Iñaki Diez Lambies

Universistat politècnica de València | MUIARFID Curso 2023 - 2024

Trabajo sobre el corpus Europarl

Traducción automática

Tabla de contenido

[Introducción 2](#_Toc158320734)

[Metodología 3](#_Toc158320735)

[Preprocesado de datos 3](#_Toc158320736)

[Evaluación de los modelos 5](#_Toc158320737)

[Moses 6](#_Toc158320738)

[Experimentación 6](#_Toc158320739)

[Evaluación 8](#_Toc158320740)

[OpenNMT – Transformer 9](#_Toc158320741)

[Experimentación 9](#_Toc158320742)

[Evaluación 10](#_Toc158320743)

[OpenNMT – LSTMs 11](#_Toc158320744)

[Experimentación 11](#_Toc158320745)

[Evaluación 12](#_Toc158320746)

[MarianMT 13](#_Toc158320747)

[Experimentación 13](#_Toc158320748)

[Evaluación 14](#_Toc158320749)

[Conclusiones 15](#_Toc158320750)

# Introducción

Este trabajo se centra en la exploración y comparación de diferentes metodologías de Traducción Automática (TA) aplicadas a un corpus específico del Parlamento Europeo, conocido como *europarl*, un recurso ampliamente utilizado en el ámbito de la lingüística computacional para el desarrollo y evaluación de sistemas de TA.

El corpus seleccionado para este estudio incluye 50.000 frases de entrenamiento, dividido en dos segmentos para el entrenamiento y ajuste de modelos de traducción mediante el enfoque estadístico de Moses y el enfoque neuronal mediante openNMT, respectivamente. Este conjunto de datos ofrece una oportunidad única para analizar las capacidades y limitaciones de cada enfoque en el contexto de la traducción del inglés al español. Además, se incorpora un conjunto de prueba de 1.000 pares de frases para evaluar la eficacia de los modelos entrenados.

El objetivo principal de este trabajo es construir y optimizar traductores automáticos utilizando dos paradigmas distintos: el modelo estadístico Moses y el modelo neuronal Transformer de openNMT. Ambos enfoques serán evaluados y comparados en términos de calidad de traducción, medido a través de la métrica BLEU, un estándar en la evaluación de la TA que cuantifica la similitud entre la traducción generada por la máquina y una o varias referencias humanas.

Además de los ejercicios básicos, este trabajo se expande para incluir el análisis de modelos avanzados mediante la implementación de redes neuronales recurrentes, específicamente LSTMs (Long Short-Term Memory), así como la exploración del ajuste del modelo preentrenado multilingüe MarianMT. Estas variantes representan un esfuerzo por mejorar la comprensión y generación de traducciones automáticas, reflejando las últimas tendencias y avances tecnológicos en el campo de la TA.

Este trabajo también incluye una propuesta de evaluación de los resultados obtenidos, a través de la comparación directa de las métricas de rendimiento entre los diferentes modelos y configuraciones empleadas. Se discutirán las implicaciones prácticas de los hallazgos, ofreciendo una perspectiva crítica sobre el estado actual de la TA y sus posibles direcciones futuras.

La documentación de este estudio se presenta en una memoria estructurada que abarca desde la introducción del problema y la metodología empleada, hasta la presentación detallada de los resultados obtenidos y su análisis correspondiente.

# Metodología

La metodología adoptada en este estudio es clave en el desarrollo y la eficacia de los sistemas de traducción automática investigados. En este segmento, detallaremos exhaustivamente los procesos automatizados y las técnicas aplicadas en cada etapa de nuestro trabajo, desde el preprocesado de los datos hasta la ejecución de los modelos de traducción. La automatización juega un papel crucial en la optimización de estos procesos, permitiendo una gestión eficiente y sistemática de grandes volúmenes de datos, así como la replicabilidad de los experimentos realizados.

Abordaremos el preprocesamiento de los datos, una fase esencial que prepara el corpus para su uso efectivo en los modelos de traducción, mediante la limpieza, normalización y tokenización de los textos. La ejecución de scripts de entrenamiento específicos para cada modelo, Moses y openNMT de Transformers, así como para las variantes avanzadas con LSTMs y MarianMT, se describirá con detalle, subrayando las configuraciones y ajustes realizados para optimizar el rendimiento de los traductores automáticos.

## Preprocesado de datos

El preprocesado de datos es un paso crítico en la construcción de cualquier sistema de traducción automática, ya que la calidad de los datos de entrada puede influir significativamente en el rendimiento del sistema. En este proyecto, hemos adoptado una serie de herramientas y scripts automatizados para asegurar un preprocesado eficiente y consistente, facilitando así la replicabilidad y la escalabilidad del proceso.

En primer lugar, se utiliza un script en Python que cumple la función de mezclar y particionar el corpus en conjuntos de entrenamiento y validación. La importancia de este proceso radica en su capacidad para reducir el sesgo y mejorar la generalización del modelo. El script asegura que los archivos de texto en español e inglés tengan la misma cantidad de líneas y luego procede a emparejar las líneas correspondientes, barajarlas y dividirlas según el porcentaje deseado, permitiendo una distribución equitativa y aleatoria de los datos. En la Ilustración 1 podemos ver la implementación de este script.

Texto

Descripción generada automáticamente

Ilustración 1 - Implementación de shuffle\_data.py

Seguidamente, el preprocesado de datos implica la limpieza del corpus, para lo cual se utiliza un script que permite utilizar de forma más sencilla las herramientas de Moses para esto, como se muestra en la Ilustración 2. Este script de Shell se encarga de estandarizar la longitud de las frases, descartando aquellas que son demasiado cortas o excesivamente largas según los umbrales definidos (por defecto, de 1 a 60 tokens). Además, ofrece ayuda descriptiva para su uso y permite ajustar dinámicamente los valores mínimos y máximos de longitud de las frases a través de argumentos opcionales, garantizando así la flexibilidad para adaptarse a diferentes requisitos de los conjuntos de datos.

Texto

Descripción generada automáticamente

Ilustración 2 - Implementación de clean\_corpus.sh

Este, junto con otros scripts creados, permiten una fácil reproducibilidad de los resultados, así como experimentos consistentes. En la Ilustración 3 podemos ver algunos de estos. Caben destacar los dos llamados *run\_moses.sh* y *run­\_opennmt.sh* que nos permiten la ejecución de los respectivos contenedores Docker para una mayor encapsulación de nuestras dependencias y de entorno de trabajo.

Texto

Descripción generada automáticamente con confianza baja

Ilustración 3 - Scripts creados para distintas facilidades

Para mayor automatización, podemos encontrar el script mostrado en la Ilustración 4 que orquesta el proceso completo de preprocesado. Este script de shell se encarga de crear los directorios necesarios, copiar los archivos de datos al lugar de trabajo, y llamar a los scripts de tokenización y limpieza sobre los conjuntos de datos apropiados. La tokenización es un paso crucial que prepara el texto para el entrenamiento, segmentándolo en unidades manejables (tokens) que el modelo de traducción puede procesar.

Interfaz de usuario gráfica, Texto

Descripción generada automáticamente

Ilustración 4 - Data run.sh

La automatización de estos pasos a través de scripts es una ventaja considerable en términos de eficiencia y precisión. Elimina el riesgo de error humano en tareas repetitivas y asegura que cada paso se ejecute de manera uniforme, lo que es especialmente beneficioso al manejar grandes volúmenes de datos. Esto no solo ahorra tiempo valioso que de otro modo se gastaría en la preparación manual de los datos, sino que también proporciona una base sólida y confiable para el entrenamiento de los modelos, lo cual es esencial para la consecución de traducciones automáticas de alta calidad. La sistematización del preprocesado aumenta la reproducibilidad de la investigación, un pilar de la metodología científica, ya que otros investigadores pueden replicar exactamente el proceso siguiendo los scripts proporcionados.

## Evaluación de los modelos

La evaluación de modelos de traducción automática es un componente esencial en el desarrollo de estos sistemas, permitiendo medir su calidad y rendimiento. En este trabajo, la evaluación se ha llevado a cabo utilizando las métricas BLEU (Bilingual Evaluation Understudy), específicas para cada tecnología empleada. Se han analizado las traducciones generadas por los modelos en un conjunto de datos de prueba independiente y más reducido que el conjunto de entrenamiento. Dicha independencia es crucial para obtener una evaluación objetiva del modelo, ya que refleja su capacidad para generalizar y aplicar el conocimiento aprendido a nuevos ejemplos que no se han visto durante el entrenamiento.

La metodología de utilizar conjuntos de prueba separados es ampliamente reconocida y se ha convertido en un estándar en la evaluación de sistemas de inteligencia artificial debido a sus múltiples ventajas. Primero, proporciona una valoración imparcial del desempeño del modelo en condiciones similares a su uso en escenarios reales, donde se encontrará con entradas no vistas previamente. Segundo, la métrica BLEU es especialmente valiosa por su capacidad para comparar la traducción de la máquina con una o varias referencias humanas, ofreciendo así una medida cuantitativa del grado de coincidencia.

La principal ventaja de la métrica BLEU es su consistencia y su capacidad para ser automatizada, lo que permite una evaluación rápida y a gran escala de los modelos. Además, al ser un estándar de la industria, facilita la comparación de resultados entre diferentes estudios y sistemas de traducción, permitiendo un avance más colaborativo y medible en el campo. No obstante, es importante reconocer que, aunque BLEU proporciona una evaluación práctica y rápida, no captura todos los matices de la calidad de la traducción, como la fluidez y la coherencia del texto, razón por la cual es recomendable complementarlo con otras evaluaciones cualitativas para una comprensión más completa del rendimiento del modelo.

# Moses

MOSES representa ofrece un sistema capaz de construir modelos de traducción estadística a partir de corpus paralelos para cualquier par de idiomas. Su metodología se basa en la selección de la traducción con la mayor probabilidad posible, empleando para ello un algoritmo de búsqueda. A pesar de que la instalación de MOSES puede ser compleja debido a sus dependencias de librerías externas, la opción de Docker ha facilitado significativamente este proceso, permitiendo a los acceder a las herramientas de MOSES de manera más sencilla y sin preocuparse por las configuraciones del sistema subyacente.

En el siguiente apartado, exploraremos la experimentación realizada con MOSES en este proyecto. Además, se discutirá el proceso de evaluación del modelo a través de la métrica BLEU, de forma que subrayará la efectividad del sistema MOSES en tareas de traducción específicas.

## Experimentación

El proceso de experimentación con el sistema de traducción estadística MOSES se ha llevado a cabo a través de una serie de scripts automatizados, que orquestan desde la generación de modelos de lenguaje hasta su evaluación final.

Texto

Descripción generada automáticamente

Ilustración 5 - Moses run.sh

El script *run.sh*, mostrado en la Ilustración 5, inicia el proceso con la creación del modelo de lenguaje utilizando el script *ngram.sh*. Este paso es crucial para construir un modelo que pueda predecir la probabilidad de secuencias de palabras en el texto de destino.

A continuación, el script procede a entrenar el modelo de traducción con *train\_model.sh* de la Ilustración 6. Aquí, se especifica el número de CPUs para paralelizar y acelerar el proceso de alineamiento con GIZA++. Se elige el método de alineamiento *grow-diag-final-and* y el reordenamiento *msd-bidirectional-fe* para el modelo de traducción, buscando un equilibrio entre precisión y cobertura en la alineación de frases.

Texto

Descripción generada automáticamente

Ilustración 6 - Moses train\_model.sh

El paso de optimización de los pesos se realiza mediante *weights\_mert.sh*, donde se aplica MERT (Minimum Error Rate Training) para ajustar los parámetros del modelo log-lineal de MOSES. La elección de MERT como método de optimización se debe a su eficacia en mejorar la calidad de la traducción ajustando los pesos para minimizar la tasa de error. Se limita el número de iteraciones para controlar el tiempo de entrenamiento y evitar el sobreajuste. Podemos ver su script en la Ilustración 6.

Texto

Descripción generada automáticamente

Ilustración 7 - Moses wrights\_mert.sh

En cada uno de estos pasos, se obtienen resultados intermedios que conforman el modelo de traducción:

1. Modelo de Lenguaje: La salida del script *ngram.sh* es un modelo de lenguaje, en este caso, un archivo *.lm* que contiene las probabilidades de las secuencias de palabras (trigramas, en este contexto).
2. Modelo de Traducción: Con *train\_model.sh*, se genera una tabla de frases y archivos de alineación que constituyen el modelo de traducción, que posteriormente será utilizado por MOSES para realizar la traducción propiamente dicha.
3. Optimización de Pesos: *weights\_mert.sh* produce una versión optimizada del archivo *moses.ini*, que contiene los pesos ajustados del modelo log-lineal.

## Evaluación

Finalmente, el experimento culmina con la traducción de un conjunto de prueba y la evaluación de estas traducciones contra un conjunto de referencia, proporcionando una puntuación BLEU que cuantifica la calidad del modelo. Este flujo de trabajo automatizado asegura que cada etapa del proceso de entrenamiento y evaluación sea reproducible y eficiente, facilitando la experimentación con distintas configuraciones y la comparación objetiva de los resultados. La automatización a través de estos scripts no solo optimiza el tiempo y los recursos, sino que también garantiza que las decisiones metodológicas y de parámetros sean coherentes y rastreables a lo largo de todo el experimento.

La calidad de los resultados obtenidos, como se refleja en la salida de la Ilustración 8, muestra un desempeño respetable en la tarea de traducción automática. El puntaje obtenido BLEU es de 28.64, el cual indica que, aunque hay margen de mejora, el sistema de traducción ha capturado una cantidad significativa de la información y el significado del texto de referencia. Los valores de precisión n-grama (61.0/34.3/22.0/14.6 para 1-grama hasta 4-grama, respectivamente) sugieren que mientras la correspondencia de palabras individuales es bastante alta, la coincidencia disminuye conforme se consideran secuencias de palabras más largas, lo cual es típico en la evaluación de sistemas de traducción automática.

Texto

Descripción generada automáticamente

Ilustración 8 - Moses resultados

El valor BP (Brevity Penalty) de 1.000 y la ratio de longitud obtenida cercana a 1 (1.004) indican que la longitud de las hipótesis de traducción es adecuada y muy cercana a la longitud de las referencias. Esto es positivo, ya que significa que el modelo no está sobre-generando ni sub-generando texto, lo cual podría afectar la legibilidad y coherencia de la traducción.

Al observar el fragmento de traducciones generadas, parece que el modelo ha logrado una estructura gramatical básica competente, aunque podría haber errores de léxico o de precisión semántica que no son evidentes sin el texto de referencia. Por ejemplo, podría haber sutilezas en el uso de términos técnicos o contextuales que no se capturan completamente.

# OpenNMT – Transformer

El avance de la traducción automática ha experimentado una revolución con la introducción de las redes neuronales dinámicas, y en particular, con el modelo Transformer, que se ha establecido como un estándar de oro debido a su eficiencia y capacidad para capturar dependencias complejas en los datos de texto. El uso de la herramienta opennmt-py, una implementación basada en PyTorch del modelo Transformer, permite entrenar sistemas de traducción automática con redes neuronales para cualquier par de idiomas, utilizando como base un corpus paralelo. A diferencia de los sistemas basados en modelos estadísticos como MOSES, opennmt-py se centra en la representación y el procesamiento secuencial de los datos a través de mecanismos de atención que facilitan la captación de relaciones contextuales, sin requerir alineación previa ni filtrado de frases.

En el siguiente punto, abordaremos la experimentación realizada con opennmt-py, describiendo cómo se han ajustado y modificado los parámetros del Transformer para adaptarse a las particularidades del conjunto de datos bilingüe proporcionado. El proceso de entrenamiento, traducción y evaluación subsiguiente del modelo neural, junto con los resultados obtenidos, se detallarán para proporcionar una comprensión completa del desempeño de la tecnología en la tarea de traducción automática. Con un enfoque en la arquitectura de la red, que incluye la configuración de las capas, el tamaño de los vectores de codificación y el número de cabezas de atención, se establecerá un marco para la interpretación de la puntuación BLEU alcanzada y se identificarán posibles vías de mejora.

## Experimentación

El archivo de configuración confeccionado para la experimentación con opennmt-py y el modelo Transformer refleja un conjunto de parámetros cuidadosamente seleccionados, que son indicativos de un sistema avanzado de traducción automática. Entre los parámetros más significativos que podrían contribuir a un buen resultado, destacan los siguientes:

1. Tamaño del Vector de Palabras (*word\_vec\_size*): Configurado a 512, este parámetro define el tamaño del embedding de las palabras, lo que influye directamente en la capacidad del modelo para capturar matices semánticos. Un tamaño más grande permite representaciones más ricas y detalladas de las palabras.
2. Número de Capas (*layers*): Seleccionado a 6, un número mayor de capas puede permitir al modelo aprender relaciones más complejas, aunque también incrementa el riesgo de sobreajuste y el costo computacional.
3. Tamaño de la Capa Feed-Forward (*transformer\_ff*): Con un valor de 2048, este parámetro determina la dimensión de la capa feed-forward dentro de cada bloque Transformer. Una dimensión más grande aquí puede mejorar la capacidad del modelo para representar y transformar las representaciones de las palabras.
4. Número de Cabezas de Atención (*heads*): El modelo utiliza 8 cabezas de atención, lo que permite que el mecanismo de atención se enfoque en diferentes partes de la secuencia de entrada simultáneamente, mejorando así la captación de diferentes tipos de relaciones entre las palabras.
5. Pasos de Calentamiento (*warmup\_steps*): Se han establecido 4000 pasos de calentamiento para el optimizador, lo cual es una parte crucial para ajustar la tasa de aprendizaje durante las primeras etapas del entrenamiento y puede conducir a una convergencia más estable y rápida.
6. Optimizador y Tasa de Aprendizaje: Se utiliza Adam con las betas establecidas en 0.9 y 0.98, y una tasa de aprendizaje inicial de 1. Esto, junto con el método de decaimiento *noam*, está diseñado para ajustar dinámicamente la tasa de aprendizaje en función del número de pasos, lo que es crítico para alcanzar un buen rendimiento en los modelos Transformer.
7. Regularización: El *dropout* de 0.3 y el *label\_smoothing* de 0.1 son técnicas de regularización que ayudan a prevenir el sobreajuste al suavizar la distribución de probabilidad de las etiquetas y al introducir ruido en las activaciones durante el entrenamiento, respectivamente.
8. Tamaño del Lote (*batch\_size* y *batch\_type*): Con un tamaño de lote de 4096 tokens y la normalización por tokens, el modelo maximiza la utilización de la memoria de la GPU y permite que se entrenen secuencias de diferentes longitudes de manera eficiente. Estos valores son bastante limitadores debido a el hardware disponible.
9. Pasos de Entrenamiento y Validación: El modelo se entrena por 2500 pasos y se valida cada 500 pasos, lo cual es un buen balance entre obtener un modelo bien entrenado y evitar entrenar más de lo necesario, lo que podría conducir a sobreajuste.

Estos parámetros reflejan una configuración robusta para el entrenamiento del modelo Transformer. La selección de estos valores intenta búsqueda de un equilibrio entre la complejidad del modelo y la eficiencia computacional, con el objetivo de alcanzar un alto nivel de precisión en la traducción.

## Evaluación

En la evaluación de los resultados de la experimentación con el modelo Transformer de OpenNMT ha dado un BLEU con un puntaje BLEU de 22.3. Este resultado, aunque no alcanza los estándares más altos como los que se podrían esperar de modelos de última generación o de sistemas de traducción finamente ajustados, todavía refleja una capacidad competente del modelo para generar traducciones coherentes y relevantes con respecto al texto fuente.

La desglosación del puntaje BLEU (56.4/29.5/17.8/11.2 para 1-grama hasta 4-grama respectivamente) muestra que mientras la precisión de unigramas es razonablemente alta, hay una caída significativa en la precisión de secuencias más largas. Esto podría sugerir que, aunque el modelo capta bien las palabras individuales y algunas expresiones comunes, tiene dificultades con estructuras sintácticas más complejas o con matices semánticos que solo emergen en frases más largas.

El BP (Brevity Penalty) y la ratio indican que la longitud de las traducciones generadas es ligeramente más corta que la de las referencias, lo que puede señalar una tendencia del modelo a generar respuestas más condensadas que podrían estar omitiendo algunos detalles o matices del texto original.

# OpenNMT – LSTMs

Las Redes Neuronales Recurrentes (RNN) con unidades LSTM (Long Short-Term Memory) representan una de las primeras arquitecturas en el campo del procesamiento del lenguaje natural y la traducción automática antes de la llegada de los modelos basados en Transformer. A diferencia de los Transformers, que utilizan mecanismos de atención para procesar secuencias de entrada en paralelo, las LSTMs abordan las secuencias de forma serial, lo que les permite capturar dependencias temporales a largo plazo de manera intrínseca y efectiva.

La incorporación de LSTMs ofrece potenciales ventajas en ciertos escenarios. Por su naturaleza, las LSTMs son particularmente adeptas en tareas donde el contexto y el orden secuencial son cruciales, pudiendo ser preferibles para corpus con estructuras lingüísticas complejas o para idiomas con alta dependencia contextual. Además, las LSTMs pueden ser más eficientes en términos de memoria y tiempo de entrenamiento cuando se trabaja con secuencias cortas, lo que las hace adecuadas para aplicaciones con recursos limitados.

En el siguiente apartado, exploraremos la experimentación llevada a cabo con las LSTMs en el marco de opennmt, detallando cómo la naturaleza inherente de estas redes se aplica a la traducción automática. Examinaremos las configuraciones específicas utilizadas, cómo estas se comparan con los resultados obtenidos de los modelos Transformer, y en qué circunstancias las LSTMs podrían ser comparables a sus contrapartes más modernas.

## Experimentación

La experimentación con LSTMs en opennmt se basa en una configuración diferente a la del modelo Transformer, ajustando los parámetros para aprovechar las características específicas de las redes neuronales recurrentes. Analicemos algunos de los parámetros clave y su posible impacto en los resultados de la experimentación:

1. Tamaño del Vector de Palabras (*word\_vec\_size*): Con un tamaño de 64, este valor es significativamente menor en comparación con el utilizado para el Transformer. El modelo se centra en una representación más compacta, lo cual es adecuado para LSTMs al manejar secuencias de datos y dependencias a largo plazo.
2. Tamaño y Tipo de RNN (*rnn\_size*, *rnn\_type*): El tamaño de la RNN se mantiene en 512, lo que sugiere una capacidad considerable para el almacenamiento y procesamiento de la información a través de las secuencias. La elección de LSTM como tipo de RNN es crucial para superar los problemas de desvanecimiento del gradiente que suelen presentarse en secuencias largas.
3. Número de Capas (*layers*, *encoder\_layers*, *decoder\_layers*): La configuración de dos capas para el codificador y el decodificador puede indicar un enfoque en la eficiencia computacional, evitando modelos demasiado complejos que pueden ser propensos al sobreajuste y al aumento del tiempo de entrenamiento.
4. Bidireccionalidad (*brnn*): Aunque no se especifica un valor, la inclusión de este parámetro hace que el modelo LSTM procese la información tanto en la dirección hacia adelante como hacia atrás, proporcionando un contexto más rico.
5. Regularización (*dropout*): Se utiliza un dropout de 0.2 para mitigar el sobreajuste, lo que es ligeramente menor que en el Transformer.
6. Optimizador y Tasa de Aprendizaje: Se usa Adam con una tasa de aprendizaje de 0.001, más baja que en el Transformer, lo que refleja una aproximación más conservadora para el ajuste de los pesos durante el entrenamiento.
7. Tamaño del Lote y Acumulación (*batch\_size*, *accum\_count*): Un tamaño de lote de 64 y una acumulación de 4 es una estrategia para manejar la memoria y el cálculo de manera efectiva, permitiendo que el modelo actualice los pesos con más frecuencia y con suerte converja más rápido.
8. Pasos de Entrenamiento y Validación (*train\_steps*, *valid\_steps*): Con 5000 pasos de entrenamiento y validación cada 500 pasos, este enfoque podría ser adecuado para asegurar que el modelo LSTM tenga suficiente exposición a los datos para aprender las dependencias complejas del lenguaje.

En resumen, la configuración seleccionada para la LSTM está orientada a un entrenamiento más largo y, con suerte, más estable, aún formando una estructura de red más simple en comparación con el Transformer.

## Evaluación

Evaluando la salida del modelo de traducción automática basado en LSTMs de OpenNMT, que podemos ver el análisis BLEU resultante donde se ha obtenido una puntuación de 22.6. Esta métrica, aunque poco mayor a la puntuación obtenida con el modelo Transformer, sigue siendo un indicador de una calidad de traducción razonable. El detalle de la puntuación BLEU (55.1/28.9/17.6/11.2 para 1-grama hasta 4-grama respectivamente) sugiere una competencia decente en la correspondencia de unigramas, pero revela una disminución esperada en la coincidencia a medida que se consideran n-gramas más largos.

El BP (Brevity Penalty) cercano a 1 y la ratio casi igual a 1 sugieren que la longitud de las hipótesis de traducción generadas está en buena correlación con la longitud de las referencias, lo cual es un buen indicador de que el modelo no está produciendo traducciones excesivamente cortas o largas.

Texto

Descripción generada automáticamente

Ilustración 9 - LSTMs resultados

Revisando las traducciones obtenidas que se aprecian en la Ilustración 9, se observa que el modelo ha capturado la esencia general del texto de origen, manteniendo la coherencia y la estructura gramatical. Sin embargo, es probable que haya ciertas imprecisiones léxicas o semánticas, reflejadas en la disminución de las puntuaciones de n-gramas más largos.

# MarianMT

El fine-tuning de modelos preentrenados se ha consolidado como una técnica fundamental en el campo del procesamiento del lenguaje natural y la traducción automática. Esta estrategia consiste en tomar un modelo que ya ha sido entrenado en un vasto corpus de datos, generalmente en una tarea general de lenguaje, y posteriormente ajustarlo a un conjunto de datos o tarea específica. El fine-tuning permite a los investigadores y practicantes aprovechar el conocimiento general del lenguaje que el modelo ha adquirido, como la estructura de la lengua y las relaciones contextuales, y adaptarlo a necesidades más concretas, lo que puede resultar en una mejora significativa del rendimiento con menos datos de entrenamiento y en menos tiempo que entrenar un modelo desde cero.

MarianMT es un ejemplo notable de un modelo de traducción automática preentrenado que ha demostrado ser eficaz para una variedad de pares de lenguas. Es conocido por su eficiencia y por la calidad de las traducciones que genera, haciéndolo un candidato ideal para el fine-tuning en tareas específicas de traducción. Los modelos como MarianMT son especialmente valiosos en escenarios donde se dispone de un corpus limitado para el entrenamiento, permitiendo a los usuarios afinar el modelo preentrenado con sus propios datos para mejorar la precisión en dominios específicos o estilos de texto particulares.

En el próximo punto, exploraremos cómo se ha aplicado el fine-tuning al modelo MarianMT para adaptarlo a un corpus de traducción específico. Analizaremos los ajustes realizados y el impacto que han tenido en la eficacia del modelo, proporcionando una comprensión más profunda de cómo las técnicas de fine-tuning pueden ser empleadas para mejorar la traducción automática en aplicaciones prácticas.

## Experimentación

La experimentación con el modelo preentrenado MarianMT, como se detalla en los scripts creados, implica un proceso de fine-tuning cuidadosamente configurado para adaptar el modelo a un corpus específico de traducción. Vamos a desglosar los parámetros clave encontrados en el script de entrenamiento y su impacto esperado en el proceso de fine-tuning:

1. Longitud de los Tokens (*max\_input\_length* y *max\_target\_length*): Establecidos en 64, estos parámetros delimitan la longitud máxima de las secuencias de entrada y salida, lo que influye en la capacidad del modelo para manejar frases de diferentes longitudes y puede afectar el rendimiento en oraciones más largas.
2. Optimizador y Tasa de Aprendizaje (*learning\_rate* y *optim*): Se utiliza una tasa de aprendizaje de 2.5e-5 con el optimizador Adam, lo cual es típico en el fine-tuning de modelos preentrenados para realizar ajustes finos y evitar la pérdida de los conocimientos previos del modelo.
3. Decaimiento del Peso (*weight\_decay*): El decaimiento del peso se establece en 0.01, lo cual ayuda a regularizar los pesos del modelo y puede prevenir el sobreajuste durante el proceso de fine-tuning.
4. Épocas de Entrenamiento (*num\_train\_epochs*): Se realiza el fine-tuning durante una sola época, lo que sugiere una adaptación rápida del modelo preentrenado al nuevo corpus, confiando en que el conocimiento previo almacenado en el modelo es suficientemente generalizable.
5. Generación de Predicciones (*predict\_with\_generate*): Este parámetro habilita la generación de traducciones durante la evaluación, lo que permite medir la calidad del modelo en condiciones similares a su uso en producción.

Los parámetros detallados en el script de fine-tuning apunta a una transición efectiva del conocimiento del modelo desde un entrenamiento general a uno específico del dominio sin comprometer los aprendizajes subyacentes, lo que se espera que resulte en traducciones de alta calidad ajustadas al corpus de destino.

## Evaluación

Evaluando los resultados obtenidos del proceso de fine-tuning del modelo MarianMT se ha obtenido un BLEU score reportado de aproximadamente 42 es indicativo de un rendimiento bastante alto, especialmente en el contexto de sistemas de traducción automática.

La traducción generada reflejada en las muestras de la Ilustración 10 sugiere que el modelo ha logrado una comprensión y una reproducción efectiva de las estructuras sintácticas y semánticas del texto de origen. La fluidez y la coherencia de las frases en el idioma objetivo indican que el modelo ha capturado con éxito los patrones lingüísticos necesarios para producir traducciones naturales y precisas. Este nivel de rendimiento es un testamento al poder de los modelos preentrenados y su capacidad para adaptarse eficazmente a través del fine-tuning a conjuntos de datos específicos.

Imagen que contiene Texto

Descripción generada automáticamente

Ilustración 10 - MarianMT resultados

Un BLEU score de más de 40 es considerablemente alto en la mayoría de los contextos de traducción automática, lo que sugiere que las traducciones no solo son gramaticalmente correctas, sino que también conservan un grado significativo de significado del texto original. Sin embargo, es importante recordar que la métrica BLEU, aunque útil, no captura todos los aspectos de la calidad de la traducción, como la adecuación y la fluidez en el nivel de la oración completa, y debe ser complementada con evaluaciones cualitativas y revisiones humanas para una valoración completa.

# Conclusiones

La investigación realizada proporciona una visión amplia sobre las tecnologías de traducción automática. Gracias a los resultados obtenidos, podemos ver en la gráfica de la Ilustración 11 muestra una comparativa visual de los puntajes BLEU obtenidos por diferentes enfoques de traducción automática.

Ilustración 11 - Comparativa resultados

De acuerdo con la gráfica, MarianMT supera significativamente a los demás modelos con un puntaje BLEU considerablemente más alto, lo que indica que las traducciones generadas por este modelo están más cerca de las traducciones de referencia. Esto subraya la efectividad del fine-tuning de modelos preentrenados, que pueden aprovechar el conocimiento de lenguaje general para adaptarse rápidamente a tareas específicas.

Por otro lado, el modelo Transformer y LSTM muestran resultados similares entre sí, aunque ambos están por detrás del modelo MarianMT. Estos resultados sugieren que, mientras que las arquitecturas basadas en redes neuronales ofrecen una mejora significativa sobre los métodos estadísticos tradicionales, el fine-tuning de modelos preentrenados puede ofrecer una ventaja adicional en términos de calidad de traducción.

MOSES, aunque es el que tiene el puntaje más bajo, aún ofrece resultados respetables. Su rendimiento más bajo en comparación con los enfoques basados en redes neuronales podría atribuirse a su incapacidad para modelar directamente relaciones complejas en los datos. Sin embargo, su tecnología sigue siendo relevante, especialmente en situaciones donde los recursos computacionales son limitados o cuando se trabaja con pares de idiomas para los que no existen grandes corpus paralelos.

El análisis comprensivo de los resultados nos ofrece varias lecciones:

1. Los modelos preentrenados, como MarianMT, ofrecen una excelente base para el fine-tuning, lo cual puede resultar en un rendimiento superior cuando se ajustan a un corpus específico.
2. Los modelos Transformer y LSTM proporcionan una sólida base para la traducción automática con redes neuronales, pero pueden requerir más datos y tiempo de entrenamiento para igualar a los modelos preentrenados ajustados.
3. Los métodos estadísticos aún tienen valor, particularmente en contextos de recursos limitados o en pares de idiomas menos comunes.

En conclusión, el trabajo realizado demuestra la viabilidad y la eficacia de múltiples enfoques en la traducción automática. La elección entre estos métodos dependerá de los recursos disponibles, los requisitos específicos del idioma y la tarea, y los objetivos finales del sistema de traducción.