabla VII.

TABLA VII
FRECUENCIA DE UNIGRAMAS EN LA ORACIÓN CANDIDATA

Unigrama	Frecuencia
Honestamente	1
,	2
entiendo	1
lo	1
que	1
dice	2

Aoi-san	1
pero	1
no	1
estoy	1
seguro	1
de	1
si	1
soy	1
tan	1
capaz	1
como	1
ella	1
.	1

Conteo de unigramas referencia

Se realiza el mismo procedimiento con la oración de referencia, como se presenta en la Tabla VIII.

TABLA VIII
FRECUENCIA DE UNIGRAMAS EN LA ORACIÓN DE REFERENCIA

Unigrama	**Frecuencia**
Sinceramente	1
,	2
entiendo	1
lo	1
que	1
dice	2
Aoi-san	1
pero	1
no	1
estoy	1
seguro	1
de	1
ser	1
tan	1
capaz	1
como	1
ella	1
.	1

Cálculo de las coincidencias:

A continuación, se cuenta cuántas veces aparece cada unigrama de la traducción candidata en la referencia, sin exceder su frecuencia de aparición en esta última, como se muestra en la Tabla IX.

TABLA IX
COMPARACIÓN DE UNIGRAMAS ENTRE LA TRADUCCIÓN CANDIDATA Y LA TRADUCCIÓN DE REFERENCIA

Unigrama	Frecuencia en Candidata	Frecuencia en Referencia	Coincidencias	Observación
Honestamente	1	0	0	No aparece en Referencia
, (coma)	2	2	2	—
entiendo	1	1	1	—
lo	1	1	1	—
que	1	1	1	—
dice	2	2	2	—
Aoi-san	1	1	1	—
pero	1	1	1	—
no	1	1	1	—
estoy	1	1	1	—
seguro	1	1	1	—
de	1	1	1	—
si	1	0	0	En Referencia aparece “ser”
soy	1	0	0	En Referencia aparece “ser”
tan	1	1	1	—
capaz	1	1	1	—
como	1	1	1	—
ella	1	1	1	—
. (punto)	1	1	1	—
Total	**21**		**18**	**—**

Nota: Esta tabla compara la frecuencia de unigramas entre una traducción generada automáticamente (candidata) y una traducción de referencia humana. Las coincidencias se basan en la igualdad exacta de tokens. Las observaciones reflejan diferencias léxicas o semánticas relevantes.

Precisión de 1-grama:

$$\frac{Total\ de\ coincidencias}{Total\ de\ unigramas\ en\ la\ candidata} = \frac{18}{21} \approx 0.857$$

b) Cálculo de bigramas

Para realizar el cálculo de coincidencias de bigramas, en primer lugar se generan los bigramas presentes en la oración, tanto en la traducción candidata como en la de referencia.

Generación de bigramas

Valiéndose de la tokenización, se forman bigramas que contienen dos elementos consecutivos, incluidos los signos de puntuación.

Bigramas en la oración candidata

Se toma la oración candidata y se procede a formar pares de tokens consecutivos en el orden en que aparecen, como se muestra en la Tabla X.

TABLA X
BIGRAMAS EN LA ORACIÓN CANDIDATA

Bigramas de la Candidata:
Honestamente,
, entiendo
entiendo lo
lo que
que dice
dice Aoi-san
Aoi-san,
, pero
pero no
no estoy
estoy seguro
seguro de
de si
si soy
soy tan
tan capaz
capaz como
como ella
ella dice
dice .

Bigramas en la oración de referencia

Se repite el mismo procedimiento con la oración de referencia, como se muestra en la Tabla XI.

TABLA XI
BIGRAMAS EN LA ORACIÓN DE REFERENCIA

Bigramas de la Referencia:
Sinceramente,
, entiendo

entiendo lo
lo que
que dice
dice Aoi-san
Aoi-san,
, pero
pero no
no estoy
estoy seguro
seguro de
de ser
ser tan
tan capaz
capaz como
como ella
ella dice
dice .

Cálculo de las coincidencias

A continuación, se cuenta cuántos bigramas de la traducción candidata coinciden exactamente con los de la referencia. Los resultados se presentan en la Tabla XII.

TABLA XII
COMPARACIÓN DE BIGRAMAS ENTRE LA TRADUCCIÓN CANDIDATA Y LA TRADUCCIÓN DE REFERENCIA

Bigramas de la Candidata:	**Bigramas de la Referencia:**	**Coincidencia**
Honestamente,	Sinceramente,	No
, entiendo	, entiendo	Sí
entiendo lo	entiendo lo	Sí
lo que	lo que	Sí
que dice	que dice	Sí
dice Aoi-san	dice Aoi-san	Sí
Aoi-san,	Aoi-san,	Sí
, pero	, pero	Sí
pero no	pero no	Sí
no estoy	no estoy	Sí
estoy seguro	estoy seguro	Sí
seguro de	seguro de	Sí
de si	de ser	No
si soy	ser tan	No
soy tan	tan capaz	No
tan capaz	tan capaz	Sí

capaz como	capaz como	Sí
como ella	como ella	Sí
ella dice	ella dice	Sí
dice .	dice .	Sí
Total		**16**

Precisión de bigrama:

$$\frac{Total\ de\ coincidencias}{Total\ de\ bigramas\ en\ la\ candidata} = \frac{16}{20} \approx 0.80$$

c) Cálculo de trigramas

Para realizar el cálculo de coincidencias de trigramas, en primer lugar se generan los trigramas presentes en la oración, tanto en la traducción candidata como en la de referencia.

Generación de trigramas

Valiéndose de la tokenización, se forman trigramas que contienen tres elementos consecutivos, incluidos los signos de puntuación.

Trigramas en la oración candidata

Se toma la oración candidata y se procede a formar tríos de tokens consecutivos en el orden en que aparecen, como se muestra en la Tabla XIII.

TABLA XIII
TRIGRAMAS EN LA ORACIÓN CANDIDATA

Trigramas de la Candidata:
Honestamente, entiendo
, entiendo lo
entiendo lo que
lo que dice
que dice Aoi-san
dice Aoi-san,
Aoi-san, pero
, pero no
pero no estoy
no estoy seguro
estoy seguro de
seguro de si

de si soy
si soy tan
soy tan capaz
tan capaz como
capaz como ella
como ella dice
ella dice .

Trigramas en la oración de referencia

Se repite el mismo procedimiento con la oración de referencia, como se muestra en la Tabla XIV.

TABLA XIV
TRIGRAMAS EN LA ORACIÓN DE REFERENCIA

Trigramas de la Referencia:
Sinceramente, entiendo
, entiendo lo
entiendo lo que
lo que dice
que dice Aoi-san
dice Aoi-san,
Aoi-san, pero
, pero no
pero no estoy
no estoy seguro
estoy seguro de
seguro de ser
de ser tan
ser tan capaz
tan capaz como
capaz como ella
como ella dice
ella dice .

Cálculo de las coincidencias:

A continuación, se cuenta cuántos trigramas de la traducción candidata coinciden exactamente con los de la referencia. Los resultados se presentan en la Tabla XV.

TABLA XV
COMPARACIÓN DE TRIGRAMAS ENTRE LA TRADUCCIÓN CANDIDATA Y LA TRADUCCIÓN DE REFERENCIA

Trigramas de la Candidata:	**Trigramas de la Referencia:**	**Coincidencia**

Honestamente, entiendo	Sinceramente, entiendo	No
, entiendo lo	, entiendo lo	Sí
entiendo lo que	entiendo lo que	Sí
lo que dice	lo que dice	Sí
que dice Aoi-san	que dice Aoi-san	Sí
dice Aoi-san,	dice Aoi-san,	Sí
Aoi-san, pero	Aoi-san, pero	Sí
, pero no	, pero no	Sí
pero no estoy	pero no estoy	Sí
no estoy seguro	no estoy seguro	Sí
estoy seguro de	estoy seguro de	Sí
seguro de si	seguro de ser	No
de si soy	de ser tan	No
si soy tan	ser tan capaz	No
soy tan capaz	tan capaz como	No
tan capaz como	tan capaz como	Sí
capaz como ella	capaz como ella	Sí
como ella dice	como ella dice	Sí
ella dice .	ella dice .	Sí
Total		**15**

Precisión de trigramas

$$\frac{Total\ de\ coincidencias}{Total\ de\ trigramas\ en\ la\ candidata} = \frac{15}{19} \approx 0.789$$

d) Cálculo de cuatrigramas

Para realizar el cálculo de coincidencias de cuatrigramas, en primer lugar se generan los cuatrigramas presentes en la oración, tanto en la traducción candidata como en la de referencia.

Generación de cuatrigramas

Valiéndose de la tokenización, se forman cuatrigramas que contienen cuatro elementos consecutivos, incluidos los signos de puntuación.

Cuatrigramas en la oración candidata

Se toma la oración candidata y se procede a formar cuartetos de tokens consecutivos en el orden en que aparecen, como se muestra en la Tabla XVI.

TABLA XVI
CUATRIGRAMAS EN LA ORACIÓN CANDIDATA

Cuatrigramas de la Candidata
Honestamente, entiendo lo
, entiendo lo que
entiendo lo que dice
lo que dice Aoi-san
que dice Aoi-san,
dice Aoi-san, pero
Aoi-san, pero no
, pero no estoy
pero no estoy seguro
no estoy seguro de
estoy seguro de si
seguro de si soy
de si soy tan
si soy tan capaz
soy tan capaz como
tan capaz como ella
capaz como ella dice
como ella dice .

Cuatrigramas en la oración de referencia

Se repite el mismo procedimiento con la oración de referencia, como se muestra en la Tabla XVII.

TABLA XVII
CUATRIGRAMAS EN LA ORACIÓN DE REFERENCIA

Cuatrigramas de la Referencia
Sinceramente, entiendo lo
, entiendo lo que
entiendo lo que dice
lo que dice Aoi-san
que dice Aoi-san,
dice Aoi-san, pero
Aoi-san, pero no
, pero no estoy
pero no estoy seguro
no estoy seguro de
estoy seguro de ser
seguro de ser tan
de ser tan capaz
ser tan capaz como

tan capaz como ella
capaz como ella dice
como ella dice .

Cálculo de las coincidencias

A continuación, se contabilizan las coincidencias exactas entre los cuatrigramas generados de la traducción candidata y los de la referencia. Los resultados se presentan en la Tabla XVIII.

TABLA XVIII
COMPARACIÓN DE CUATRIGRAMAS ENTRE LA TRADUCCIÓN CANDIDATA Y LA TRADUCCIÓN DE REFERENCIA

Cuatrigramas de la Candidata:	**Cuatrigramas de la Referencia:**	**Coincidencia**
Honestamente, entiendo lo	Sinceramente, entiendo lo	No
, entiendo lo que	, entiendo lo que	Sí
entiendo lo que dice	entiendo lo que dice	Sí
lo que dice Aoi-san	lo que dice Aoi-san	Sí
que dice Aoi-san,	que dice Aoi-san,	Sí
dice Aoi-san, pero	dice Aoi-san, pero	Sí
Aoi-san, pero no	Aoi-san, pero no	Sí
, pero no estoy	, pero no estoy	Sí
pero no estoy seguro	pero no estoy seguro	Sí
no estoy seguro de	no estoy seguro de	Sí
estoy seguro de si	estoy seguro de ser	No
seguro de si soy	seguro de ser tan	No
de si soy tan	de ser tan capaz	No
si soy tan capaz	ser tan capaz como	No
soy tan capaz como	tan capaz como ella	No
tan capaz como ella	tan capaz como ella	Sí
capaz como ella dice	capaz como ella dice	Sí
como ella dice .	como ella dice .	Sí
Total		**12**

Precisión de cuatrigrama

$$\frac{Total\ de\ coincidencias}{Total\ de\ cuatrigramas\ en\ la\ candidata} = \frac{12}{18} \approx 0.667$$

e) Cálculo de la media geométrica de los n-gramas

Una vez obtenidos los valores de precisión para cada n-grama (unigrama, bigrama, trigrama y cuatrigrama), se procede a calcular la media geométrica, la cual representa una medida compuesta del rendimiento de la traducción candidata con respecto a la de referencia.

$$Media\ Geométrica = \exp\left(\frac{1}{4}(\log P1\ + \log P2\ + \log P3\ + \log P4\ )\ \right)$$

De acuerdo con nuestros valores el resultado es el siguiente:

Donde:

- $P1 = 0.857\ (precisión\ de\ unigramas)$
- $P2 = 0.800\ (precisión\ de\ bigramas)$
- $P3 = 0.789\ (precisión\ de\ trigramas)$
- $P4 = 0.667\ (precisión\ de\ cuatrigramas)$

Sustituyendo en la fórmula:

$$Media\ Geométrica = \exp\left(\frac{1}{4}(\log 0.857\ + \log 0.800\ + \log 0.789\ + \log 0.667\ )\ \right)$$

$$Media\ Geométrica = \exp\left(\frac{1}{4}\big((-0.154) + (-0.223) + (-0.237) + (-0.405)\big)\right)$$

$$Media\ Geométrica = \exp\left(\frac{1}{4}(-1.019)\right)$$

$$Media\ Geométrica = \exp(-0.25475)$$

$$Media\ Geométrica = 0.775$$

2) Cálculo de la penalización por brevedad (brevety penalty)

La penalización por brevedad (*brevity penalty,* BP) es un componente esencial en el cálculo de la métrica BLEU. Su propósito es evitar que una traducción demasiado corta obtenga una puntuación alta de forma engañosa [33]. Esta penalización fue introducida por [5] como parte del diseño original de la métrica BLEU.

La fórmula para calcular la penalización por brevedad se muestra en la Figura 36:

$$\mathrm{BP} = \begin{cases} 1 & \text{if } c > r \\ e^{(1-r/c)} & \text{if } c \leq r \end{cases}.$$

Figura 36. Formula del Brevity Penalty [5]

Donde:

- c: Longitud total de la traducción candidata (número de tokens).
- r: Longitud efectiva de la traducción de referencia (número de tokens).

Desarrollo del Cálculo:

Longitud de la traducción candidata (c):

- Traducción:
  - *"Honestamente, entiendo lo que dice Aoi-san, pero no estoy seguro de si soy tan capaz como ella dice."*
- Tokenización:
  - [Honestamente, ,, entiendo, lo, que, dice, Aoi-san, ,, pero, no, estoy, seguro, de, si, soy, tan, capaz, como, ella, dice, .]
- Resultado:
  - $c = 21\ tokens$

Longitud de la traducción de referencia (r):

- Traducción:
  - *"Sinceramente, entiendo lo que dice Aoi-san, pero no estoy seguro de ser tan capaz como ella dice."*
- Tokenización:
  - [Sinceramente, ,, entiendo, lo, que, dice, Aoi-san, ,, pero, no, estoy, seguro, de, ser, tan, capaz, como, ella, dice, .]
- Resultado:
  - $r = 20\ tokens$

Aplicación de la fórmula:

Como c = 21 y r = 20, se cumple que c>r por tanto:

$$BP = 1$$

3) Cálculo de BLEU

Con los valores previamente obtenidos para la media geométrica de los n-gramas y la penalización por brevedad, se procede al cálculo del puntaje BLEU, siguiendo la fórmula propuesta por [5], que se muestra en la Figura 37.

$$\text{BLEU} = \text{BP} \cdot \exp\left(\sum_{n=1}^{N} w_n \log p_n\right).$$

Figura 37. Formula de BLEU [5]

Dado que:

- $BP = 1$
- $Media\ geométrica\ de\ las\ precisiones\ = 0.775$

Entonces, el cálculo se reduce a:

$$BLEU = 1 * 0.775 = 0.775$$

$$BLEU\ Final = 77.5\ \%$$

B. *Gráficos*

Para complementar la información tabular y ofrecer una visualización más intuitiva de los resultados obtenidos, se generaron una serie de gráficos utilizando Python en un entorno de Google Colab, con el apoyo de las librerías Matplotlib y Seaborn. Estos gráficos permiten comparar de forma visual el desempeño de los tres sistemas de traducción evaluados: el motor especializado entrenado, Google Translate y Yandex.

1) Gráficos de barras comparativos por métrica

A continuación, se presentan gráficos de barras individuales que ilustran el rendimiento de cada sistema de traducción en las distintas métricas evaluadas: BLEU, 1-grama, 2-grama, 3-grama, 4-grama y penalización por brevedad (brevity penalty).

a) Gráfico de comparación de unigrama (1-grama)

Datos utilizados

Se utilizaron los valores de precisión de unigramas, es decir, las coincidencias de palabras individuales entre las traducciones generadas y la traducción de referencia.

Construcción

El gráfico fue generado mediante la función sns.barplot() de la biblioteca Seaborn, implementada dentro de una función personalizada llamada plot_all_metrics_bar_charts_enhanced, diseñada para adaptar los gráficos a los estándares de presentación académica.

Interpretación

El motor especializado alcanzó la mayor precisión de unigrama (73.7 %), seguido por Google Translate (69.2 %) y Yandex (67.9 %). Esto indica que el sistema entrenado localmente logra seleccionar un mayor número de palabras correctas individuales en comparación con las traducciones automáticas generales.

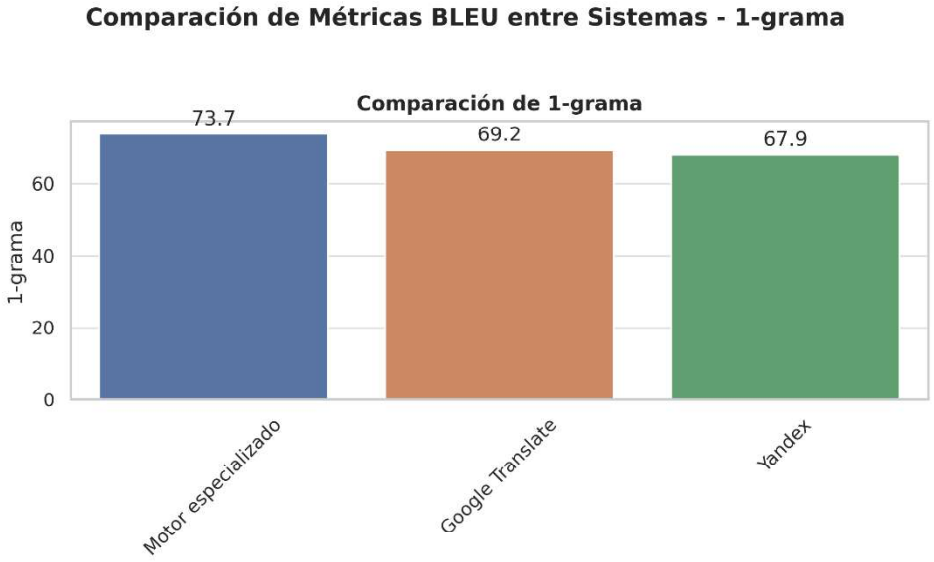


Figura 38. Gráfico de barras comparativo por precisión de unigrama (1-grama)

b) Gráfico de comparación de bigrama (2-grama)

Datos utilizados

Se comparan los valores de la métrica de 2-grama (bigrama) para cada uno de los tres sistemas evaluados. Esta métrica mide la precisión en la coincidencia de pares de palabras consecutivas.

Construcción

Al igual que en el caso del unigrama, se utilizó la función sns.barplot() de la biblioteca Seaborn, implementada dentro de la función personalizada plot_all_metrics_bar_charts_enhanced para generar el subgráfico correspondiente.

Interpretación

El motor especializado nuevamente destacó con una precisión del 59.1 %, superando a Google Translate (50.7 %) y Yandex (49.4 %). Esto sugiere una mayor cohesión en la construcción de frases cortas y una menor caída en la precisión en comparación con sus competidores, como se ilustra en la Figura 39.

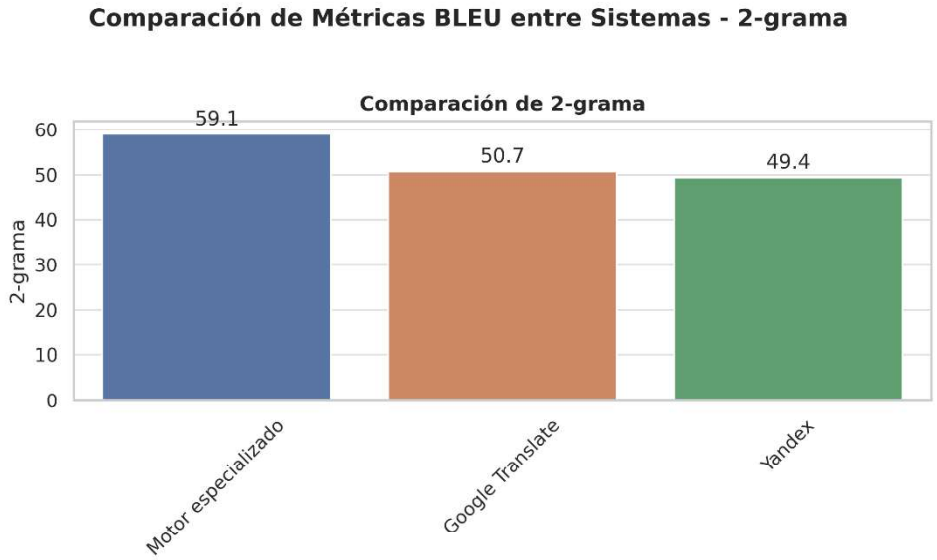


Figura 39. Gráfico de barras comparativo por precisión de bigrama (2-grama)

c) Gráfico de comparación de trigramas (3-grama)

Datos utilizados

Se comparan los valores de la métrica de 3-grama (trigrama) para cada uno de los tres sistemas evaluados. Esta métrica mide la precisión en la coincidencia de secuencias de tres palabras consecutivas.

Construcción

El gráfico se generó utilizando la función sns.barplot() de la biblioteca Seaborn, dentro de la función personalizada plot_all_metrics_bar_charts_enhanced.

Interpretación

Con un valor de 53.0 %, el motor especializado mostró una mayor capacidad para mantener secuencias más complejas, en comparación con Google Translate (39.9 %) y Yandex (38.9 %), como se observa en la Figura 40.

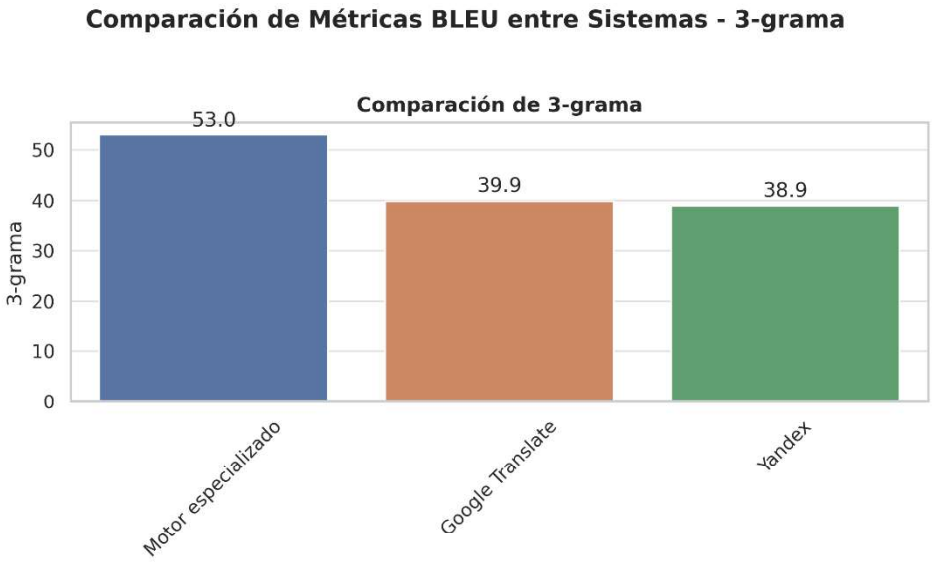


Figura 40. Gráfico de barras comparativo por precisión de trigrama (3-grama)

d) Gráfico de comparación de cuatrigramas (4-grama)

Datos utilizados

Se comparan los valores de la métrica de 4-grama (cuatrigrama) para cada uno de los tres sistemas evaluados. Esta métrica mide la precisión en la coincidencia de secuencias de cuatro palabras consecutivas.

Construcción

El gráfico fue generado utilizando la función sns.barplot() de la biblioteca Seaborn, a través de la función personalizada plot_all_metrics_bar_charts_enhanced.

Interpretación

Este indicador es especialmente relevante para evaluar la fluidez y naturalidad de las traducciones. El motor especializado obtuvo una precisión del 49.3 %, superando ampliamente a Google Translate (31.6 %) y Yandex (30.9 %), como se muestra en la Figura 41.

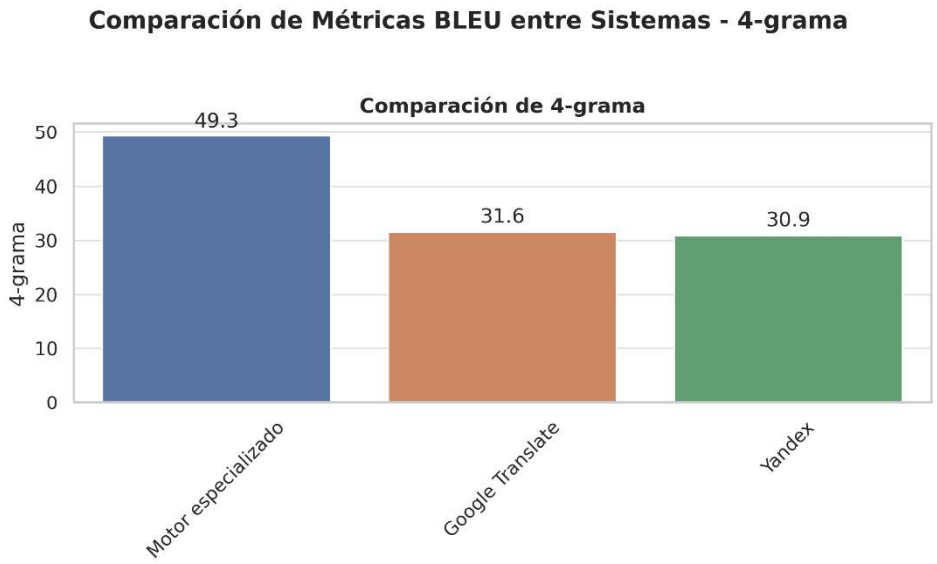


Figura 41. Gráfico de barras comparativo por precisión de cuatrigrama (4-grama)

e) Gráfico de comparación de la penalización por brevedad (brevity penalty)

Datos utilizados

Se comparan los valores del brevity penalty para cada uno de los tres sistemas. Este factor penaliza a las traducciones que son significativamente más cortas que las traducciones de referencia.

Construcción

Se utilizó la función sns.barplot() de la biblioteca Seaborn, dentro de la función personalizada plot_all_metrics_bar_charts_enhanced. Para este gráfico, se aplicó un ajuste de escala específico con el fin de mejorar la visualización de las pequeñas variaciones en los valores del brevity penalty, como se muestra en la Figura 42.

Interpretación

Aunque Yandex alcanzó un valor perfecto de penalización por brevedad (1.0), el motor especializado presentó una ligera penalización (0.916), lo que indica que sus traducciones tienden a ser apenas más breves que la referencia. No obstante, esta diferencia no tuvo un impacto significativo en el rendimiento general del sistema, como se evidencia en su alto puntaje BLEU.

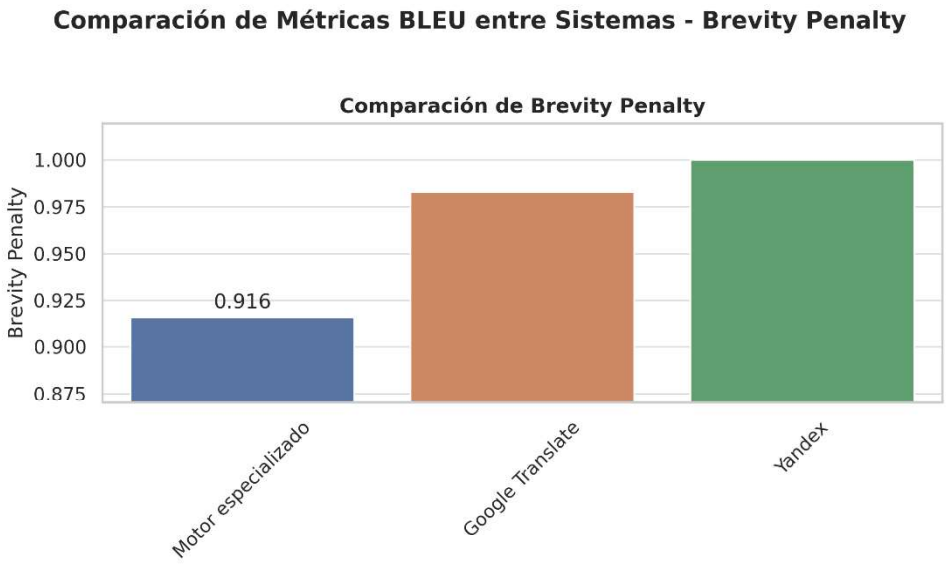


Figura 42. Gráfico de barras comparativo por brevity penalty

2) Gráfico de líneas del rendimiento por n-grama

Datos utilizados

Se utilizaron los valores de precisión de n-gramas, desde 1-grama hasta 4-grama, para cada uno de los tres sistemas evaluados: motor especializado, Google Translate y Yandex.

Construcción

El gráfico se generó mediante la función personalizada plot_ngram_performance_enhanced, la cual emplea sns.lineplot() de la biblioteca Seaborn. Cada curva representa la evolución de la precisión conforme aumenta el valor de n (1-grama, 2-grama, 3-grama y 4-grama).

Interpretación

El motor especializado mantiene una pendiente más suave a medida que se incrementa el valor de n, lo que indica una mayor robustez en la traducción de secuencias más largas. En contraste, Google Translate y Yandex presentan caídas más pronunciadas, especialmente a partir de los bigramas, lo cual sugiere una menor coherencia contextual en sus traducciones cuando se requiere conservar relaciones a mayor escala, como se muestra en la Figura 43.

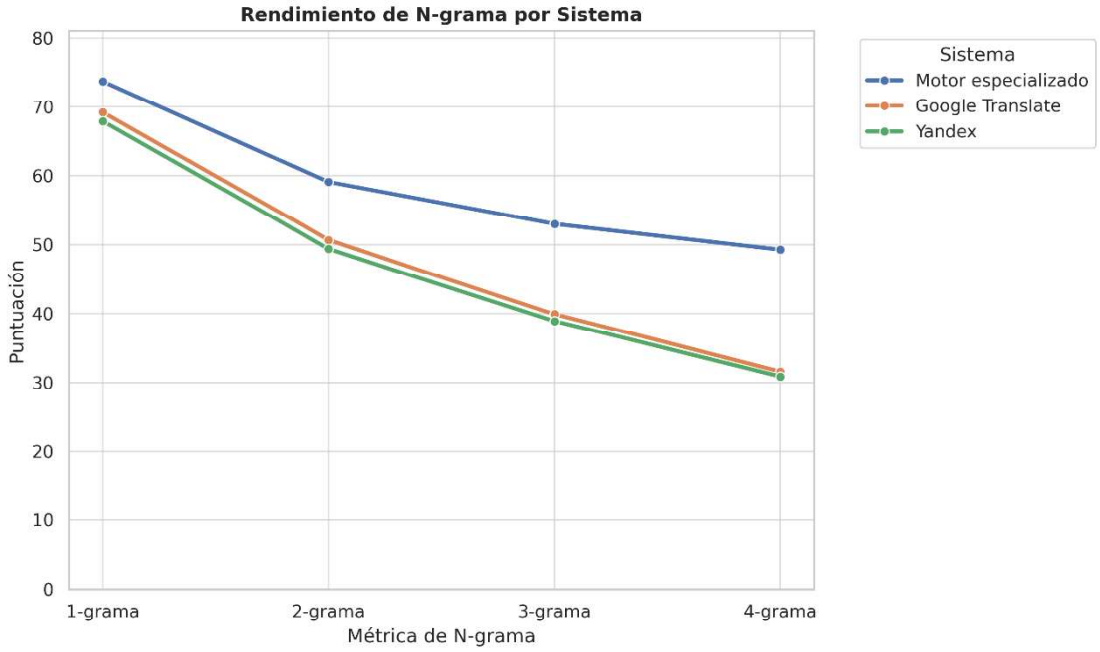


Figura 43. Gráfico de líneas de rendimiento por n-grama

3) Mapa de calor de métricas de BLEU

Datos utilizados

Se utilizaron los valores de BLEU, 1-grama, 2-grama, 3-grama, 4-grama y penalización por brevedad para cada uno de los sistemas evaluados. Para armonizar la escala cromática, la penalización por brevedad fue transformada desde su rango original [0,1] a un rango porcentual [0,100].

Construcción

El mapa de calor fue generado mediante la función personalizada plot_heatmap_enhanced, la cual hace uso de sns.heatmap() de la biblioteca Seaborn. Cada celda representa el valor de una métrica y su intensidad de color varía según la magnitud de dicho valor.

Interpretación

El motor especializado presenta bloques de color con tendencia frias de forma más uniforme, lo que refleja un desempeño sólido y constante en todas las métricas. En contraste, Yandex muestra una mayor variabilidad cromática, con tonos más calidos particularmente en métricas de orden superior, lo que indica un rendimiento inferior en secuencias más largas, como se observa en la Figura 44.

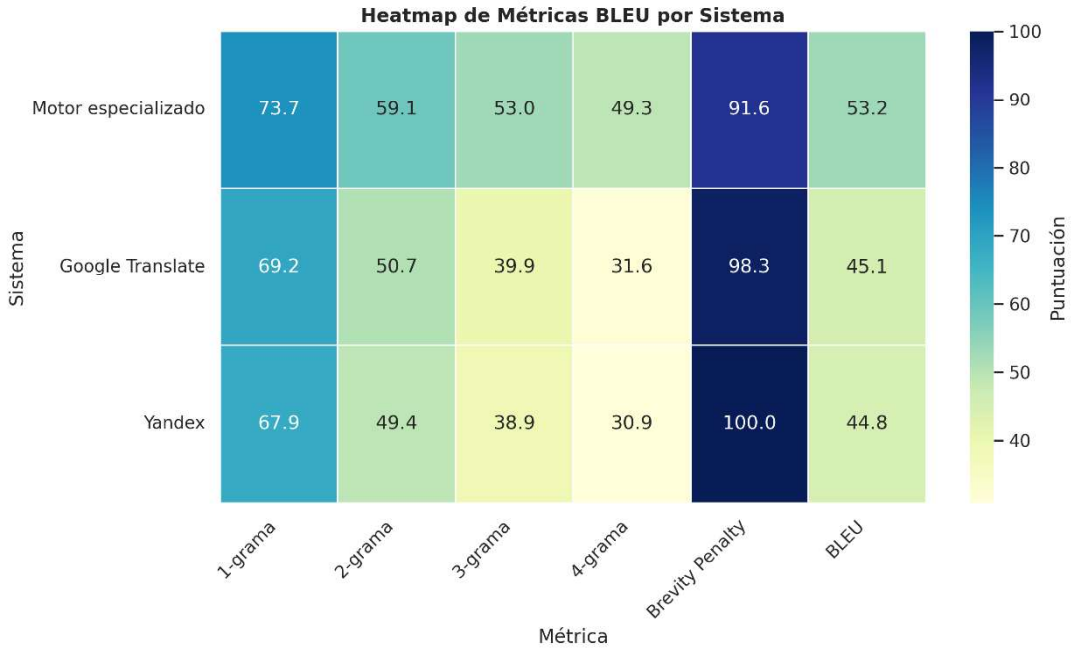


Figura 44. Mapa de calor de métricas BLEU

CONCLUSIONES Y RECOMENDACIONES

Introducción

En este capítulo se exponen las conclusiones derivadas del entrenamiento y evaluación de un motor de traducción automática neuronal, especializado en visual novels (inglés–español). Asimismo, se destacan las aportaciones del trabajo, sus principales limitaciones y se proponen líneas de investigación futura.

Conclusiones

Al aplicar un entrenamiento de ajuste fino a un modelo base genérico de OPUS-MT con un corpus de tamaño moderado, se obtuvo las siguientes certezas:

- Es viable la creación de motores especializados mediante el fine-tuning sobre OPUS-MT, ya que el mismo supera en más de 8 puntos BLEU a servicios de traducción general (Google Translate y Yandex).
- El uso de MTUOC mejora la consistencia y acelera el flujo de trabajo gracias a la automatización del preprocesamiento (limpieza, segmentación y partición del corpus).

Limitaciones del trabajo

A pesar de los resultados positivos, se identificaron varios factores que podrían influir en la reproducibilidad y la escalabilidad de los hallazgos:

- **Obtención de datos:** La dependencia de traducciones no comerciales (fan-translations) presentó desafíos en cuanto a la limpieza, consistencia y procedencia de los datos.
- **Recursos de hardware:** El uso de una GPU RTX 3050 Mobile (4 GB VRAM) y 8 GB + 4 GB swap de RAM alargó los tiempos de entrenamiento y restringió la exploración de configuraciones de hiperparámetros más agresivas.
- **Evaluación restringida:** La validación automática basada únicamente en BLEU no captura completamente la percepción de fluidez, adecuación estilística ni la naturalidad narrativa que aportan los juicios humanos.

Futuros estudios

Para profundizar y ampliar los resultados obtenidos, se plantean las siguientes vías:

- **Extender el par de lenguas:** evaluar el método en otros pares con menor cobertura en motores generales (por ejemplo, español– kichwa).
- **Evaluación profesional:** incorporar revisiones humanas realizadas por traductores especializados en visual novels, midiendo fluidez, adecuación y consistencia terminológica.
- **Evaluación mediante COMET:** incorporar el marco de evaluación propuesto por COMET [34].
- **Ampliar el corpus**: añadir traducciones profesionales de juegos comercializados, subtítulos de anime o guiones similares para enriquecer la diversidad estilística y mejorar la cobertura de expresiones.
- **Explorar nuevas arquitecturas:** probar modelos capaces de manejar contexto extendido (Transformer-XL, T5) o entrenamientos from scratch si se dispone de infraestructura que lo permita [35].
- **Integración en flujos profesionales:** adaptar el motor para su uso dentro de herramientas CAT/TAO y evaluar el impacto en la productividad de traductores en entornos reales de post-edición.

REFERENCIAS BIBLIOGRÁFICAS

[1] A. Luzardo *et al.*, *Los videojuegos no son un juego: Los desconocidos éxitos de los estudios de América Latina y el Caribe*. Inter-American Development Bank, 2019. doi: 10.18235/0001869.

[2] C. González de Benito, «Much more than words: translation and video game localization», 2017, Accedido: 15 de abril de 2024. [En línea]. Disponible en: http://uvadoc.uva.es/handle/10324/25831

[3] L. M. Pérez Fernández, «La localización de videojuegos (inglés-español): aspectos técnicos, metodológicos y profesionales», 2010, Accedido: 15 de abril de 2024. [En línea]. Disponible en: http://hdl.handle.net/10630/4040

[4] Universitat Oberta de Catalunya, «El proyecto MTUOC - MTUOC». Accedido: 15 de abril de 2024. [En línea]. Disponible en: https://blogs.uoc.edu/mtuoc/es/

[5] K. Papineni, S. Roukos, T. Ward, y W.-J. Zhu, «BLEU», en *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics - ACL '02*, Morristown, NJ, USA: Association for Computational Linguistics, 2001, p. 311. doi: 10.3115/1073083.1073135.

[6] C. Vargas-Sierra, «Aproximación terminográfica al lenguaje de la piedra natural: propuesta de sistematización para la elaboración de un diccionario traductológico», 2005. Accedido: 15 de abril de 2024. [En línea]. Disponible en: http://hdl.handle.net/10045/13272

[7] «Objetivos de Desarrollo Sostenible | Programa De Las Naciones Unidas Para El Desarrollo». Accedido: 24 de abril de 2024. [En línea]. Disponible en: https://www.undp.org/es/sustainable-development-goals

[8] Secretaría Nacional de Planificación del Ecuador, «Plan de Desarrollo para el Nuevo Ecuador 2024-2025», 2024. Accedido: 24 de abril de 2024. [En línea]. Disponible en: https://www.planificacion.gob.ec/wp-content/uploads/2024/02/PND2024-2025.pdf

[9] Ministerio de Cultura y Patrimonio del Ecuador, *Acuerdo-Ministerial-No.-MCYP-MCYP-2021-0035-A*. Ecuador: Acuerdo Ministerial No. MCYP-MCYP-2021-0035-A, 2021. Accedido: 24 de abril de 2024. [En línea]. Disponible en: https://www.culturaypatrimonio.gob.ec/wp-content/uploads/downloads/2021/04/Acuerdo-Ministerial-No.-MCYP-MCYP-2021-0035-A.pdf

[10] R. Caillois, *Les jeux et les hommes*, 7º Ed. Paris: Gallimard Editions, 1991.

[11] G. Frasca, «Videogames of the oppressed :videogames as a means for critical thinking and debate», 2001. [En línea]. Disponible en: https://api.semanticscholar.org/CorpusID:110396183

[12] S. Belli y C. López, «A brief history of videogames», *Athenea Digital. Revista de pensamiento e investigación social*, n.º 14, p. 159, nov. 2008, doi: 10.5565/rev/athenead/v0n14.570.

[13] R. Rouse, *Game Design: Theory & Practice*. Texas: Plano, 2001.

[14] «Guitar Hero Live». Accedido: 11 de julio de 2024. [En línea]. Disponible en: https://support.activision.com/guitar-hero-live

[15] M. A. Jimenez-Crespo, *Translation and Web Localization*. Routledge, 2013. doi: 10.4324/9780203520208.

[16] C. Mangiron, «Found in Translation: Evolving Approaches for the Localization of Japanese Video Games», *Arts*, vol. 10, n.º 1, p. 9, ene. 2021, doi: 10.3390/arts10010009.

[17] C. Rodriguez del Rosario, «Creación de motores de traducción automática (estadística y neuronal) inglés-español especializados en el campo de la aviación con la herramienta MTUOC», 2021, Accedido: 24 de abril de 2024. [En línea]. Disponible en: http://hdl.handle.net/10609/133768

[18] M. Hearne y A. Way, «Statistical Machine Translation: A Guide for Linguists and Translators», *Lang Linguist Compass*, vol. 5, n.º 5, pp. 205-226, may 2011, doi: 10.1111/j.1749-818X.2011.00274.x.

[19] P. Koehn, *Statistical Machine Translation*. Cambridge University Press, 2009. doi: 10.1017/CBO9780511815829.

[20] M. A. Jiménez-Crespo, «The role of translation technologies in Spanish language learning», *Journal of Spanish Language Teaching*, vol. 4, n.º 2, pp. 181-193, jul. 2017, doi: 10.1080/23247797.2017.1408949.

[21] B. Macukow, «Neural Networks – State of Art, Brief History, Basic Models and Architecture», en *Computer Information Systems and Industrial Management*, W. Saeed Khalid and Homenda, Ed., Cham: Springer International Publishing, 2016, pp. 3-14.

[22] A. Vaswani *et al.*, «Attention is All you Need», en *Advances in Neural Information Processing Systems*, I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, y R. Garnett, Eds., Curran Associates, Inc., 2017. [En línea]. Disponible en: https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf

[23] T. B. Brown *et al.*, «Language Models are Few-Shot Learners», 2020.

[24] O. Briony June, *Researching information systems and computing*. SAGE Publications, 2006.

[25] I. Rivera-Trigueros, «Machine translation systems and quality assessment: a systematic review», *Lang Resour Eval*, vol. 56, n.º 2, pp. 593-619, jun. 2022, doi: 10.1007/s10579-021-09537-5.

[26] L. Moreno Pons, «La traducción automática en la literatura: creación de un corpus paralelo inglés-español para entrenar y evaluar un sistema neuronal en el marco del proyecto MTUOC», 2022, Accedido: 24 de abril de 2024. [En línea]. Disponible en: http://hdl.handle.net/10609/146721

[27] D. Hansen y P.-Y. Houlmont, «A Snapshot into the Possibility of Video Game Machine Translation», *arXiv preprint arXiv:2209.08827*, 2022, doi: https://doi.org/10.48550/arXiv.2209.08827.

[28] M. Shintemirova, «Translation of Metaphors in Official and Automatic Subtitling and MT Evaluation», *Journal of Computational and Applied Linguistics*, vol. 1, pp. 77-93, 2023, doi: https://doi.org/10.33919/JCAL.23.1.4.

[29] R. K. Yin, *Case Study Research: Design and Methods.*, 5th ed. SAGE Publications, Inc., 2014.

[30] F. A. Salazar Fierro, «Framework para la implantación de sistemas gestores de aprendizaje (LMS Learning Management System) en instituciones de educación superior», info:eu-repo/semantics/doctoralThesis, Universidad Nacional Mayor de San Marcos, 2025. Accedido: 18 de mayo de 2025. [En línea]. Disponible en: https://hdl.handle.net/20.500.12672/25817

[31] J. Tiedemann y S. Thottingal, «OPUS-MT – Building open translation services for the World», en *Proceedings of the 22nd Annual Conference of the European Association for Machine Translation*, A. Martins, H. Moniz, S. Fumega, B. Martins, F. Batista, L. Coheur, C. Parra, I. Trancoso, M. Turchi, A. Bisazza, J. Moorkens, A. Guerberof, M. Nurminen, L. Marg, y M. L. Forcada, Eds., Lisboa, Portugal: European Association for Machine Translation, nov. 2020, pp. 479-480. [En línea]. Disponible en: https://aclanthology.org/2020.eamt-1.61/

[32] J. Tiedemann *et al.*, «Democratizing neural machine translation with OPUS-MT», *Lang Resour Eval*, vol. 58, n.º 2, pp. 713-755, jun. 2024, doi: 10.1007/s10579-023-09704-w.

[33] P. Koehn, «Statistical Significance Tests for Machine Translation Evaluation», en *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*, D. Lin y D. Wu, Eds., Barcelona, Spain: Association for Computational Linguistics, jul. 2004, pp. 388-395. [En línea]. Disponible en: https://aclanthology.org/W04-3250/

[34] R. Rei, C. Stewart, A. C. Farinha, y A. Lavie, «COMET: A Neural Framework for MT Evaluation», en *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Stroudsburg, PA, USA: Association for Computational Linguistics, 2020, pp. 2685-2702. doi: 10.18653/v1/2020.emnlp-main.213.

[35] J. Tiedemann y Y. Scherrer, «Neural Machine Translation with Extended Context», en *Proceedings of the Third Workshop on Discourse in Machine Translation*, Stroudsburg, PA, USA: Association for Computational Linguistics, 2017, pp. 82-92. doi: 10.18653/v1/W17-4811.

ANEXOS

Anexo A: Repositorios

Los scripts utilizados en este proyecto para limpieza, preprocesamiento y entrenamiento pueden consultarse a través del repositorio público de MTUOC:

Repositorio:

https://github.com/aoliverg/MTUOC-old

El motor base usado para el fine tuning puede encontrarse dentro de este puede encontrarse dentro del repositorio público de OPUS-MT-TRAIN:

Repositorio:

https://github.com/Helsinki-NLP/OPUS-MT-train

Anexo B: Script en Google Colab para el procesamiento de diálogos

Con el objetivo de preparar los datos necesarios para el entrenamiento del motor de traducción automática, se desarrolló un script en Google Colab que permite cargar y procesar un archivo de texto plano que contiene los diálogos extraídos del videojuego. Este script realiza un preprocesamiento básico para extraer las líneas relevantes y separarlas por idioma, facilitando su posterior uso como corpus paralelo.

El flujo de trabajo implementado consta de los siguientes pasos:

- **Carga del archivo de entrada:** Mediante la interfaz de Google Colab, el usuario puede subir un archivo de texto con los diálogos previamente extraídos. Este archivo debe contener las líneas multilingües separadas por un delimitador especial (ˢˡˣ), que representa las versiones paralelas de cada frase en distintos idiomas, como se muestra en la figura 45.

Figura 45. Subida del archivo

- **Extracción de líneas de diálogo:** El script identifica y recopila únicamente aquellas líneas que contienen el delimitador ␣␣, considerando que estas corresponden a entradas válidas del corpus bilingüe. Cada línea se almacena en una lista para su posterior procesamiento, como se muestra en la figura 46.

Figura 46. Extracción de las líneas de dialogo

- **Almacenamiento de los diálogos completos:** Como paso opcional, se guarda un archivo llamado dialogues_all_languages.txt que contiene todas las líneas extraídas con sus versiones multilingües intactas. Este archivo permite conservar una copia íntegra del corpus original, como se muestra en la figura 47.

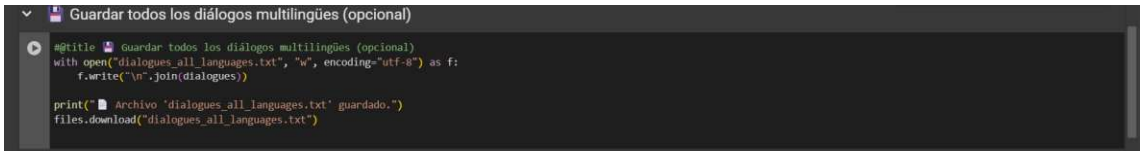


Figura 47. Almacenamiento de diálogos

- **Extracción de los diálogos en inglés:** A partir del corpus completo, se procesan las líneas para obtener exclusivamente la versión en inglés, que se guarda en un archivo denominado dialogues_english_only.txt, como se muestra en la figura 48.

Figura 48. Extracción de diálogos en ingles

- **Extracción de los diálogos en español:** De forma análoga, se extraen las versiones en español de los diálogos. Estas se almacenan en el archivo dialogues_spanish_only.txt, como se muestra en la figura 49.

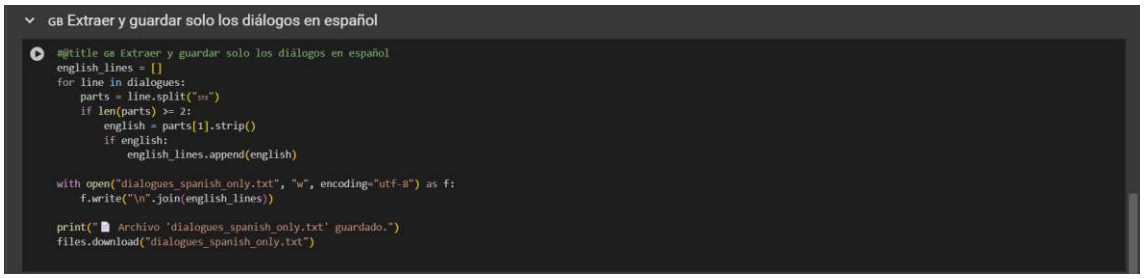


Figura 49. Extracción de diálogos en español

Este procesamiento automatizado permitió construir de forma eficiente un corpus paralelo alineado, indispensable para el entrenamiento del modelo de traducción automática especializado. El uso de Google Colab facilitó la portabilidad y ejecución del script en un entorno accesible sin necesidad de configuración local avanzada.

Anexo C: Script de limpieza y estructuración del corpus paralelo

Tras la extracción inicial de los diálogos en inglés y español, se implementó un segundo script en Google Colab cuyo propósito es preparar y estructurar los datos en un formato adecuado para su utilización en el entrenamiento del motor de traducción automática mediante MTUOC. Este proceso forma parte del preprocesamiento necesario para asegurar la calidad y coherencia del corpus paralelo.

El script realiza las siguientes acciones principales:

- Carga de archivos fuente: Se cargan los archivos de texto dialogues_english_only.txt y dialogues_spanish_only.txt, los cuales contienen las líneas paralelas extraídas y segmentadas previamente. Cada línea corresponde a una oración o fragmento de diálogo en su respectivo idioma, como se muestra en la figura 50.

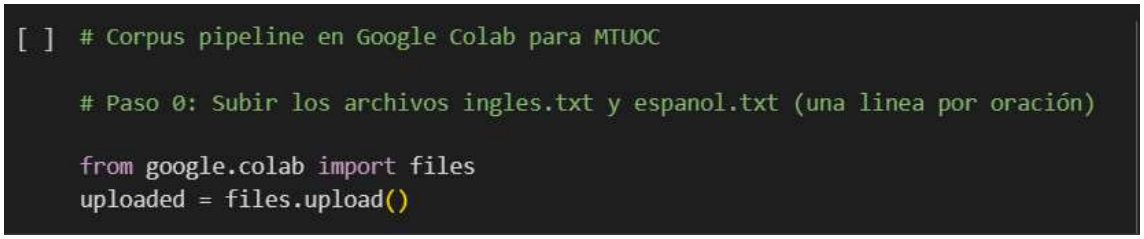
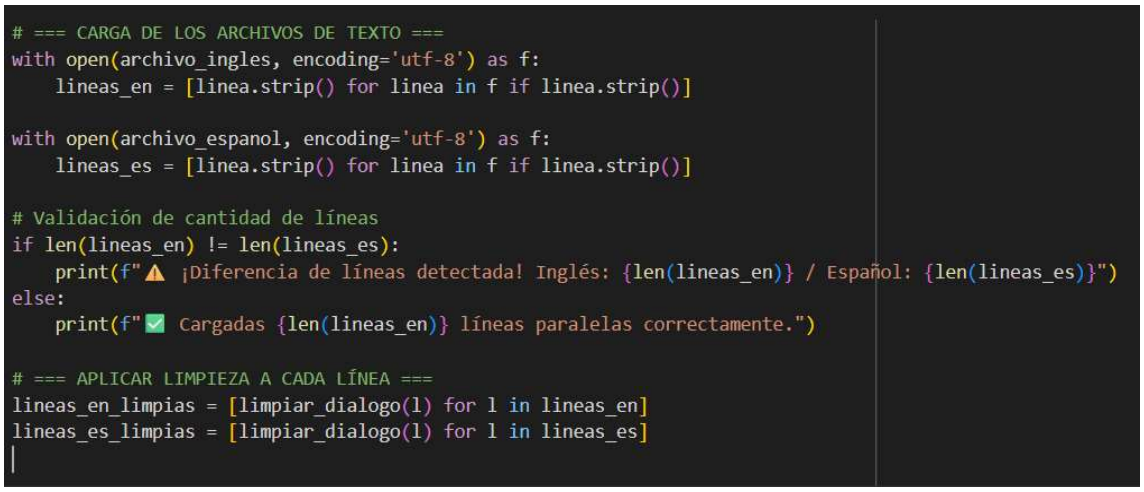


Figura 50. Carga de los archivos en inglés y español

- Limpieza del contenido textual: Se aplica una función de limpieza que utiliza expresiones regulares para eliminar metadatos de los diálogos, como marcadores de hablante del tipo

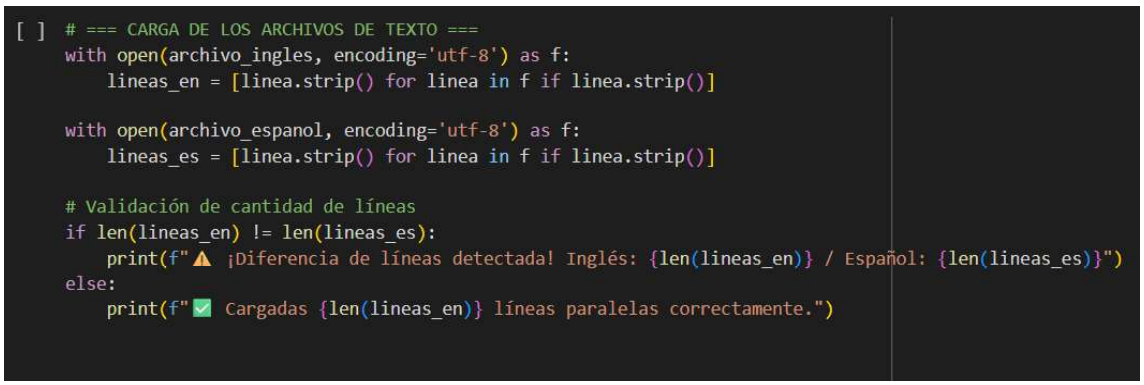
【Nombre】 : , los cuales son comunes en novelas visuales y no aportan valor al proceso de traducción automática. Esto permite obtener únicamente el contenido textual relevante de cada línea, como se muestra en la figura 51.



```
# === CARGA DE LOS ARCHIVOS DE TEXTO ===
with open(archivo_ingles, encoding='utf-8') as f:
    lineas_en = [linea.strip() for linea in f if linea.strip()]

with open(archivo_espanol, encoding='utf-8') as f:
    lineas_es = [linea.strip() for linea in f if linea.strip()]

# Validación de cantidad de líneas
if len(lineas_en) != len(lineas_es):
    print(f"⚠️ ¡Diferencia de líneas detectada! Inglés: {len(lineas_en)} / Español: {len(lineas_es)}")
else:
    print(f"✅ Cargadas {len(lineas_en)} líneas paralelas correctamente.")

# === APLICAR LIMPIEZA A CADA LÍNEA ===
lineas_en_limpias = [limpiar_dialogo(l) for l in lineas_en]
lineas_es_limpias = [limpiar_dialogo(l) for l in lineas_es]
```

Figura 51. Limpieza básica de los diálogos.

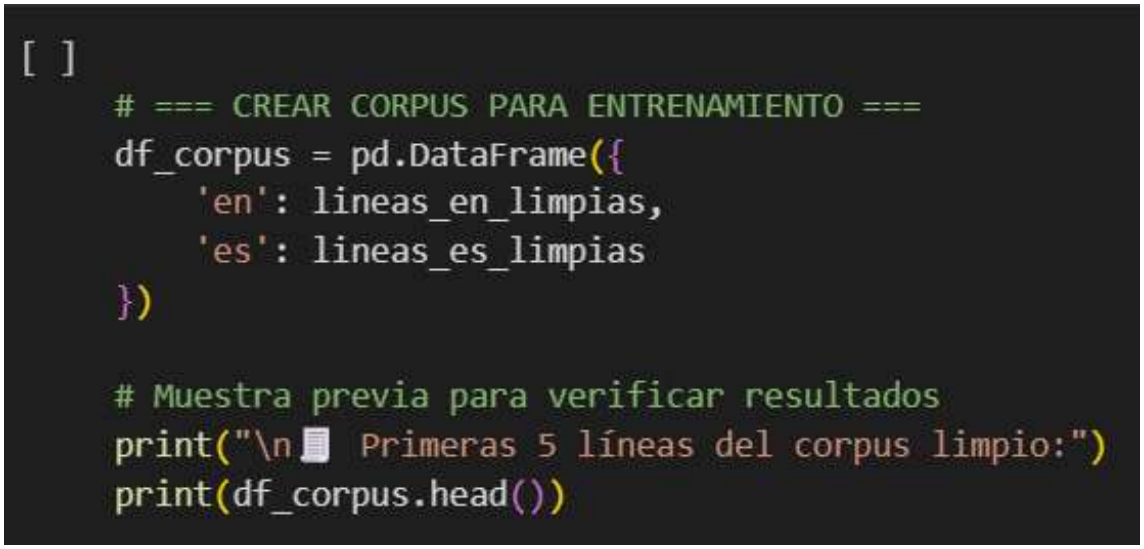
- Validación de alineación: Se verifica que la cantidad de líneas en ambos archivos sea idéntica, garantizando así la correspondencia uno a uno entre oraciones en inglés y en español. Esta validación es esencial para mantener la integridad del corpus paralelo, como se muestra en la figura 52.



```
[ ]  # === CARGA DE LOS ARCHIVOS DE TEXTO ===
     with open(archivo_ingles, encoding='utf-8') as f:
         lineas_en = [linea.strip() for linea in f if linea.strip()]

     with open(archivo_espanol, encoding='utf-8') as f:
         lineas_es = [linea.strip() for linea in f if linea.strip()]

     # Validación de cantidad de líneas
     if len(lineas_en) != len(lineas_es):
         print(f"⚠️ ¡Diferencia de líneas detectada! Inglés: {len(lineas_en)} / Español: {len(lineas_es)}")
     else:
         print(f"✅ Cargadas {len(lineas_en)} líneas paralelas correctamente.")
```

Figura 52. Validación de alineación.

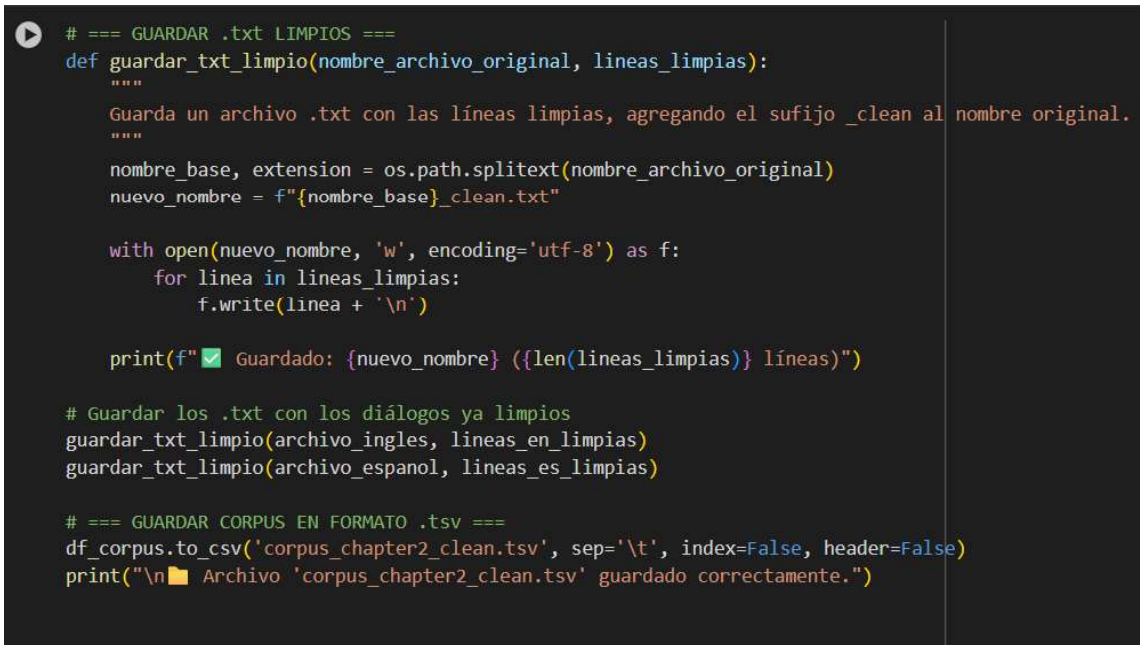
- Construcción del corpus estructurado: Las líneas ya limpias son combinadas en una estructura tabular utilizando la biblioteca pandas, conformando un DataFrame que permite visualizar y manipular los pares de traducción, como se muestra en la figura 53.



```
[ ]
    # === CREAR CORPUS PARA ENTRENAMIENTO ===
    df_corpus = pd.DataFrame({
        'en': lineas_en_limpias,
        'es': lineas_es_limpias
    })

    # Muestra previa para verificar resultados
    print("\n📄 Primeras 5 líneas del corpus limpio:")
    print(df_corpus.head())
```

Figura 53. Creación del corpus

- Exportación de archivos de salida:
    - Se generan dos nuevos archivos .txt con los diálogos ya limpiados, añadiendo el sufijo _clean a los nombres originales. Estos archivos pueden ser utilizados en etapas posteriores del flujo de trabajo de traducción.
    - Adicionalmente, se guarda el corpus completo en un archivo corpus_chapter2_clean.tsv, con formato de valores separados por tabulaciones (TSV), que es compatible con herramientas de entrenamiento como marian-train de MarianNMT, como se muestra en la figura 54.



```
# === GUARDAR .txt LIMPIOS ===
def guardar_txt_limpio(nombre_archivo_original, lineas_limpias):
    """
    Guarda un archivo .txt con las líneas limpias, agregando el sufijo _clean al nombre original.
    """
    nombre_base, extension = os.path.splitext(nombre_archivo_original)
    nuevo_nombre = f"{nombre_base}_clean.txt"

    with open(nuevo_nombre, 'w', encoding='utf-8') as f:
        for linea in lineas_limpias:
            f.write(linea + '\n')

    print(f"✅ Guardado: {nuevo_nombre} ({len(lineas_limpias)} líneas)")

# Guardar los .txt con los diálogos ya limpios
guardar_txt_limpio(archivo_ingles, lineas_en_limpias)
guardar_txt_limpio(archivo_espanol, lineas_es_limpias)

# === GUARDAR CORPUS EN FORMATO .tsv ===
df_corpus.to_csv('corpus_chapter2_clean.tsv', sep='\t', index=False, header=False)
print("\n📁 Archivo 'corpus_chapter2_clean.tsv' guardado correctamente.")
```

Figura 54. Guardado de los archivos de salida.

Este segundo script fue fundamental para garantizar que el corpus usado en el entrenamiento del motor estuviera limpio, alineado y estructurado adecuadamente. Además, su implementación en Google Colab facilitó el procesamiento en línea sin requerimientos técnicos adicionales por parte del entorno local del usuario.