



INSTITUTO POLITÉCNICO NACIONAL

---

---

CENTRO DE INVESTIGACIÓN EN COMPUTACIÓN

T E S I S

Interpretabilidad del modelo BERT en el contexto de  
la similitud semántica

QUE PARA OBTENER EL GRADO DE:

MAESTRÍA EN CIENCIAS DE LA COMPUTACIÓN

PRESENTA:

Ing. Diana Anahí Ledesma Roque

DIRECTORES DE TESIS:

Dra. Olga Kolesnikova

Dr. Ricardo Menchaca Méndez



Ciudad de México

Enero 2024



**INSTITUTO POLITÉCNICO NACIONAL**  
**SECRETARIA DE INVESTIGACIÓN Y POSGRADO**

SIP-13  
REP 2017

**ACTA DE REGISTRO DE TEMA DE TESIS  
Y DESIGNACIÓN DE DIRECTOR DE TESIS**

Ciudad de México, a **23** de **noviembre** del **2023**

El Colegio de Profesores de Posgrado del **Centro de Investigación en Computación** en su Sesión

(Unidad Académica)

Extraordinaria No **18** celebrada el día **21** del mes **noviembre** de **2023**, conoció la solicitud presentada por el (la) alumno (a):

Apellido Paterno:	LEDESMA	Apellido Materno:	ROQUE	Nombre (s):	DIANA ANAHÍ
-------------------	---------	-------------------	-------	-------------	-------------

Número de registro: **A 2 2 0 5 6 9**

del Programa **Maestría en Ciencias de la Computación**

Referente al registro de su tema de tesis; acordando lo siguiente:

1.- Se designa al aspirante el tema de tesis titulado:

***"Interpretabilidad del modelo BERT en el contexto de la similitud semántica "***

Objetivo general del trabajo de tesis:

Brindar interpretabilidad lingüística de los componentes principales del modelo transformador BERT en el contexto de la similitud semántica

2.- Se designa como Directores de Tesis a los profesores:

Director: **Dra. Olga Kolesnikova** 2º Director: **Dr. Ricardo Menchaca Méndez**

No aplica:

3.- El Trabajo de investigación base para el desarrollo de la tesis será elaborado por el alumno en:

**Centro de Investigación en Computación**

que cuenta con los recursos e infraestructura necesarios.

4.- El interesado deberá asistir a los seminarios desarrollados en el área de adscripción del trabajo desde la fecha en que se suscribe la presente, hasta la aprobación de la versión completa de la tesis por parte de la Comisión Revisora correspondiente.

Director(a) de Tesis

**Dra. Olga Kolesnikova**

2º Director de Tesis

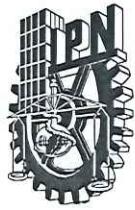
**Dr. Ricardo Menchaca Méndez**

Aspirante

**C. Diana Anahí Ledesma Roque**

Presidente del Colegio

**INSTITUTO POLITÉCNICO NACIONAL  
CENTRO DE INVESTIGACIÓN  
EN COMPUTACIÓN  
DIRECCIÓN  
IPN-CIC**



INSTITUTO POLITÉCNICO NACIONAL  
SECRETARÍA DE INVESTIGACIÓN Y POSGRADO

SIP-14  
REP 2017

ACTA DE REVISIÓN DE TESIS

En la Ciudad de **México** siendo las **12:00** horas del día **08** del mes de **diciembre**  
del **2023** se reunieron los **miembros de la Comisión Revisora de la Tesis**, designada por el Colegio de  
Profesores de Posgrado de: **Centro de Investigación en Computación** para examinar la tesis titulada:  
**"Interpretabilidad del modelo BERT en el contexto de la similitud semántica "** del (la) alumno (a):

Apellido Paterno:	LEDESMA	Apellido Materno:	ROQUE	Nombre (s):	DIANA ANAHÍ
-------------------	---------	-------------------	-------	-------------	-------------

Número de registro: **A 2 2 0 5 6 9**

Aspirante del Programa Académico de Posgrado: **Maestría en Ciencias de la Computación**

Una vez que se realizó un análisis de similitud de texto, utilizando el software antiplagio, se encontró que el trabajo de tesis tiene **05** % de similitud. **Se adjunta reporte de software utilizado.**

Después que esta Comisión revisó exhaustivamente el contenido, estructura, intención y ubicación de los textos de la tesis identificados como coincidentes con otros documentos, concluyó que en el presente trabajo **SI  NO  SE CONSTITUYE UN POSIBLE PLAGIO.**

**JUSTIFICACIÓN DE LA CONCLUSIÓN:** *(Por ejemplo, el % de similitud se localiza en metodologías adecuadamente referidas a fuente original)*  
**El 5% de similitud mostrado por el Turnitin se localiza en bases de datos de acceso abierto y a la metodología del estado del arte adecuadamente referenciadas a los trabajos originales.**

**\*\*Es responsabilidad del alumno como autor de la tesis la verificación antiplagio, y del Director o Directores de tesis el análisis del % de similitud para establecer el riesgo o la existencia de un posible plagio.**

Finalmente y posterior a la lectura, revisión individual, así como el análisis e intercambio de opiniones, los miembros de la Comisión manifestaron **APROBAR  SUSPENDER  NO APROBAR**  la tesis por **UNANIMIDAD  o MAYORÍA**  en virtud de los motivos siguientes:

Cumple con los requisitos reglamentarios.

COMISIÓN REVISORA DE TESIS

Dra. Olga Kolesnikova  
Director de Tesis

Dr. Grigori Sidorov

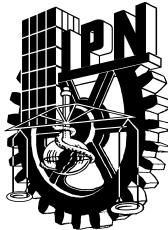
Dr. Rolando Menchaca Méndez

Dr. Ricardo Menchaca Méndez  
2º Director de Tesis

Dr. Ildar Batyrshin

Dr. Erik Zamora Gómez

INSTITUTO POLITÉCNICO NACIONAL  
CENTRO DE INVESTIGACIÓN  
EN COMPUTACIÓN  
Dr. Francisco Hiram Calvo Castro  
PRESIDENTE DEL COLEGIO DE  
PROFESORES



**INSTITUTO POLITÉCNICO NACIONAL**  
**SECRETARÍA DE INVESTIGACIÓN Y POSGRADO**

**CARTA DE AUTORIZACIÓN DE USO DE OBRA PARA DIFUSIÓN**

En la Ciudad de México el día 20 del mes de diciembre del año 2023, la que suscribe Diana Anahí Ledesma Roque alumna del programa Maestría en Ciencias de la Computación con número de registro A220569, adscrito al Centro de Investigación en Computación manifiesta que es autora intelectual del presente trabajo de tesis bajo la dirección de la Dra. Olga Kolesnikova y el Dr. Ricardo Menchaca Méndez y cede los derechos del trabajo intitulado "Interpretabilidad del modelo BERT en el contexto de la similitud semántica", al Instituto Politécnico Nacional, para su difusión con fines académicos y de investigación.

Los usuarios de la información no deben reproducir el contenido textual, gráficas o datos del trabajo sin el permiso expresado del autor y/o director(es). Este puede ser obtenido escribiendo a la siguiente dirección de correo: amsedel91@gmail.com. Si el permiso se otorga, el usuario deberá dar el agradecimiento correspondiente y citar la fuente de este.

---

Diana Anahí Ledesma Roque

## Resumen

Se propusieron varios métodos para abstraer la salida del modelo transformador BERT base al abordar la tarea de similitud semántica en pares de oraciones mediante un regresor lineal. Se exploró una red recurrente LSTM y métodos de agregación, tanto promedio como máximo, con el objetivo de mejorar las técnicas de abstracción del token de clasificación CLS. Sin embargo, los resultados indicaron que ninguno de los métodos propuestos logró superar la calidad de abstracción del token CLS.

Además, se abordó el desafío de la interpretabilidad lingüística del modelo BERT mediante el análisis de sus autoatenciones. La comunidad científica ha explorado la interpretabilidad de BERT mediante enfoques de aprendizaje supervisado, obteniendo conclusiones diversas. En este sentido, este trabajo se enfocó en explorar la interpretabilidad del modelo mediante el enfoque del aprendizaje no supervisado. Dicho análisis se llevó a cabo al enfrentar la tarea de similitud semántica en pares de oraciones en el idioma inglés, considerando el ajuste fino del modelo mediante el token CLS.

Para el análisis de las matrices de autoatención del modelo, estas fueron reducidas dimensionalmente y sometidas a un análisis exploratorio mediante algoritmos de agrupamiento. Se examinaron tres características lingüísticas: longitud de secuencia, similitud de estructuras gramaticales y similitud semántica. Los hallazgos destacan la relevancia de la longitud de la secuencia en las capas iniciales del modelo, mientras que la atención a la similitud semántica se concentra en las capas intermedias a superiores. Además, se observó una tendencia de agrupamiento de secuencias con estructuras gramaticales similares a lo largo de las capas del modelo.

## Abstract

Several methods were proposed to abstract the output of the base BERT transformer model when addressing the task of semantic similarity in pairs of sentences using a linear regressor. A recurrent LSTM network and aggregation methods, both average and max pooling, were explored to enhance the abstraction quality of the CLS classification token. However, the results indicated that none of the proposed methods succeeded in surpassing the abstraction quality of the CLS token.

Additionally, the challenge of linguistic interpretability of the BERT model was addressed through the analysis of its self-attention mechanisms. The scientific community has explored the interpretability of BERT through supervised learning approaches, yielding diverse conclusions. In this regard, this work focused on exploring the interpretability of the model using the unsupervised learning approach. This analysis was conducted by tackling the task of semantic similarity in pairs of English sentences, considering fine-tuning the model through the CLS token.

For the analysis of the model’s attention matrices, these were dimensionally reduced and subjected to exploratory analysis using clustering algorithms. Three linguistic features were examined: sequence length, similarity of grammatical structures, and semantic similarity. The findings highlight the relevance of sequence length in the initial layers of the model, while attention to semantic similarity is concentrated in the intermediate to upper layers. Moreover, a tendency of grouping sequences with similar grammatical structures across the model’s layers was observed.

# Agradecimientos

Agradezco profundamente a mis padres, quienes con su sacrificio han trazado el camino de mi educación y han inculcado en mí valores fundamentales, los cuales han sido la piedra angular de mi carácter. Su amor incondicional ha sido mi mayor apoyo.

Gracias a Diego, mi compañero de vida y mejor amigo, por su paciencia, comprensión, su amor y sus consejos. Eres mi inspiración y la persona que me alienta y me impulsa en todo momento a superarme y ser un mejor ser humano.

Muchas gracias a mis directores de tesis, la Dra. Olga Kolesnikova y el Dr. Ricardo Menchaca Méndez. Han sido no solo excelentes maestros, sino también seres humanos excepcionales. Aprecio enormemente su guía en este exigente trayecto, así como sus valiosos consejos, tiempo y atención. Gracias por impulsarme a ir más allá, por contagiarme su entusiasmo por la investigación. Gracias por creer en mí incluso antes de que yo misma lo hiciera, porque debido a ello no me quedó más remedio que comenzar en creer en mi capacidad.

Expreso mi sincero agradecimiento a los miembros de mi jurado por sus observaciones, por señalar puntos de mejora, por su aliento constante durante la realización de este trabajo, por su entusiasmo y, sobre todo, por su comprensión.

Además, quiero expresar mi gratitud hacia mis profesores en el Centro de Investigación en Computación, quienes no solo comparten sus conocimientos en el día a día, sino que también son fuente de inspiración al brindar un esfuerzo adicional por amor a su vocación.

Mi reconocimiento a mi casa de estudios, el Instituto Politécnico Nacional, así como a todas las personas que, con su dedicación diaria, desempeñan diversos roles para mantener en funcionamiento el Centro de Investigación en Computación (CIC). Además, agradezco al CONAHCYT y al pueblo mexicano, cuyos impuestos han posibilitado el desarrollo de este trabajo mediante el otorgamiento de una beca.

Finalmente, deseo que cada persona que se dedique a la investigación encuentre entusiasmo, curiosidad, pasión, asombro y autodescubrimiento en su camino. Considero que estos aspectos son fundamentales, sobre cualquier tipo de grado académico.

# Índice general

<b>Índice de figuras</b>	<b>11</b>
<b>1. Introducción</b>	<b>13</b>
1.1. Definición del problema . . . . .	13
1.2. Motivación . . . . .	14
1.3. Hipótesis . . . . .	15
1.4. Justificación . . . . .	15
1.5. Objetivos . . . . .	16
1.5.1. General . . . . .	16
1.5.2. Particulares . . . . .	16
1.6. Contribución del problema . . . . .	17
1.7. Organización de la tesis . . . . .	17
<b>2. Marco teórico</b>	<b>19</b>
2.1. Problemas del lenguaje . . . . .	19
2.1.1. Ambigüedad . . . . .	19
2.1.2. Diversidad . . . . .	20
2.2. Regresión lineal . . . . .	20
2.3. Agrupamiento . . . . .	22
2.3.1. K-means . . . . .	23
2.3.2. Jerárquico . . . . .	24
2.3.3. DBSCAN . . . . .	26
2.3.4. Espectral . . . . .	27
2.3.5. Mezcla Gausiana . . . . .	29
2.4. Reducción dimensional . . . . .	30
2.4.1. Autocodificador . . . . .	30
2.5. Evaluación de agrupamientos . . . . .	31
2.5.1. Métricas extrínsecas . . . . .	31

2.5.2. Métricas intrínsecas . . . . .	32
2.6. Pruebas estadísticas para análisis de agrupamientos . . . . .	34
2.6.1. Prueba Mann-Whitney U . . . . .	35
2.6.2. Prueba Kruskal-Wallis . . . . .	35
2.7. Arquitectura del transformador . . . . .	36
2.7.1. Bloque codificador . . . . .	37
2.7.2. Bloque decodificador . . . . .	38
2.7.3. Atención de múltiples cabezales . . . . .	39
2.7.4. Normalización . . . . .	41
2.7.5. Red completamente conectada basada en posición . . . . .	42
2.7.6. Incrustaciones de entrada . . . . .	42
2.7.7. Codificación posicional . . . . .	43
2.7.8. Capa final . . . . .	43
2.7.9. Complejidad computacional de la autoatención del transformador .	43
2.8. BERT . . . . .	44
2.8.1. Arquitectura del modelo BERT . . . . .	45
2.8.2. Pre-entrenamiento del modelo BERT . . . . .	45
2.8.3. Ajuste fino del modelo BERT . . . . .	46
2.8.4. Antecedentes del modelo BERT . . . . .	47
<b>3. Estado del arte</b>	<b>50</b>
3.1. Similitud semántica a través del token CLS . . . . .	50
3.2. Interpretabilidad del modelo BERT a través de poda . . . . .	51
3.3. Interpretabilidad de aspectos lingüísticos en el modelo BERT . . . . .	52
<b>4. Metodología</b>	<b>59</b>
4.1. Recursos de <i>hardware</i> y <i>software</i> . . . . .	59
4.2. Conjuntos de datos . . . . .	60
4.2.1. STS-Benchmark . . . . .	60
4.2.2. SICK-R . . . . .	62
4.3. Similitud semántica a través del token CLS y otros métodos de abstracción.	62
4.3.1. Preprocesamiento del corpus . . . . .	63
4.3.2. Entrenamiento . . . . .	63
4.3.3. Validación y optimización de hiperparámetros . . . . .	64
4.3.4. Medición de desempeño . . . . .	65
4.4. Análisis de autoatenciones . . . . .	67
4.4.1. Manejo de las representaciones de atención . . . . .	68
4.4.2. Reducción de dimensionalidad mediante autocodificador . . . . .	69

4.4.3. Algoritmos de agrupamiento y evaluación . . . . .	70
4.4.4. Interpretación de aspectos lingüísticos . . . . .	72
<b>5. Resultados experimentales</b>	<b>76</b>
5.1. Similitud semántica mediante el token CLS y métodos propuestos . . . . .	76
5.1.1. Análisis y discusión sobre métodos propuestos y token CLS . . . . .	76
5.2. Interpretabilidad lingüística en autoatenciones . . . . .	78
5.2.1. Resultados de reducción dimensional y agrupación . . . . .	79
5.2.2. Análisis de longitud de secuencia (LS) . . . . .	84
5.2.3. Análisis de similitud semántica (SS) . . . . .	88
5.2.4. Análisis de similitud de estructura gramatical (SEG) . . . . .	93
<b>6. Conclusiones y trabajo futuro</b>	<b>99</b>
6.1. Conclusiones . . . . .	99
6.2. Aportaciones . . . . .	100
6.3. Trabajo a Futuro . . . . .	101
<b>Bibliografía</b>	<b>103</b>

# Índice de figuras

2.1.	Procedimiento para el análisis de agrupamientos. Imagen modificada de [1].	23
2.2.	Densidad y conectividad alcanzable en algoritmo DBSCAN. Imagen tomada de [2].	27
2.3.	Estructura general del autocodificador. Figura tomada de [3].	31
2.4.	Arquitectura del modelo transformador. Figura construida conforme a [4].	37
2.5.	Atención de múltiples cabezales, figura modificada de [4].	41
2.6.	Pre-entrenamiento (izquierda) y ajuste fino (derecha) del modelo BERT. Figura construida conforme a [5].	45
2.7.	Ajuste fino de BERT para diferentes tareas de procesamiento de lenguaje natural de forma supervisada. Figura construida conforme [5].	48
3.1.	Distribución del peso de atención de la capa promedio de GridLoc para cada tarea de SentEval, demostrando que las tareas de superficie tienen capas distinguibles, pero las tareas sintácticas y semánticas son indistinguibles. Figura modificada de [6].	55
3.2.	Incrustaciones en 2D de la palabra en inglés “ <i>die</i> ” en diferentes contextos. Figura modificada de 3.2.	56
4.1.	Metodología para resolver el problema de similitud semántica mediante métodos propuestos (mean, max y BiLSTM) y el token CLS.	63
4.2.	Detalle de los métodos de abstracción implementados para abordar la tarea de similitud semántica.	64
4.3.	Metodología para analizar las autoatenciones del modelo BERT.	67
4.4.	Atenciones de cabezales apilados para cada capa del modelo BERT. Cada capa contiene los 12 cabezales con matrices de atención aplanadas.	68
4.5.	Estructura de las representaciones formadas para la entrada al modelo autocodificador.	70
4.6.	Autocodificador propuesto basado en redes LSTM para reducción de dimensionalidad.	71

4.7. Distribución de los datos con respecto a la longitud de secuencia, en el conjunto de prueba STS-Benchmark (a) y en el conjunto de prueba SICK-R (b). . . . .	73
4.8. Distribución de las etiquetas de similitud semántica en el conjunto de prueba STS-Benchmark (a) y el conjunto de prueba SICK-R (b). . . . .	74
5.1. Diagramas de dispersión con agrupamientos del conjunto STS-Benchmark.	81
5.2. Diagramas de dispersión con agrupamientos del conjunto SICK-R. . . . .	82
5.3. Diagramas de caja para análisis de longitud de secuencia de STS-Benchmark.	84
5.4. Diagramas de caja para análisis de longitud de secuencia de SICK-R. . . . .	85
5.5. Diagramas de dispersión de las capas 1, 4, 8 y 12 señalando rangos de tamaño de secuencias para el conjunto STS-Benchmark. . . . .	87
5.6. Diagramas de dispersión de las capas 1,5,9 y 11 señalando rangos de tamaño de secuencias para el conjunto SICK-R. . . . .	88
5.7. Diagramas de caja para analizar la similitud semántica en STS-Benchmark.	89
5.8. Diagramas de caja para analizar la similitud semántica en SICK-R. . . . .	90
5.9. Diagramas de dispersión resaltando las muestras con similitudes semánticas de 0 y 5 para el conjunto STS-Benchmark. . . . .	92
5.10. Diagramas de dispersión resaltando las muestras con similitudes semánticas de 1 y 5 para el conjunto SICK-R . . . . .	93
5.11. Diagramas de caja para analizar la similitud de estructura gramatical de la secuencia más frecuente con respecto a otras estructuras del conjunto STS-Benchmark. . . . .	94
5.12. Diagramas de caja para analizar la similitud de estructura gramatical de la secuencia más frecuente con respecto a otras estructuras del conjunto SICK-R. . . . .	95
5.13. Diagramas de dispersión de las capas 2, 3, 8 y 12, resaltando estructuras con similitud mayor o igual a 0.95 para la estructura más frecuente del conjunto STS-Benchmark. . . . .	97
5.14. Diagramas de dispersión de las capas 2, 3, 10 y 12, resaltando estructuras con similitud mayor o igual a 0.95 para la estructura más frecuente del conjunto SICK-R. . . . .	98

# Capítulo 1

## Introducción

### 1.1. Definición del problema

La efectividad en la resolución de problemas del lenguaje, como la ambigüedad y la diversidad, es crucial para la mayoría de las tareas de procesamiento del lenguaje natural (PLN). Por esta razón, la similitud semántica puede considerarse como un pilar fundamental para abordar otras tareas en el ámbito del procesamiento del lenguaje natural.

En los últimos años, el enfoque hacia las tareas de procesamiento del lenguaje natural, incluida la similitud semántica, ha experimentado un notable éxito gracias a los modelos de aprendizaje profundo, destacando en particular la arquitectura del modelo transformador. Un ejemplo sobresaliente de esta arquitectura es el modelo de representaciones bidireccionales de codificadores de transformadores, conocido como BERT (*Bidirectional Encoder Representations from Transformers*), junto con sus diversas variantes.

Existen numerosos trabajos dedicados a mejorar la precisión en tareas de procesamiento del lenguaje natural mediante transformadores; sin embargo, hay significativamente menos investigaciones centradas en comprender el funcionamiento interno de estos modelos. La mayoría de los estudios dedicados a la interpretabilidad del modelo BERT han fundamentado sus observaciones en el aprendizaje supervisado. Estos trabajos emplean sondas o clasificadores independientes, junto con conjuntos de datos especializados, utilizando la precisión del modelo como indicador de la presencia o ausencia de aspectos lingüísticos específicos, [7], [8], [9], [6], [10],[11], [12], [13]. Aunque el análisis del modelo mediante el aprendizaje supervisado ha proporcionado resultados interesantes, este enfoque podría tener limitaciones al comprender la estructura subyacente de los datos o al descubrir interacciones entre múltiples aspectos lingüísticos.

En la comunidad científica, existe controversia acerca de en qué capas el modelo BERT abstrae ciertos aspectos lingüísticos, particularmente en lo que respecta a la sintaxis y la

semántica. Algunos estudios sostienen que la sintaxis se manifiesta en capas intermedias, mientras que la semántica se desarrolla en capas superiores. Por otro lado, hay trabajos que argumentan que la semántica se forma desde capas intermedias o que la sintaxis emerge en las capas iniciales.

Los estudios que han abordado la interpretabilidad de la semántica lo han hecho principalmente desde tareas específicas como el etiquetado de roles semánticos, el análisis de entidades nombradas, y la verificación de aspectos temporales, del sujeto, del objeto directo, entre otros. Sin embargo, no hay un enfoque en la tarea específica de la similitud semántica. Además, la mayoría de estos estudios se enfocan en analizar las incrustaciones contextualizadas del modelo, demostrando un interés menor en llevar a cabo un análisis de las atenciones y del token CLS.

Es evidente que hay una oportunidad de investigación en el campo de la interpretabilidad de modelos tan complejos y relevantes como BERT. Realizar un análisis de interpretabilidad es crucial, ya que proporciona una mayor confianza y claridad en modelos que a menudo son considerados como cajas negras. Además, este análisis podría revelar áreas o formas de mejora en el modelo.

## 1.2. Motivación

El modelo transformador, ha demostrado ser excepcionalmente eficaz en la resolución de tareas en el procesamiento del lenguaje natural. Sin embargo, para aprovechar el potencial del modelo transformador y elevar su rendimiento, resulta relevante llevar a cabo un análisis detallado de su interpretabilidad.

Esta investigación pretende proporcionar valiosas observaciones que revelen el comportamiento de ciertos aspectos lingüísticos específicos a lo largo de las capas del modelo transformador BERT. La interpretación de un modelo tan complejo como BERT en términos de aspectos lingüísticos no solo proporciona una visión más clara de su funcionamiento interno, sino que también puede señalar posibles áreas de mejora y detectar patrones problemáticos que podrían influir en las decisiones del modelo. Además, esta interpretación puede brindar intuiciones valiosas que inspiren la exploración de nuevas ideas y enfoques, así como revelar oportunidades de investigación en el campo de los modelos transformadores, contribuyendo por consiguiente, al progreso del procesamiento del lenguaje natural en general.

Además, se propone un enfoque de análisis desde el método del aprendizaje no supervisado, un marco de trabajo que ha demostrado ofrecer resultados confiables, al igual que el aprendizaje supervisado. Se espera que este enfoque motive a otros investigadores a adoptar una perspectiva similar. Los hallazgos de este trabajo buscan enriquecer

los resultados actuales, tanto en lo que respecta al token CLS como a las atenciones del modelo.

### 1.3. Hipótesis

Los métodos de agregación y una red del tipo LSTM bidireccional aplicados a la salida del modelo BERT, logran una mejor abstracción de información lingüística que la realizada por el token de clasificación CLS.

Por otro lado, las autoatenciones del modelo BERT abstraen aspectos superficiales del lenguaje, así como aspectos más complejos como la sintaxis y la semántica, a medida que avanza en sus múltiples capas.

Finalmente, el aprendizaje no supervisado proporciona algoritmos y herramientas que permiten realizar un análisis de interpretabilidad lingüística confiable en modelos complejos como BERT.

### 1.4. Justificación

A pesar de que el modelo transformador ha estado presente en la comunidad científica durante un tiempo considerable, su vigencia y relevancia continúan siendo sólidas. Se mantiene como un pilar fundamental para abordar los desafíos en el procesamiento del lenguaje natural. Un ejemplo destacado es el modelo de Representaciones de Codificadores Bidireccionales de Transformadores (BERT), desarrollado en 2018 por [5]. Este modelo ha demostrado un rendimiento de última generación al alcanzar resultados destacados en diversas tareas de comprensión del lenguaje, como se evidencia en el conjunto de datos GLUE (General Language Understanding Evaluation), con un aumento promedio del rendimiento que oscila entre el 4,5 % y el 7,0 %. Además, BERT ha sobresalido en la tarea de respuesta a preguntas según el conjunto de datos Stanford Question Answering Dataset (SQuAD).

Otro ejemplo arraigado en el modelo transformador es la inteligencia artificial Chat Generative Pre-trained Transformer, conocida como ChatGPT. Este fue lanzado en el año 2022 y desarrollado por OpenAI, como se describe en [14]. Este modelo ha dejado una impresión significativa tanto en la comunidad científica como en el público en general debido a sus notables capacidades en diversas áreas, que abarcan desde medicina, negocios, finanzas, educación, entretenimiento, programación, generación de contenido y ventas, como se destaca en [15].

Un análisis de interpretabilidad en el ámbito de modelos transformadores como BERT se vuelve crucial al proporcionar no solo una mayor confianza y claridad, sino también

una comprensión más profunda que va más allá de considerarlos simplemente como cajas negras. Además, ofrece información valiosa que facilita la identificación de áreas de mejora o ajustes en el modelo.

Abordar la tarea de similitud semántica para resolver el problema de la interpretabilidad resulta relevante, ya que la similitud semántica suele ser considerada como un pilar fundamental en las tareas de procesamiento del lenguaje natural.

## 1.5. Objetivos

### 1.5.1. General

Brindar interpretabilidad lingüística de los componentes principales del modelo transformador BERT en el contexto de la similitud semántica.

### 1.5.2. Particulares

- Realizar la búsqueda, selección y procesamiento de conjuntos de datos enfocados a la tarea de similitud semántica en oraciones.
- Implementar métodos de abstracción de las representaciones de la última capa de BERT y llevar a cabo el ajuste fino correspondiente, con el fin de abordar la tarea de similitud semántica.
- Evaluar y comparar la capacidad de abstracción del token de clasificación CLS en relación con los métodos de abstracción propuestos.
- Desarrollar e implementar un modelo autocodificador con el objetivo de gestionar y reducir la dimensionalidad de las matrices de autoatención.
- Desarrollo de una herramienta visual que facilite la búsqueda de patrones lingüísticos, el análisis exploratorio y la comprensión del comportamiento de los datos.
- Realizar análisis exploratorios de las autoatenciones del modelo BERT a nivel de capa.
- Identificar y analizar los aspectos lingüísticos que el modelo BERT abstrae en la autoatención, así como su comportamiento a lo largo de las diferentes capas.

## 1.6. Contribución del problema

Este trabajo busca ofrecer un enfoque de análisis de la interpretabilidad de las atenciones del modelo BERT desde la perspectiva del aprendizaje no supervisado. Esto podría conllevar beneficios como una mejor comprensión de la estructura intrínseca de los datos. Además, el análisis exploratorio no supervisado puede proporcionar intuiciones valiosas para la exploración de nuevas ideas y enfoques, así como destacar áreas de mejora. También tiene el potencial de revelar relaciones complejas entre varios aspectos lingüísticos y patrones que podrían no ser tan evidentes en un contexto supervisado.

La realización de un análisis exploratorio puede resultar abrumador, especialmente cuando carecemos de información externa sobre el comportamiento de los datos o cuando el análisis es exhaustivo, como es el caso de este trabajo. Con el objetivo de facilitar el proceso de análisis y brindar confianza en el análisis exploratorio, se implementa una herramienta de apoyo capaz de sugerir hiperparámetros en algoritmos de agrupamiento y reducción dimensional. Esta herramienta también posibilita la observación de aspectos lingüísticos, como la longitud de secuencia, la similitud semántica y la similitud de la estructura gramatical, con el propósito de ofrecer una mayor intuición y mejorar la comprensión del comportamiento interno de los datos.

Este estudio tiene como objetivo ampliar el análisis del token de clasificación CLS, explorando su capacidad para abstraer información lingüística en comparación con otros métodos propuestos, a través de un enfoque supervisado.

Además, esta investigación contribuye proporcionando conocimiento sobre el comportamiento de las autoatenciones del modelo BERT al abordar la tarea de similitud semántica mediante un enfoque no supervisado. Esto incluye información detallada sobre la longitud de la secuencia, la estructura gramatical y la similitud semántica.

## 1.7. Organización de la tesis

El presente trabajo de tesis se estructura de la siguiente manera:

**Capítulo 2. Marco teórico.** En este capítulo, se revisan las técnicas de exploración del aprendizaje no supervisado, como es la reducción dimensional mediante autocodificador y se revisa algunos algoritmos de agrupamiento. También se revisa las formas de evaluación de los agrupamientos y uso de pruebas estadísticas. Posteriormente se brinda una descripción detallada del modelo transformador, los mecanismos de atención. Se revisa el modelo BERT, abarcando tanto su fase de pre-entrenamiento como su ajuste fino.

### **Capítulo 3. Estado del arte.**

Esta sección expone trabajos que han abordado el problema de la similitud semántica a través del modelo transformador BERT. De forma muy breve se aborda el rendimiento del transformador al implementar la poda de cabezales. Finalmente, de forma amplia se consideran investigaciones que han explorado la interpretabilidad lingüística del modelo transformador BERT, mediante las representaciones contextualizadas del modelo y el uso de sondas o clasificadores independientes.

### **Capítulo 4. Metodología.**

En este capítulo, se presenta una exposición detallada de la selección de los corpus utilizados y se describe la metodología empleada para analizar el modelo BERT, centrándose en sus componentes clave, como el token de clasificación CLS y sus autoatenciones. Se detallan las pruebas llevadas a cabo para evaluar la capacidad de abstracción del token CLS en comparación con los métodos propuestos, tales como los métodos de agregación y la red recurrente LSTM. Además, se describe el proceso de reducción de las matrices de atención a un vector bidimensional, así como los algoritmos de agrupamiento y las métricas de evaluación utilizadas para el análisis exploratorio a nivel de capa.

### **Capítulo 5. Resultados experimentales.**

En este capítulo, se exponen los resultados alcanzados respecto a la capacidad de abstracción de los métodos de agregación, la red recurrente LSTM y el token de clasificación CLS. Asimismo, se detallan los resultados del análisis lingüístico de las atenciones del modelo BERT en la tarea de resolver la similitud semántica. Además, se discuten las observaciones derivadas de los experimentos realizados.

### **Capítulo 6. Conclusiones y trabajo a futuro.**

Finalmente, se presentan las reflexiones y conclusiones surgidas del análisis de los resultados experimentales del Capítulo 5. Se señalan también posibles experimentos futuros que podrían enriquecer y proporcionar una perspectiva más amplia sobre la interpretabilidad lingüística del modelo BERT.

# Capítulo 2

## Marco teórico

En esta sección se revisan algunos conceptos que se utilizarán a lo largo de la metodología de este trabajo. Se inicia con los problemas del lenguaje. Posteriormente, se presenta una exposición detallada de la teoría que sustenta al regresor lineal, ya que este desempeña un papel fundamental en el ajuste fino para la resolución de la similitud semántica.

Como parte de la fundamentación teórica para comprender la interpretabilidad en las autoatenciones del modelo, se examina la técnica de reducción dimensional mediante autoencoderos. Además, se exploran los procedimientos, algoritmos y métricas de evaluación asociados con los agrupamientos. Finalmente, se realiza una revisión teórica de la arquitectura del modelo transformador original y del modelo BERT, proporcionando una base sólida para la comprensión de los procedimientos.

### 2.1. Problemas del lenguaje

En el lenguaje natural, existen algunos problemas, como la ambigüedad y la diversidad, que pueden dificultar la comprensión y la comunicación humana.

#### 2.1.1. Ambigüedad

Se entiende que una forma lingüística, como una oración o parte de ella, es ambigua cuando tiene dos o más significados posibles, y, por lo tanto, dos o más posibles interpretaciones [16].

Existen varias formas de ambigüedad; algunas de ellas son la ambigüedad léxica, la ambigüedad léxica morfológica y la ambigüedad sintáctica [17].

- **Ambigüedad léxica:** Este tipo de ambigüedad se da cuando la palabra puede tomar más de un significado [17]. Puede afectar la interpretación de una frase; por

ejemplo, la palabra "*banco*" puede hacer alusión a un asiento pero también a una institución dedicada a realizar operaciones financieras. Estas definiciones son solo 2 de las 11 posibles definiciones que puede tomar la palabra "*banco*" según la Real Academia Española (RAE). Cabe mencionar que los fenómenos de la polisemia y la homonimia están muy relacionados con la ambigüedad léxica.

- **Ambigüedad léxica morfológica:** No solo distingue el significado de la palabra, sino que también depende del rol sintáctico o categoría gramatical de la palabra dentro de una oración [17, 18]. Por ejemplo, en la oración "*Vino de Querétaro*", la palabra "*vino*" podría referirse a una bebida alcohólica comportándose como un sustantivo, pero también podría referirse al verbo venir.
- **Ambigüedad sintáctica:** Se presenta cuando una oración tiene asociadas más de una representación sintáctica [18]. Es decir, existe más de una interpretación de la oración debido a las relaciones o combinaciones entre las palabras de la oración. Un ejemplo sería la frase "*La mujer vio al niño con un telescopio*", que podría interpretarse de dos maneras: una interpretación es que una mujer ve a un niño a través de un telescopio, y otra interpretación es que una mujer ve a un niño que sostiene un telescopio.

### 2.1.2. Diversidad

Uno de los principales problemas del lenguaje es la ambigüedad, sin embargo, también existe su contraparte, la diversidad en el lenguaje.

Entiéndase por diversidad del lenguaje como aquella en la que un significado puede tener muchas cadenas, formas u oraciones para representar el significado o la idea que se quiere comunicar [17]. Por ejemplo, “Ella estaba agotada y se fue a descansar.” “Ella estaba cansada y se fue a reposar”. Cabe destacar que en la diversidad la sinonimia juega un papel importante.

## 2.2. Regresión lineal

La regresión lineal es un modelo paramétrico, es decir, mantiene un número fijo de parámetros al hacer algunas suposiciones sobre la distribución de los datos [19].

De acuerdo con [19], la regresión lineal asume que la respuesta a una entrada es una función lineal, que se puede definir como:

$$y(\mathbf{x}) = \mathbf{w}^T \mathbf{x} + \epsilon = \sum_{j=1}^D w_j x_j + \epsilon \quad (2.1)$$

Donde  $\mathbf{x}$  es el vector de entrada,  $\mathbf{w}$  es el vector de peso y  $\epsilon$  es el error residual que define la distancia entre las etiquetas reales y las estimaciones hechas por la predicción lineal, es decir,  $\epsilon = (y_i - \hat{y}_i)$ .

Se suele considerar que el error residual tiene una distribución normal  $\mathcal{N}(\mu, \sigma^2)$  donde  $\mu$  es la media y  $\sigma$  la varianza. En la formulación estándar de la regresión lineal, asumimos que la variable de respuesta  $y$  sigue una distribución normal. Una conexión explícita entre la regresión y la distribución gausiana es la siguiente, [19]:

$$p(y|x, \theta) = \mathcal{N}(y|\mu(x), \sigma^2(x)) \quad (2.2)$$

En el caso más simple de regresión lineal, se asume que  $\mu$  es una combinación lineal  $\mu = \mathbf{w}^T \mathbf{x}$  y que la varianza es fija  $\sigma^2(\mathbf{x}) = \sigma^2$ , de tal forma que los parámetros del modelo serían  $\theta = (\mathbf{w}, \sigma^2)$ .

La regresión lineal también puede modelar relaciones no lineales a través de sustituir  $x$  con una función no lineal de las entradas  $\varphi(x)$ , es decir:

$$p(y|x, \theta) = \mathcal{N}(y|\mathbf{w}^T \varphi(\mathbf{x}), \sigma^2(x)) \quad (2.3)$$

También se puede realizar una regresión polinómica si el grado de la variable de entrada  $x$  es  $d \geq 1$ , pero si  $d = 1$  se conoce como regresión lineal. La forma de aplicar una regresión lineal a más de una entrada tendría la siguiente forma:  $E[y|x] = w_0 + w_1 x_1 + w_2 x_2$ .

Una forma de estimar los parámetros del modelo estadístico de la regresión lineal, es a través de la máxima verosimilitud (*likelihood*), definida como:

$$\hat{\theta} = \arg \max p(D|\theta) \quad (2.4)$$

De acuerdo con [19], si se considera que tenemos muestras independientes e idénticamente distribuidas *iid*, entonces la verosimilitud se calcula en función del producto de las probabilidades individuales de las muestras. Debido a que el logaritmo es una función monótona creciente, este se puede utilizar para convertir los productos de probabilidad en sumas y simplificar su cálculo. La estimación de la verosimilitud, se puede definir entonces como:

$$l(\theta) = \log p(\mathcal{D}|\theta) = \sum_{i=1}^N \log p(y_i|\mathbf{x}_i, \theta) \quad (2.5)$$

El método de verosimilitud utilizado en la regresión lineal e incorporando la distribución normal, tiene la siguiente forma de acuerdo con [19]:

$$l(\theta) = \sum_{i=1}^N \log \left[ \left( \frac{1}{2\pi\sigma^2} \right)^{\frac{1}{2}} \exp \left( -\frac{1}{2\sigma^2} (y_i - \mathbf{w}^T \mathbf{x}_i)^2 \right) \right] = \frac{-1}{2\sigma^2} RSS(\mathbf{w}) - \frac{N}{2} \log(2\pi\sigma^2) \quad (2.6)$$

Donde RSS es la suma residual de cuadrados:

$$RSS(\mathbf{w}) = \sum_{i=1}^N (y_i - \mathbf{w}^T x_i)^2 \quad (2.7)$$

RSS también puede verse como la suma de errores cuadrados SSE y la expresión  $SSE/N$  es conocida como el error cuadrático medio o MSE. De acuerdo con esta observación, es que se considera justificable el uso de MSE como estimador de máxima verosimilitud, ya que maximizar la probabilidad logarítmica con respecto a  $w$  da la misma estimación de los parámetros  $w$  al calcular el error cuadrático medio MSE. A pesar de que MSE y máxima verosimilitud tienen diferentes valores dan la misma ubicación del óptimo, [3].

## 2.3. Agrupamiento

El agrupamiento es un componente útil del análisis de exploración de datos, que pertenece al enfoque del aprendizaje no supervisado. Nos permite comprender la estructura interna de un conjunto de datos e identificar patrones y relaciones entre los datos.

El agrupamiento puede conceptualizarse como un problema de optimización [20], donde el objetivo es formar agrupaciones en las que los objetos dentro de cada grupo o *cluster* sean altamente similares entre sí, mientras que los objetos en diferentes grupos sean significativamente distintos [21].

No hay una interpretación clara de lo que es un grupo, pero se han tratado de dar algunas interpretaciones, como la de [22] que define a los elementos del grupo como objetos que comparten distancias pequeñas entre sí, o la de [23] que define que las muestras de un grupo tienen regiones altamente densas rodeadas de regiones vacías; incluso [24] sugiere que un grupo válido puede ser aquel que produce una respuesta valiosa para el investigador.

De acuerdo con [1], el proceso de análisis de agrupamiento consiste en 4 pasos con una vía de retroalimentación. El primer paso es la selección o extracción de características, donde se eligen las características más relevantes o se realizan transformaciones en los

datos para obtener características útiles. A continuación, viene la selección del algoritmo de agrupamiento basado en alguna medida de proximidad y la definición del número de conglomerados  $K$  dependientes de una función objetivo. Posteriormente, se realiza la evaluación de los agrupamientos mediante el uso de métricas intrínsecas (que miden la calidad del agrupamiento), extrínsecas (en base a información externa como etiquetas de verdad) o relativas (comparación de diferentes estructuras de agrupamiento). Finalmente, el último paso es la interpretación de los resultados, es decir, obtener información significativa de los datos. Estos pasos pueden verse en la figura 2.1.

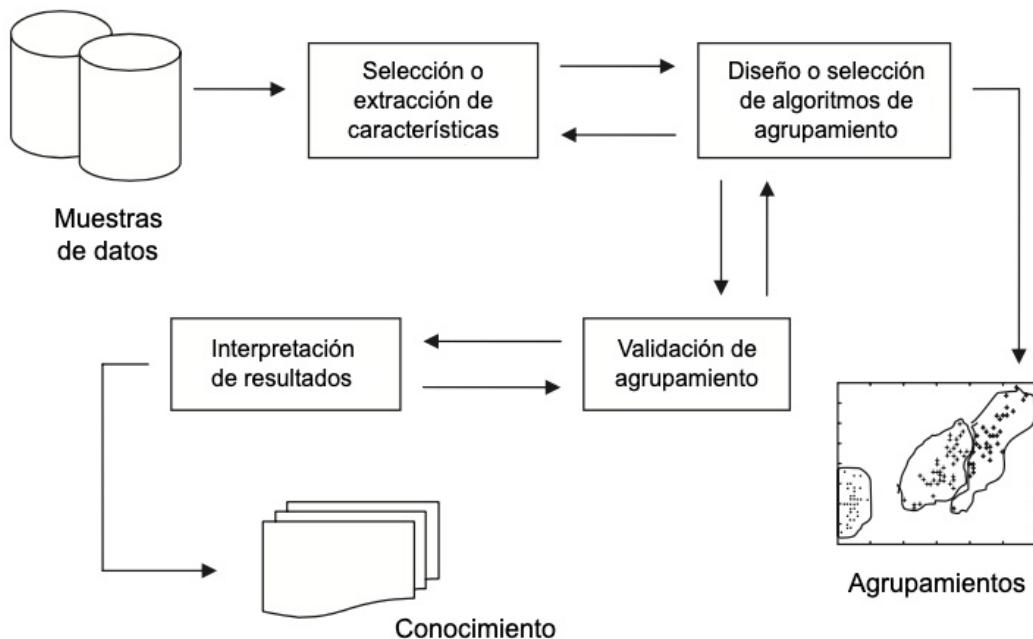


Figura 2.1: Procedimiento para el análisis de agrupamientos. Imagen modificada de [1].

En las siguientes secciones, se proporcionará una descripción de los algoritmos de agrupamiento más comunes y útiles. Además, en los siguientes apartados se tratarán las técnicas de reducción de dimensionalidad y métricas de evaluación.

### 2.3.1. K-means

Es un algoritmo de agrupamiento particional iterativo y su idea base, definida por [25], es encontrar agrupamientos en términos de su distancia cuadrada. K-means define un clúster o grupo como un punto representativo. El punto es definido como la media (centroide) de los objetos que son asignados al clúster, por eso su nombre. Si usamos  $\mu_k$

para definir el punto medio para un clúster  $k$  y  $z_{nk}$  como una variable indicadora binaria que es 1 si el objeto  $n$  es asignado al clúster  $k$  y cero en otro caso. Cada objeto debe ser asignado a un solo clúster. Esto es:  $\sum_k z_{nk} = 1$ . Esto lleva a la siguiente expresión para  $\mu_k$ :

$$\frac{\sum_n z_{nk} x_n}{\sum_n z_{nk}} \quad (2.8)$$

Los *clusters* o grupos son definidos como los centros de los puntos que se le asignaron al *cluster* y los puntos son asignados al *cluster* al cual están más cerca, es decir el *cluster*  $k$  que da la distancia mínima  $(x_n - \mu_k)^T(x_n - \mu_k)$ . Si conocemos los clusters  $\mu_1, \dots, \mu_k$  se puede computar las asignaciones. Para el cálculo de la distancia podríamos considerar la distancia euclidiana, Mahalanobis, etc. El algoritmo de k-means, considerando que comenzamos con valores iniciales aleatorios para las medias de los clusters  $\mu_1, \dots, \mu_k$ , es el siguiente [25]:

1. Para cada punto  $x_n$ , encontrar  $k$  que minimice  $(x_n - \mu_k)^T(x_n - \mu_k)$ , es decir, la media del grupo más cercano y establecer  $z_{nk} = 1$  y  $z_{nj} = 0$  para todo  $j \neq k$ .
2. Si todas las asignaciones ( $z_{nk}$ ) no cambian desde la iteración previa, detener.
3. Actualizar cada  $\mu_k$  con la ecuación 2.8.
4. Regresar a 1.

El algoritmo k-means es no determinista [20], ya que es sensible a la inicialización de los centroides iniciales. Para abordar este problema, es común realizar múltiples inicializaciones y seleccionar la opción que produce los mejores resultados. Además, variantes del algoritmo, como k-means++, proponen estrategias de inicialización más sofisticadas. Aunque k-means es efectivo para detectar conglomerados con forma hiperesférica [1], su principal desventaja es la necesidad de especificar el número de conglomerados. Sin embargo, técnicas como el método del codo pueden ayudar en la determinación de este parámetro.

### 2.3.2. Jerárquico

En este algoritmo, los datos no se dividen en un número predeterminado de grupos como en el algoritmo de k-means, sino que realiza una serie de particiones que pueden variar desde un único grupo que contiene  $n$  muestras hasta  $n$  grupos que contienen una muestra; esta forma de agrupamiento jerárquico se conoce como del tipo divisivo. Existe

otra forma de agrupamiento que es del tipo aglomerativo, que consiste en realizar una serie de fusiones sucesivas de  $n$  individuos en grupos. Con el método jerárquico, las divisiones y fusiones son irrevocables [24].

Dado que este algoritmo en última instancia llegará a un solo conglomerado (aglomerativo) o bien dividirá el conjunto de muestras en  $n$  grupos, es necesario que el investigador defina un número de agrupamientos óptimo con la finalidad de decidir cuándo detenerse.

Existen varias formas de representar las fusiones o las divisiones realizadas por un algoritmo jerárquico a lo largo del análisis. Por ejemplo, un diagrama *n-tree* o un dendrograma [21].

### Método aglomerativo

El método aglomerativo es el más utilizado del algoritmo jerárquico, en donde, como ya se ha mencionado, parten de  $n$  grupos de un solo miembro y finalizan con uno solo. En cada paso se fusionan individuos o grupos de individuos más cercanos o similares; sin embargo, existen diferentes formas de definir una distancia entre un individuo y un grupo o entre dos grupos [24]. Estos métodos de definición de distancia se conocen como métodos de enlace o *linkage methods*. A continuación, se describen brevemente los más populares [21]:

- Enlace único: Emplea la distancia del vecino más cercano para medir la disimilitud entre dos grupos.
- Enlace completo: El método de enlace completo utiliza la distancia del vecino más lejano para medir la distancia entre dos grupos.
- Enlace promedio: Se define como la distancia promedio entre todos los pares de puntos, donde cada punto pertenece a un conglomerado diferente.
- Enlace de centroide: Se define como la distancia euclíadiana entre los centroides (medias) de los dos conglomerados.
- Enlace de Ward: También denominado método de varianza mínima ya que busca formar particiones de manera que minimice la pérdida de información por cada fusión (menor aumento en la varianza total) en base al criterio de suma de cuadrados de error.

### Método divisivo

Como se mencionó anteriormente, el enfoque divisivo sigue el camino contrario al método aglomerativo. Comienza con un conglomerado que abarca todas las muestras, y

el número de conglomerados aumenta a medida que se dividen conglomerados existentes según ciertos criterios. Existen dos tipos de métodos de agrupamiento divisivo: los monotéticos, que realizan divisiones basadas en la posesión o no de un atributo específico, y los políticos, que dividen los datos en función de los valores de todos los atributos [21].

A pesar de sus ventajas, los métodos jerárquicos, y en particular los divisivos, presentan desafíos computacionales significativos. La búsqueda de una bipartición óptima en el método divisivo se clasifica como un problema *np-hard* [21]. Además, surge un problema adicional relacionado con la monotonicidad de estos algoritmos, es decir, la determinación de cuál conglomerado dividir en cada etapa del proceso.

### 2.3.3. DBSCAN

El algoritmo de agrupación espacial basado en densidad, conocido como DBSCAN (Density-Based Spatial Clustering of Applications with Noise), es un algoritmo que utiliza la densidad de los puntos de datos para formar agrupaciones, con la ventaja de generar agrupaciones con formas arbitrarias [1].

DBSCAN, según [2], define las muestras como conglomerados si están en regiones densas y como ruido o puntos atípicos a las muestras en regiones con baja densidad. Los grupos están definidos por un conjunto de al menos  $MinPts$  puntos u objetos en una región densa con radio  $Eps$ . Estos dos parámetros son definidos por el usuario. Para comprender el funcionamiento de este algoritmo es necesario introducir algunos conceptos y nomenclatura. Sea  $D$  un conjunto (base de datos) de puntos de datos. Los agrupamientos basados en densidad requieren de una función de distancia  $dist(p, q)$  para pares de puntos. Se denota  $NEps(p)$  como el vecindario  $Eps$  de un punto  $p$  y se determina como  $NEps(p) = q \in D | dist(p, q) \leq Eps$ . Un punto  $p$  es directamente alcanzable por densidad de un punto  $q$  con respecto a  $Eps$  y  $MinPts$  si hay una cadena de puntos si  $p \in NEps(q)$  y  $|NEps(q)| \geq MinPts$ . Un punto  $p$  es densamente conectado a un punto  $q$  si hay un punto  $o$  tal que  $p$  y  $q$  sean alcanzables por densidad desde  $o$ , con respecto a  $Eps$  y  $MinPts$ . En la figura 2.2, se ilustra cómo  $p$  es altamente alcanzable desde  $q$ , pero no al revés. También se observa cómo los puntos  $a$  y  $c$  están conectados por densidad a través de  $b$ .

Un agrupamiento debe cumplir las siguientes condiciones, según [2]:

- Si  $\forall p, q$ , si  $p \in C$  y  $q$  es alcanzable por densidad desde  $p$  con respecto a  $Eps$  y  $MinPts$ , entonces  $q \in C$ .
- $\forall p, q \in C$  entonces  $p$  es densamente conectado a  $q$  con respecto a  $Eps$  y  $MinPts$ .

Se define como ruido o puntos atípicos a todos los puntos de datos de un conjunto de datos que no pertenecen a un agrupamiento, es decir, ruido =  $p \in D | p \notin C_i \forall i$ . Se definen

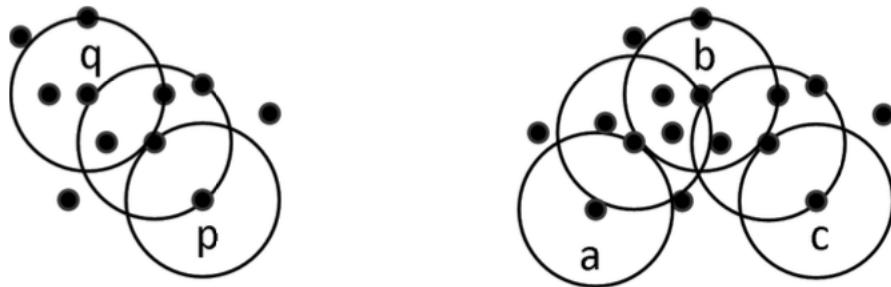


Figura 2.2: Densidad y conectividad alcanzable en algoritmo DBSCAN. Imagen tomada de [2].

así 3 diferentes tipos de puntos:

- Puntos centrales o *core points*: puntos que tienen un vecindario denso  $|NEps(p)| \geq MinPts$
- Puntos de borde o *border points*: puntos que pertenecen a un agrupamiento pero cuyo vecindario no es denso
- Puntos atípicos o *noise points*: puntos que no pertenecen a algún agrupamiento.

El algoritmo de DBSCAN comienza asignando de forma aleatoria algún punto  $p$ , recuperando todos los puntos alcanzados por densidad desde  $p$  con respecto a  $Eps$  y  $MinPts$ . Realiza consultas desde  $p$  y los puntos cercanos a  $p$ . Si  $p$  es un punto central, se forma un grupo; de lo contrario, si  $p$  no es un punto central, se clasifica como ruido o atípico, y el proceso se repite para el siguiente punto. Si  $p$  es un punto de borde, se alcanzará al recopilar los puntos alcanzables por densidad desde algún punto central, y finalmente, se incluirá en el grupo. El algoritmo terminará cuando todos los puntos hayan sido asignados a algún agrupamiento o como ruido, según [2].

La complejidad temporal de DBSCAN es  $O(n \log n)$ ; sin embargo, en altas dimensiones puede comportarse como  $O(n^2)$ . Usando una estructura de datos de cuadrícula, se puede reducir su complejidad a  $O(n)$ , según [2].

### 2.3.4. Espectral

La agrupación espectral se basa en la estructura de una matriz de similitud, que agrupa los puntos en conglomerados disjuntos, con puntos en el mismo grupo que tienen alta similitud y puntos en diferentes grupos que tienen baja similitud [26].

El algoritmo espectral puede brindar buenas soluciones a geometrías complicadas como espirales entrelazadas. La idea central de este algoritmo subyace en la idea de utilizar los eigenvectores de una matriz de adyacencia para determinar agrupaciones. Existen dos principales variantes en este tipo de algoritmo, que son el uso del laplaciano no normalizado y el laplaciano normalizado, según [2].

La resolución del algoritmo espectral tiene los siguientes pasos de acuerdo con [2]:

- Construcción de un grafo de similitud para todos los puntos.
- Mediante el uso de los eigenvectores del Laplaciano se incorporan los puntos de datos a un espacio en el que los agrupamientos son más evidentes.
- Se aplica un algoritmo clásico de agrupamiento para particionar los datos.

El nombre “Espectral” del algoritmo proviene del hecho de que los resultados de agrupamiento se obtienen mediante el análisis del espectro del Laplaciano del grafo.

Como ya se mencionó, el primer paso es construir el grafo de similitud  $G = (V, E)$  no dirigido, donde  $V$  se refiere al conjunto de nodos y  $E$  a las aristas del grafo. A partir de este grafo, se construye su matriz de adyacencia  $W$ . Para construir la matriz  $W$ , se puede utilizar grafos de  $K$  vecinos más cercanos, grafo de vecindario  $\epsilon$ , o por grafo completamente conectado. Una vez que se cuenta con el grafo y la matriz de adyacencia, el siguiente paso es calcular el Laplaciano  $L$  mediante:

$$L = D - W \quad (2.9)$$

Donde  $D$  es la matriz diagonal de grado.

Si se usa el enfoque de no normalización del Laplaciano, entonces se utiliza directamente  $L = D - W$  para posteriormente obtener los  $k$  eigenvectores superiores (los  $k$  eigenvalores más pequeños) de  $L$ . Con base en los  $k$  eigenvectores obtenidos, se forma la matriz de  $k$  eigenvectores  $F$ . Cada fila de la matriz  $F$  se trata como un vértice, los cuales se agrupan en  $k$  conglomerados mediante algún algoritmo clásico de agrupamiento como k-means, [2].

Por otro lado, existen varias formas de implementar el enfoque del Laplaciano normalizado, por ejemplo, la versión simétrica. Esta versión consiste en computar  $L_{\text{sim}} = D^{-1/2} L D^{-1/2}$ . Posteriormente, se calculan los  $k$  eigenvectores superiores para formar la matriz  $F$ . Las filas de la matriz  $F$  se normalizan y finalmente, las filas de la matriz  $F$  normalizada se tratan como vértices que se dividen en  $k$  conglomerados mediante un algoritmo de agrupamiento tradicional, [2].

### 2.3.5. Mezcla Gausiana

De acuerdo con [21], el algoritmo de mezcla gaussiana es un algoritmo de agrupamiento basado en modelos probabilísticos que trata de optimizar el ajuste entre los datos y el modelo. Se considera que los datos provienen de una mezcla de distribuciones de probabilidad, en este caso, la distribución gaussiana, las cuales representan a conglomerados diferentes.

En los modelos de mezcla gaussiana, se asume que los datos  $D = \{x_1, x_2, \dots, x_n\}$  en un espacio dimensional  $d$  provienen de un vector aleatorio con una densidad [21]:

$$f(\mathbf{x}) = \sum_{j=1}^k p_j \varphi(\mathbf{x}|\mu_j, \Sigma_j) \quad (2.10)$$

donde  $p_j$  son las proporciones de mezcla con la condición de que  $\sum_{j=1}^k p_j = 1$ , y  $\varphi(x|\mu, \Sigma)$  denota la densidad de la distribución gaussiana con vector de media  $\mu$  y matriz de covarianza  $\Sigma$ , es decir:

$$\varphi(\mathbf{x}|\mu, \Sigma) = \frac{\exp[-\frac{1}{2}(\mathbf{x} - \mu)^T \Sigma^{-1} (\mathbf{x} - \mu)]}{\sqrt{(2\pi)^d |\Sigma|}} \quad (2.11)$$

Para llevar a un criterio de agrupamiento, la matriz de covarianza es descompuesta:

$$\Sigma_k = \lambda_k D_k A_k D_k^T \quad (2.12)$$

donde  $\lambda_k = |\Sigma_k|^{1/d}$ ,  $D_k$  es la matriz de eigenvectores de la covarianza,  $A_k$  es una matriz diagonal con los eigenvalores normalizados y ordenados de forma decreciente.  $\lambda_k$  determina el volumen,  $D_k$  determina la orientación y  $A_k$  la forma del  $k$ -ésimo grupo, [21].

Para estimar los parámetros se puede proponer el enfoque de máxima verosimilitud de mezcla. En el enfoque de mezcla se tiene como objetivo maximizar la verosimilitud (*likelihood*) de los parámetros  $\theta$  ( $\mu$  y  $\Sigma$ ) mediante *log-likelihood*:

$$L(\theta|\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n) = \sum_{i=1}^n \ln \left[ \sum_{j=1}^k p_j \varphi(\mathbf{x}_i|\mu_j, \Sigma_j) \right] \quad (2.13)$$

Finalmente, el algoritmo de *Expectation-Maximization* (EM) puede ser utilizado para calcular los parámetros  $\theta$  mediante *log-likelihood* de 2.13. Variaciones de los parámetros  $\lambda_k$ ,  $D_k$  y  $A_k$  conducen a diferentes modelos de agrupamiento, como los de covarianza esférica o covarianza diagonal, [21].

## 2.4. Reducción dimensional

Es frecuente encontrarnos con el problema de la maldición de dimensionalidad en problemas de aprendizaje automático. Manejar una gran cantidad de datos puede incluso entorpecer la obtención de una buena solución o ser lento de resolver. Cuando nuestros datos tienen altas dimensiones, estos suelen ser más dispersos debido a que tenemos mayor espacio. Esta situación puede llevar incluso al sobreajuste. Una manera de solucionarlo es incrementando el número de instancias para alcanzar una determinada densidad; sin embargo, esto no es viable en la práctica debido a que la cantidad de instancias necesarias crece exponencialmente con respecto a la dimensión, [27].

Sin embargo, tenemos la alternativa de utilizar algún método de reducción dimensional que nos permita eliminar la redundancia de los datos y obtener representaciones más compactas que conserven la información más relevante. Reducir el número de dimensiones también nos facilita la visualización de los datos a un espacio 2D y 3D. La ventaja de tener una vista condensada de los datos con alta dimensionalidad a un espacio 2D o 3D es que nos permite detectar patrones en los datos, como son las agrupaciones, y realizar un análisis de nuestros datos [27].

En este apartado se describirá la reducción dimensional mediante el modelo autocodificador.

### 2.4.1. Autocodificador

Los autocodificadores, conocidos en inglés como *autoencoders*, son redes neuronales que aprenden representaciones latentes o codificaciones sin supervisión. Estas codificaciones suelen tener un tamaño mucho menor que la entrada, lo que permite que estos modelos puedan usarse para reducción de dimensionalidad, [27]. La forma en que se entrena esta red es tratando de reconstruir o copiar su entrada en su salida.

El autocodificador consiste de 2 bloques, una función de codificación  $h = f(x)$  y un decodificador que produce una reconstrucción  $r = g(h)$ , donde  $h$  abstrae los aspectos importantes de su entrada y suele conocerse como representación latente o capa oculta. La estructura del autocodificador puede verse en la figura 2.3, donde se muestra cómo una entrada  $x$  es mapeada a una salida de reconstrucción  $r$  a través de la representación interna  $h$ . Se observa el bloque de codificación  $f$ (mapeo de  $x$  a  $h$ ) y el bloque de decodificación  $g$ (mapeo de  $h$  a  $r$ ). El modelo está diseñado para no copiar perfectamente su entrada, sino para priorizar y abstraer los aspectos más importantes de la entrada, [3].

Los autocodificadores han generalizado la idea del codificador  $p_{\text{codificador}}(h|x)$  y decodificador  $p_{\text{decodificador}}(x|h)$ . Los autocodificadores han sido utilizados comúnmente para

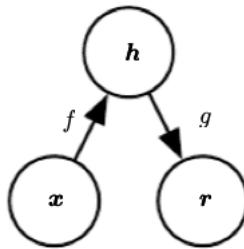


Figura 2.3: Estructura general del autocodificador. Figura tomada de [3].

reducción de dimensionalidad y aprendizaje de características; sin embargo, recientemente se han utilizado para modelos generativos.

Generalmente, no se está interesado en la salida del autocodificador, sino en la representación latente. Cuando la representación latente tiene una dimensión menor que su entrada, se conoce como subcompleto y tiende a capturar las características más importantes de los datos. La función de costo del autocodificador se define como  $L = \|x - g(f(x))\|$ , que penaliza el error cuadrático medio cuando  $g(f(x))$  es diferente de  $x$ .

## 2.5. Evaluación de agrupamientos

En el aprendizaje no supervisado, no hay forma de demostrar que los grupos encontrados por los algoritmos de agrupamiento son válidos; no hay una solución consistente y concluyente para la validación de clústeres. Por lo tanto, es necesario desarrollar algunos criterios con validez. Otro desafío del agrupamiento no supervisado es definir el número de agrupamientos. Debido a esto es necesario requerir algunos métodos o medidas para intentar validar un agrupamiento, [21].

Las medidas de validación de agrupamiento se pueden categorizar en dos tipos principales: validación externa de agrupamiento (extrínsecas) y validación interna de agrupamiento (intrínsecas), de acuerdo con [2]. La diferencia principal radica en si se utiliza o no información externa para la validación del agrupamiento. Sin embargo, hay literatura que incluso considera un tipo más, el criterio relativo, [21], [1].

A continuación, se describen algunas de las métricas consideradas como intrínsecas y extrínsecas.

### 2.5.1. Métricas extrínsecas

La validación externa requiere información externa que no está presente en los datos para evaluar en qué medida la estructura de agrupamiento definida por algún algoritmo

de agrupamiento coincide con alguna estructura externa. Un ejemplo de esto podría ser la información proporcionada por etiquetas de clase, [2]. Algunas métricas extrínsecas populares se describen en la tabla 2.1:

Cuadro 2.1: Métricas extrínsecas, tomadas de [2].

Métrica	Definición	Rango
Entropía	$-\sum_i p_i (\sum_j \frac{p_{ij}}{p_i} \log \frac{p_{ij}}{p_i})$	$[0, \log K']$
Información mutua	$-\sum_i \sum_j p_{ij} \log \frac{p_{ij}}{p_i p_j}$	$(0, \log K')$
Estadística de rand	$[(\binom{n}{2}) - \sum_i (\binom{n_i}{2}) - \sum_j (\binom{n_j}{2}) + 2 \sum_{ij} (\binom{n_{ij}}{2})] / (\binom{n}{2})$	$(0, 1]$
Coeficiente de Jaccard	$\sum_{ij} (\binom{n_{ij}}{2}) / [\sum_i (\binom{n_i}{2}) + \sum_j (\binom{n_j}{2}) - \sum_{ij} (\binom{n_{ij}}{2})]$	$[0, 1]$
Índice Fowlkes & Mallows	$\sum_{ij} (\binom{n_{ij}}{2}) / \sqrt{\sum_i (\binom{n_i}{2}) \sum_j (\binom{n_j}{2})}$	$[0, 1]$

Algo común de las métricas de la tabla 2.1 es que pueden calcularse a partir de una tabla de contingencia. De acuerdo con [2], dado un conjunto  $D$  con  $n$  objetos, supongamos que hay una partición  $P = \{P_1, \dots, P_k\}$  de  $D$ , donde  $\bigcup_{i=1}^k P_i = D$  y  $P_i \cap P_j = \emptyset$  para  $1 \leq i \neq j \leq K$ , y  $K$  es el número de agrupamientos. Como la etiqueta de verdad es dada, entonces se puede generar otra partición en  $D$ ,  $C = \{C_1, \dots, C_{k'}\}$ , donde  $\bigcup_{i=1}^{k'} C_i = D$  y  $C_i \cap C_j = \emptyset$  para  $1 \leq i \neq j \leq k'$ , donde  $k'$  es el número de clases.  $n_{ij}$  denota el número de objetos en el agrupamiento  $P_i$  de la clase  $C_j$ . De la tabla 2.1, note que  $p_{ij} = n_{ij}/n$ ,  $p_i = n_i/n$  y  $p_j = n_j/n$ .

La entropía mide la pureza de los agrupamientos con respecto a las etiquetas de clase. La Información Mutua mide la cantidad de información que se pierde o se gana al cambiar del conjunto de clases al conjunto de agrupamientos. En cuanto al estadístico de Rand, el coeficiente de Jaccard y el índice de Fowlkes y Mallows evalúan la calidad del agrupamiento por los acuerdos o desacuerdos de los pares de objetos de datos en diferentes particiones.

### 2.5.2. Métricas intrínsecas

Las medidas intrínsecas evalúan la calidad de una estructura de agrupamiento sin tener en cuenta la información externa. A menudo, estas medidas se basan en dos criterios, [2]:

- Compacidad: Que mide cuán relacionados están los objetos de un agrupamiento en función de su varianza. Una mayor varianza indica menor compacidad.
- Separación: Mide cuán diferentes o separados están los agrupamientos entre sí, por ejemplo, la distancia de los centros entre agrupamientos y las distancias mínimas entre pares de objetos en diferentes agrupamientos.

Cuadro 2.2: Métricas intrínsecas, tomadas de [2].

Métrica	Definición
Índice Calinski-Harabasz	$\frac{\sum_i n_i \cdot d^2(c_i, c) / (NC - 1)}{\sum_i \sum_{x \in C_i} d^2(x, c_i) / (n - NC)}$
Índice de Dunn	$\min_i \left\{ \min_j \left( \frac{\min_{x \in C_i, y \in C_j} d(x, y)}{\max_k (\max_{x, y \in C_k} d(x, y))} \right) \right\}$
Índice de silueta	$\begin{aligned} & \frac{1}{NC} \sum_i \left\{ \frac{1}{n_i} \sum_{x \in C_i} \frac{b(x) - a(x)}{\max[b(x), a(x)]} \right\} \\ & a(x) = \frac{1}{n_i - 1} \sum_{y \in C_i, y \neq x} d(x, y) \\ & b(x) = \min_{j, j \neq i} \left[ \frac{1}{n_j} \sum_{y \in C_j} d(x, y) \right] \end{aligned}$
Índice de Davies-Bouldin	$\frac{1}{NC} \sum_i \max_{j, j \neq i} \left\{ [\frac{1}{n_i} \sum_{x \in C_i} d(x, c_i) + \frac{1}{n_j} \sum_{x \in C_j} d(x, c_j)] / d(c_i, c_j) \right\}$

En la tabla 2.2 se pueden observar algunas métricas intrínsecas. De acuerdo con [2],  $D$  es el conjunto de datos,  $n$  es el número de objetos en  $D$ ,  $c$  es el centro de  $D$ ,  $P$  es el número de atributos de  $D$ ,  $NC$  es el número de agrupamientos,  $C_i$  es el  $i$ -ésimo agrupamiento,  $n_i$  es el número de objetos en  $C_i$ ,  $c_i$  es el centro de  $C_i$ ,  $k$  es el número de vecinos más cercanos y  $d(x, y)$  es la distancia entre  $x$  y  $y$ .

De acuerdo con [2]:

- **Calinski-Harabasz:** Evalúa la validez del clúster midiendo la relación entre la dispersión promedio entre los agrupamientos y la dispersión promedio dentro de los agrupamientos, donde un valor más grande indica una mejor separación entre los agrupamientos.
- **Índice de Dunn:** Mide la relación entre la separación mínima entre agrupamientos y la máxima compactación intra agrupamientos, donde un valor más alto sugiere una mejor partición, con clústeres más compactos y bien separados.
- **Índice de silueta:** Mide cuán bien definidos están los agrupamientos al evaluar la coherencia interna y la separación entre los agrupamientos. Un valor más alto (más cercano a 1) indica agrupamientos bien definidos, mientras que un valor bajo (cerca de -1) sugiere que los objetos podrían estar asignados incorrectamente.
- **Índice de Davies-Bouldin:** Evalúa la compacidad y la separación de los agrupamientos. Un valor más bajo indica una mejor partición, donde los agrupamientos son compactos y bien separados.

## 2.6. Pruebas estadísticas para análisis de agrupamientos

A menudo, los científicos o ingenieros requieren de un procedimiento matemático (prueba estadística) de toma de decisiones basado en datos para proporcionar conclusiones o fundamentar decisiones. En este proceso, el ingeniero o científico postula algo acerca de un sistema. Esta conjetura sobre una o más poblaciones se puede expresar como una hipótesis estadística [28].

Las pruebas estadísticas pueden ser útiles en el análisis de agrupamientos, ya que permiten comparar grupos o condiciones y determinar si existen diferencias significativas entre ellos mediante el uso de la media, varianza, etc.

De acuerdo con [28], para establecer la estructura de la prueba de hipótesis es necesario establecer una hipótesis nula  $H_0$  y una hipótesis alternativa  $H_1$ . La hipótesis nula se refiere a la hipótesis que se desea probar y su rechazo conduce a la aceptación de una hipótesis alternativa. La hipótesis alternativa representa lo que se desea probar. Los resultados de una prueba de hipótesis son:

- Rechazar  $H_0$  a favor de  $H_1$  debido a evidencia suficiente en los datos.
- No rechazar  $H_0$  debido a falta de evidencia en los datos.

El procedimiento de toma de decisiones puede conducir a dos tipos de error. El error de tipo 1 se da cuando se rechaza  $H_0$  cuando esta es verdadera. El error de tipo 2 se da cuando no rechazamos  $H_0$  cuando es falsa. De acuerdo con el enfoque clásico de pruebas de hipótesis, se suele establecer un valor de significancia  $\alpha$  fijo (usualmente 0.05 o 0.01) que establece un umbral o probabilidad que controla el riesgo de cometer el error de tipo 1, [28].

Cuando se realiza una prueba estadística, se obtiene un valor  $p$  (probabilidad). Usualmente, ese valor  $p$  se compara con el valor de significancia elegido  $\alpha$ . Si el valor  $p$  es menor que el valor de significancia, entonces se rechaza  $H_0$ . Si el valor  $p$  es mayor, no se rechaza  $H_0$ . Sin embargo, existe otro tipo de enfoque que no depende de un valor de significancia fijo  $\alpha$ , sino que consiste en la evaluación y análisis de los valores  $p$ , lo cual brinda una interpretación más flexible y útil cuando no hay un umbral de significancia apropiado, pero con una interpretación más subjetiva y dependiente del investigador, [28].

En el contexto del análisis de agrupamientos, la hipótesis nula  $H_0$  podría afirmar que no hay diferencias significativas entre los grupos identificados por los algoritmos de agrupamiento, es decir, que los grupos se deben al azar. Mientras que la hipótesis alternativa

podría sostener que hay diferencias significativas entre los grupos identificados por el algoritmo de agrupamiento, es decir, que los grupos representan patrones en los datos.

Para considerar o seleccionar el tipo de prueba estadística, se consideran algunos factores como el tipo de dato, cantidad de grupos y estructura de diseño experimental. Por ejemplo, cuando nuestros datos son numéricos, asumen una distribución normal (datos paramétricos), una homogeneidad de varianza, se cuentan con más de 2 grupos se puede realizar un análisis de varianza ANOVA. Sin embargo, si los datos de análisis son no paramétricos, podríamos considerar otro tipo de prueba como Mann-Whitney U o Kruskal-Wallis, [29].

### 2.6.1. Prueba Mann-Whitney U

De acuerdo con [30], Mann-Whitney U es una prueba que considera solo dos muestras o grupos de datos que son independientes y no paramétricos. Las muestras se combinan y se ordenan por rango. Se determina si los valores de las dos muestras están mezclados aleatoriamente en el orden de rango o si están agrupados en los extremos opuestos al combinarse. Un orden aleatorio de rangos significa que las muestras no son diferentes, mientras que un grupo de valores de una muestra indicaría diferencia entre ellas. Para determinar la estadística de Mann-Whitney U para cada una de las muestras (considerando el menor de los estadísticos), se usa la siguiente fórmula:

$$U_i = n_1 n_2 + \frac{n_i(n_i + 1)}{2} - \sum R_i \quad (2.14)$$

donde  $U_i$  es la estadística de prueba,  $n_i$  es el número de valores de la muestra de interés,  $n_1$  el número de valores de la primera muestra,  $n_2$  el número de valores de la segunda muestra y  $\sum R_i$  es la suma de rangos de la muestra de interés. Conforme a [30], para la obtención del valor p con tamaños de muestra pequeños se consulta una tabla de valores críticos, pero si las muestras tienen un valor grande, se utiliza el cálculo del puntaje z y posteriormente se usa una tabla de distribución normal o software estadístico. El cálculo del puntaje z se define como:

$$z* = \frac{U_i - \bar{x}_U}{S_U} \quad (2.15)$$

donde  $\bar{x}_U$  es la media y  $S_U$  es la desviación estándar de la estadística  $U$ .

### 2.6.2. Prueba Kruskal-Wallis

De acuerdo con [30], la prueba  $H$  de Kruskal-Wallis es una prueba estadística no paramétrica para comparar más de dos muestras o grupos que son independientes. Cuando

la estadística de Kruskal arroja resultados significativos se considera que al menos uno de los grupos es diferente del resto de las muestras, aunque esta no identifica ni en cuantos ni en cuales de los grupos ocurre la diferencia por lo que es necesario implementar pruebas *post hoc* (prueba de Dunn por ejemplo).

Al realizar la prueba de Kruskal-Wallis  $H$  se examina las medianas de la población  $\theta_i$ . Para determinar la estadística se combinan todas las muestras y se ordenan conjuntamente los valores. La estadística se define [30]:

$$H = \frac{12}{N(N+1)} \sum_{i=1}^k \frac{R_i^2}{n_i} - 3(N+1) \quad (2.16)$$

Donde  $N$  es el número de valores de las muestras combinadas,  $R_i$  es la suma de los rangos de una muestra específica, y  $n_i$  es el número de valores de la suma de rangos correspondiente. Conforme a [30], una vez que se calcula la estadística de Kruskal-Wallis se puede comparar con una tabla de valores críticos para examinar los grupos en busca de diferencias. Si el número de grupos o la cantidad de datos es muy grande se utiliza la distribución  $\chi^2$  para obtener el valor crítico y realizar una aproximación para muestras grandes. Si el estadístico  $H$  es significativo entonces al menos un grupo o muestra es diferente al resto. Cuando se tienen múltiples grupos, es necesario realizar un ajuste para que el error tipo 1 no incremente, por ejemplo, se puede utilizar el ajuste de Bonferroni que se define, [30]:

$$\alpha_B = \frac{\alpha}{k} \quad (2.17)$$

Donde  $\alpha_B$  es el nivel de significancia ajustado,  $\alpha$  es el nivel de significancia original y  $k$  es el número de muestras o grupos.

## 2.7. Arquitectura del transformador

Antes del desarrollo del transformador, se usaban los modelos de redes recurrentes (RNN) y redes neuronales convolucionales (CNN) para el análisis de secuencias de texto en procesamiento de lenguaje natural.

El modelo original del transformador o *transformer* fue inventado por el equipo de investigación de *Google* en el año 2017 para resolver tareas de traducción automática, a través del trabajo de investigación “*Attention Is All You Need*” [4]. La innovación en el transformador es la eliminación de la recurrencia, y en su lugar se proponen los mecanismos de atención. El modelo del transformador permitió un procesamiento más paralelizable, ya que procesa todas las palabras de una secuencia de forma paralela, requiriendo menos

tiempo de entrenamiento que los modelos secuenciales [4]. Además, el uso de los mecanismos de atención en los transformadores ha sido clave para la resolución de algunos de los problemas en el lenguaje, ayudando a brindar mejores representaciones del contexto y resolver ciertos problemas de ambigüedad [19].

El modelo del transformador se puede observar en la figura 2.4.

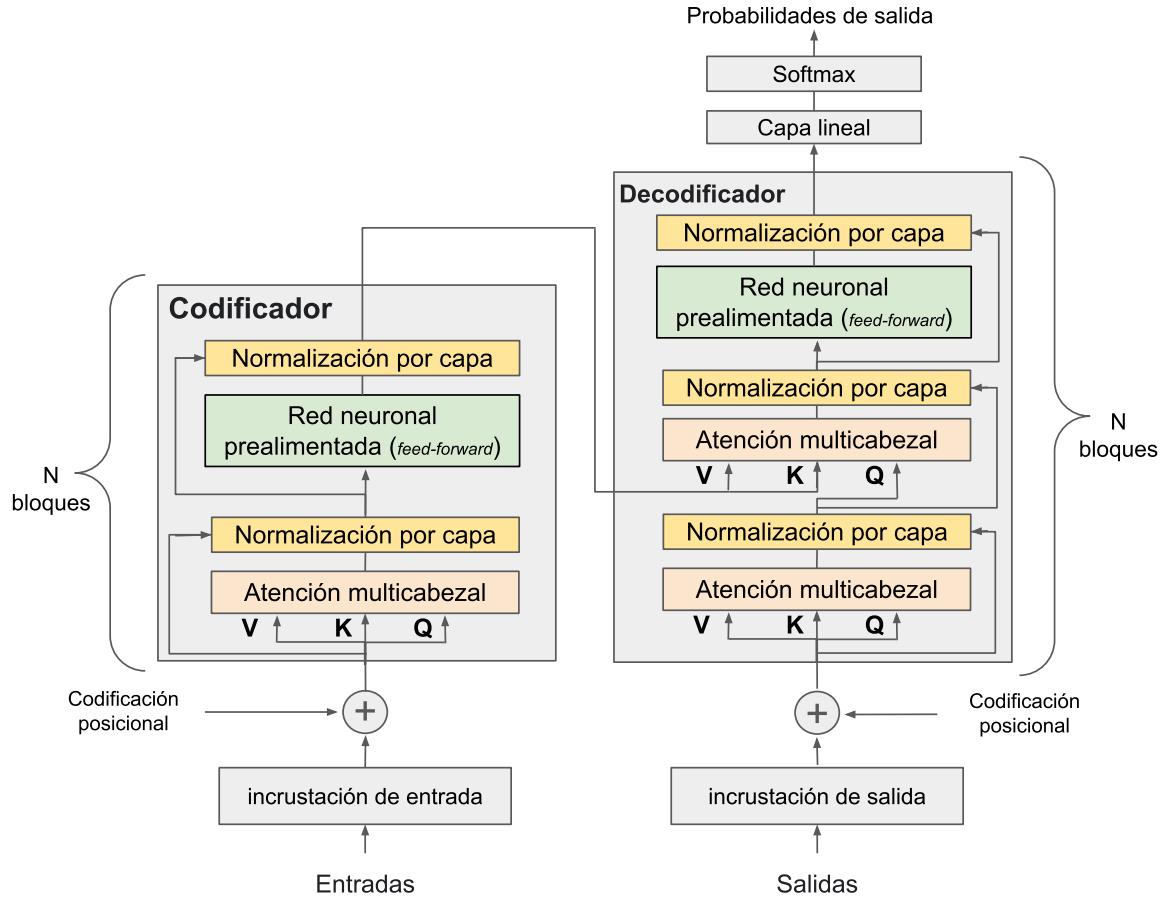


Figura 2.4: Arquitectura del modelo transformador. Figura construida conforme a [4].

### 2.7.1. Bloque codificador

En el modelo original del transformador, se apilan 6 capas codificadoras o *encoders*, es decir, que la salida de una capa *codificadora* es la entrada de la siguiente capa codificadora. El bloque codificador (lado izquierdo de la figura 2.4) está compuesto por dos subcapas principales: el mecanismo de atención multicabezal o *multi-head attention* y la subcapa de red de alimentación directa o *feedforward network*, completamente conectada

por posición [4]. En ambas subcapas se pueden apreciar conexiones residuales o *residual connection*, permitiendo que la información relevante de la entrada, como es la codificación posicional o *positional encoding*, fluya y no se pierda en el camino. Asimismo, se puede observar que a la salida de las subcapas de atención de múltiples cabezales y la red completamente conectada, se realiza una normalización, en específico, una normalización por capas o *layer normalization*, para proporcionar mayor estabilidad al gradiente y acelerar la convergencia.

La arquitectura de las 6 capas del bloque codificador es idéntica; sin embargo, su contenido no es idéntico a la capa anterior. Solo la primera capa (la capa inferior) recibe las incrustaciones o *embeddings*, agregándole la codificación posicional como entrada, mientras que el resto de las 5 capas no. El mecanismo de atención multicabezal aprende de la capa previa, y cada cabezal pone atención en diferentes asociaciones de palabras sobre la secuencia de entrada y determinará las relaciones más fuertes de una palabra entre todas las demás palabras [31].

Las salidas de cada capa en el modelo y la capa de *embedding* tienen una dimensión constante; para el caso del transformador original, se asignó un  $d_{model} = 512$ .

### 2.7.2. Bloque decodificador

En el bloque decodificador, al igual que en el bloque codificador, se apilan 6 capas decodificadoras, lo que implica que la salida de una capa *decodificadora* se convierte en la entrada de la siguiente capa *decodificadora*. La capa decodificadora consta de la subcapa de múltiples cabezales y la subcapa de la red completamente conectada, junto con sus respectivas normalizaciones por capa y conexiones residuales; sin embargo, se añade una subcapa de atención multicabezal intermedia. En esta subcapa, el decodificador presta atención a la oración codificada proporcionada por la salida del sexto codificador, basándose en la secuencia destino (traducida) proveniente de los *embeddings* de entrada del decodificador. Observa a qué partes de la oración codificada necesita prestarle atención, es decir, establecer pesos de relación entre las palabras del codificador y las palabras de la secuencia traducida del decodificador.

La diferencia entre la primera y la segunda (intermedia) atención multicabezal radica en que la primera subcapa realiza autoatención, es decir, solo observa las palabras que forman la secuencia de entrada objetivo en el decodificador y las relaciones de palabras entre sí, mientras que la subcapa intermedia lleva a cabo una atención más amplia, observando y relacionando las palabras entre la salida de la primera subcapa de atención multicabezal del decodificador y la salida de la capa codificadora. La composición de las subcapas del bloque decodificador se puede observar en el lado derecho de la figura 2.4.

La entrada de la secuencia objetivo, al igual que en el codificador, solo entra en la primera subcapa de la atención multicabezal del primer bloque decodificador, transformando las palabras de la secuencia a *embeddings* (con dimensión  $d_{\text{model}}$ ) y agregando la codificación posicional.

En la primera subcapa de la atención multicabezal del decodificador, se realiza un enmascarado (antes del cálculo de la autoatención) para garantizar que cada palabra en la secuencia objetivo solo ponga atención a las palabras que la preceden en la secuencia, es decir, evitando que el modelo haga trampa, impidiéndole ver partes futuras de la secuencia durante la decodificación en tareas de generación de secuencia, [31].

En el bloque decodificador, se utiliza la secuencia de salida verdadera como entrada en el paso de tiempo  $t + 1$  en lugar de la salida generada por el modelo en el paso de tiempo  $t$ , con el objetivo de mejorar la calidad de la predicción del modelo. Sin embargo, esta técnica solo se implementa durante la etapa de entrenamiento del modelo. Conocida como forzado del maestro o *teacher forcing* [32], esta estrategia es comúnmente empleada en modelos de generación de secuencias.

### 2.7.3. Atención de múltiples cabezales

La atención de múltiples cabezales se basa en la idea de los mecanismos de atención, los cuales se pueden pensar como una búsqueda en un diccionario en el que se tiene un conjunto de vectores de características o valores (*values*), y el modelo decide dinámicamente (de acuerdo a la entrada) qué valor usar, en función de cuán similar es el vector de consulta de entrada (*query*) a un conjunto de claves (*keys*) [19].

De acuerdo con el modelo original [4], la dimensión  $d_{\text{model}}$  se divide en  $h = 8$  cabezas. Para cada cabeza,  $d_k = d_v = d_{\text{model}}/h = 64$  dimensiones, donde  $d_{\text{model}} = 512$  (dimensión de la entrada o las salidas de cada capa codificadora/decodificadora en el modelo), el término  $d_k$  se refiere a la dimensión de las consultas y claves, mientras que  $d_v$  se refiere a la dimensión de los valores.

El tipo de atención implementada en [4] usa la atención de producto punto escalado o *scaled dot-product attention*:

$$\text{Attention}(Q, K, V) = \text{softmax} \left( \frac{QK^T}{\sqrt{d_k}} \right) V \quad (2.18)$$

Como se puede observar en la ecuación 2.18, las consultas **Q**, las claves **K** y los valores **V** se empaquetan en sus respectivas matrices.

En lugar de realizar una única operación de autoatención con claves, valores y consultas con dimensión  $d_{\text{model}}$ , es más beneficioso proyectar linealmente las consultas, claves y

valores  $h$  veces con proyecciones lineales diferentes, con dimensiones  $d_k$ ,  $d_k$  y  $d_v$ , respectivamente [4]. La intuición detrás de esto es que cada cabezal captura diferentes nociones de similaridad y relaciones entre las palabras que conforman la secuencia, [19]. La operación de atención se realiza de forma paralela en cada una de las versiones proyectadas de consultas, claves y valores.

La salida producida por cada cabezal tiene una dimensión  $d_v$  por lo que tienen que ser concatenadas de forma que podamos pasar por una última proyección, que da lugar a los valores de salida de la atención multicabezal. De esta manera, las dimensiones de los valores de la salida con los valores de la entrada en la capa de autoatención de múltiples cabezales son iguales.

La atención de múltiples cabezas se define en la ecuación 2.19 de acuerdo con [4]:

$$\text{MultiHead}(Q, K, V) = \text{Concat}(\text{head}_1, \dots, \text{head}_h)W_O \quad (2.19)$$

$$\text{head}_i = \text{Attention}(QW_i^Q, KW_i^K, VW_i^V) \quad (2.20)$$

Las proyecciones o matrices de pesos para la autoatención son  $W_i^Q \in \mathbb{R}^{d_{\text{model}} \times d_k}$ ,  $W_i^K \in \mathbb{R}^{d_{\text{model}} \times d_k}$ ,  $W_i^V \in \mathbb{R}^{d_{\text{model}} \times d_v}$  y  $W^O \in \mathbb{R}^{hd_v \times d_{\text{model}}}$  [4].

En la figura 2.5, se puede apreciar el diagrama de flujo que representa la estructura de la atención con múltiples cabezales (derecha en la figura). Se observa cómo la entrada ( $V$ ,  $K$  y  $Q$ ) pasa por una linealidad a través de las matrices de peso  $W^{Qi}$ ,  $W^{Ki}$  y  $W^{Vi}$ , donde el subíndice  $i$  representa al número de cabezal  $h$ . La salida de la linealidad de las proyecciones de los valores, claves y consultas da lugar a las nuevas submatrices  $V$ ,  $K$  y  $Q$  respectivamente, con dimensión  $dk$  para permitir la operación de atención por producto punto escalado (izquierda en la figura), que los cabezales realizan de forma paralela. Las salidas de la atención del producto punto escalado se concatenan para regresar a su dimensión inicial  $d_{\text{model}}$  y pasar por una última linealidad a través de la matriz  $W^O$ .

La subcapa intermedia de atención de múltiples cabezales en el decodificador extrae información del codificador teniendo en cuenta sus claves y valores ( $K$ ,  $V$ ) durante las operaciones de atención del producto escalar. Además, extrae información de la primera subcapa de atención multicabezal enmascarada (atención enmascarada) del decodificador, considerando sus consultas ( $Q$ ) durante las operaciones de atención del producto escalar [31]. En cambio, la primera subcapa de autoatención de múltiples cabezales en el bloque codificador extrae información de sus propias consultas, claves y valores ( $Q$ ,  $K$ ,  $V$ ) durante las operaciones de autoatención. Este mismo proceso ocurre para la primera subcapa de autoatención de múltiples cabezales del bloque decodificador.

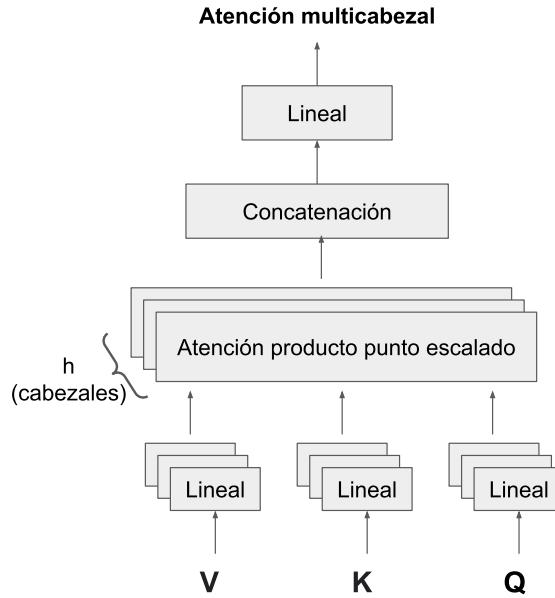


Figura 2.5: Atención de múltiples cabezales, figura modificada de [4].

#### 2.7.4. Normalización

En general, realizar una normalización en los modelos de aprendizaje profundo acelera el entrenamiento del modelo, haciendo que la red sea más estable durante el proceso. Esto permite que la red pueda ser entrenada con tasas de aprendizaje más altas y que se requiera menos precaución en la inicialización de los parámetros [33].

En el modelo Transformer, tanto la salida de la subcapa de atención como la salida de la subcapa completamente conectada o *feedforward* hacen uso de una subcapa de normalización. Esta subcapa de normalización utiliza una conexión residual que proviene de la entrada de la subcapa e implementa la normalización por capas o *layer normalization* [31]. La conexión residual ayuda a que la información no se pierda, y la normalización por capas se ajusta bien debido a la naturaleza secuencial del texto.

La subcapa de normalización en el modelo Transformer original se puede describir de la siguiente manera [31]:

$$\text{LayerNorm}(x + \text{SubLayer}(x)) \quad (2.21)$$

Donde  $\text{SubLayer}(x)$  es la subcapa en sí misma y  $x$  es la información disponible en el paso de entrada de  $\text{SubLayer}(x)$ . La entrada a  $\text{LayerNorm}$  es el vector  $v = x + \text{SubLayer}(x)$  y puede ser definida como:

$$\text{LayerNorm}(v) = \gamma \frac{v - \mu}{\sigma} + \beta \quad (2.22)$$

Donde  $\mu$  es la media de  $v$ , dada por [34]:

$$\mu^l = \frac{1}{H} \sum_{i=1}^H v_i^l \quad (2.23)$$

Donde  $H$  denota el número de unidades ocultas en la capa, que para el caso del modelo de transformador original sería  $H = d_{\text{model}}$  y  $l$  denota el número de capa.

La desviación estándar  $\sigma$  está dada por [34]:

$$\sigma^l = \sqrt{\frac{1}{H} \sum_{i=1}^H (v_i^l - \mu^l)^2} \quad (2.24)$$

Donde  $\lambda$  es el parámetro de reescalado o *scaling* y  $\beta$  es el parámetro de desplazamiento o *shift*, los cuales son útiles para permitir que el modelo aprenda la distribución de los datos y ajuste los datos normalizados a la distribución deseada [33].

### 2.7.5. Red completamente conectada basada en posición

La función de la red completamente conectada o *feedforward network* (FFN) dentro del modelo transformador es aplicar una transformación no lineal a la salida de la atención y permitir que la red aprenda representaciones de características abstractas.

La entrada y salida de la capa *feedforward* en el modelo original del transformador tienen una dimensión  $d_{\text{model}} = 512$ , y la capa interna tiene una dimensionalidad  $d_{\text{ff}} = 2048$ . La subcapa *feedforward* en el transformador permite que la red capture patrones de posición locales en la secuencia de entrada. La segunda capa utiliza una transformación lineal y una función de activación de unidad lineal rectificada (*rectified linear unit*, ReLU) para procesar las características de la secuencia. Teniendo en cuenta esta descripción, podemos expresar la subcapa *feedforward* [4]:

$$FFN(x) = \max(0, xW_1 + b_1)W_2 + b_2 \quad (2.25)$$

### 2.7.6. Incrustaciones de entrada

Dada una sentencia o secuencia de entrada, cada palabra o token que conforma la secuencia se transforma en un vector de dimensión  $d_{\text{model}}$ . Para realizar este proceso, se lleva a cabo una normalización del texto y una tokenización para convertir la sentencia

de entrada en tokens. A cada token se le asigna un identificador numérico del tipo entero que se utilizará en el proceso de incrustación o *embedding*. El transformador contiene una subcapa de incrustación aprendida, en la cual se pueden aplicar varios métodos de incrustación a la entrada tokenizada, como *skip-gram*, *continuous bag of words model* (CBOW) de Word2Vec, de esta manera, las incrustaciones proporcionarían información al transformador sobre cómo se relacionan las palabras [31].

### 2.7.7. Codificación posicional

Dado que el transformador no contiene recurrencia y procesa los tokens de las secuencias de una forma paralelizable, se inyecta información sobre la posición de los tokens en la secuencia a través de la codificación posicional o *positional encoding* en las incrustaciones o *embeddings* de palabras de entrada, las cuales tienen una dimensión  $d_{\text{model}}$ , de modo que se pueden sumar. Hay muchas opciones de codificaciones posicionales, como el aprendido y fijo, los cuales pueden producir resultados similares. Para el caso del transformador original, se planteó una forma fija a través de funciones seno y coseno con diferentes frecuencias [4].

### 2.7.8. Capa final

El modelo original del transformador produce una secuencia de salida de un solo elemento a la vez [31]. Por lo tanto, a la salida del decodificador se obtiene un vector de dimensión  $d_{\text{model}}$  que se debe mapear a una palabra. Para realizar esto, se implementa una capa lineal que producirá los pesos (valores numéricos) o *logits* con una dimensión del tamaño del vocabulario (palabras únicas del corpus). La salida de la capa lineal pasa por una función softmax para convertir los pesos en probabilidades. Se elige la probabilidad más alta y se proporciona como salida la palabra asociada a esa probabilidad. Este proceso se repite para cada paso de tiempo (cada palabra) de la secuencia. La capa final se puede observar en el lado superior derecho de la figura 2.4.

### 2.7.9. Complejidad computacional de la autoatención del transformador

Los mecanismos de atención del modelo transformador permitieron una mayor paralelización. De acuerdo con [4] la capa de autoatención es más rápida que la recurrencia de los modelos predecesores (RNN) del transformador cuando la longitud de la secuencia  $n$  es menor que la dimensionalidad de la representación  $d$ . Para un mecanismo de autoatención su complejidad por capa es de  $O(n^2 \times d)$ , donde  $n^2$  proviene de la multiplicación de

la matriz de puntuaciones de atención (se calculan los pesos de atención entre todos los tokens de entrada) y  $d$  que corresponde a la dimensión de los vectores de representación del modelo transformador, mientras tanto la complejidad por capa para una red recurrente de acuerdo con [4], se aproxima a  $O(n \times d^2)$  ya que para cada paso de tiempo, existe la multiplicación de las matrices  $W_{xh}$ ,  $W_{hh}$  y  $W_{ho}$  donde [4] aproxima  $O(d^2)$ , entendiendo que  $d$  (dimensión del estado oculto  $h$ ) en la práctica suele ser una dimensión mucho más grande que  $n$  en las redes recurrentes, y si adicionalmente consideramos que para una secuencia tenemos  $n$  tokens o  $n$  operaciones secuenciales, obtenemos  $O(n \times d^2)$ . La cantidad de operaciones secuenciales para el caso de la autoatención requiere un número constante de operaciones ejecutadas secuencialmente, debido a que las operaciones de la auto atención se realizan en paralelo y las dimensiones de las matrices no cambian con la longitud de la secuencia, mientras que para el caso del modelo recurrente se debe calcular la salida de la red para cada uno de los  $n$  elementos de la secuencia, debido a que cada salida depende de las entradas anteriores.

## 2.8. BERT

El modelo de representación de codificador bidireccional de transformadores BERT en inglés, conocido como *Bidirectional Encoder Representations from Transformers*, está diseñado para entrenar previamente representaciones bidireccionales de texto no etiquetado, considerando el contexto izquierdo y derecho en todas las capas [5]. El modelo BERT es de tipo no causal, es decir, se utiliza para crear representaciones de texto pero no para generar texto [19].

BERT es un modelo que puede analizarse en dos fases. La primera fase es la de preentrenamiento con aprendizaje no supervisado, y la segunda fase es conocida como ajuste fino o *fine-tuning* a través del aprendizaje supervisado. Este modelo es bastante versátil, ya que gracias a esta separación de fases, se puede realizar una transferencia de aprendizaje y explotar más la fase de ajuste fino para resolver una amplia variedad de tareas de procesamiento del lenguaje natural, incluida la similitud semántica. La fase de preentrenamiento y la fase de ajuste fino se pueden observar en la figura 2.6.

En [5], se dan cuenta de que las técnicas hasta ese entonces desarrolladas (año 2019) restringían la potencia de las representaciones preentrenadas, dado que los modelos de lenguaje eran unidireccionales, limitando la elección de arquitecturas a usar durante el entrenamiento previo. En el transformador de [4], por ejemplo, la secuencia de palabras que se van procesando en la autoatención del decodificador era de izquierda a derecha, donde cada token solo ponía atención a los tokens previos. Estas limitaciones podrían ser subóptimas para tareas en donde es crucial incorporar contexto de ambas direcciones. El

modelo de BERT se vuelve bastante poderoso, porque en [5], el contexto de las representaciones se toma de forma bidireccional, es decir, de izquierda a derecha y de derecha a izquierda.

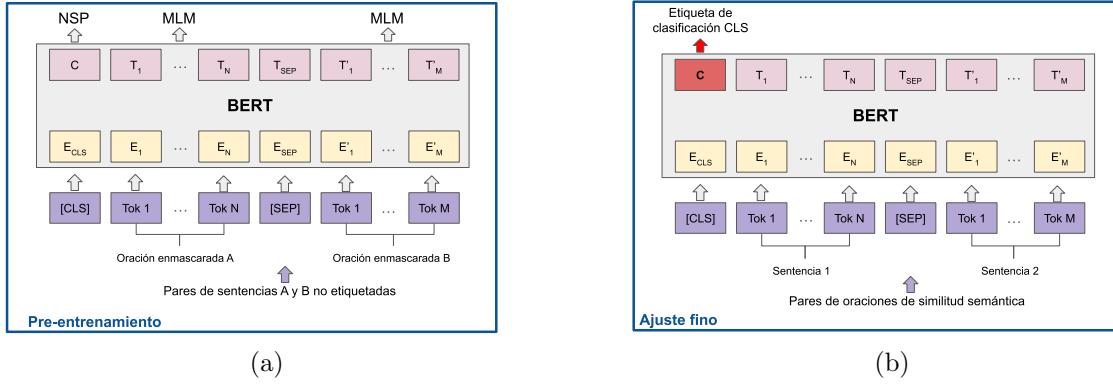


Figura 2.6: Pre-entrenamiento (izquierda) y ajuste fino (derecha) del modelo BERT. Figura construida conforme a [5].

### 2.8.1. Arquitectura del modelo BERT

La arquitectura del modelo BERT en [5] consiste en la apilación de bloques codificadores o *encoders* basados en la implementación original del transformador de [4].

Existen diferencias en cuanto a la configuración de hiperparámetros entre [4] y [5], ya que en BERT se plantean dos configuraciones: *BERT<sub>base</sub>* y *BERT<sub>large</sub>*. El *BERT<sub>base</sub>* tiene  $L = 12$  capas codificadoras,  $H = 768$  estados ocultos,  $A = 12$  cabezales de atención, mientras que el modelo *BERT<sub>large</sub>* tiene  $L = 24$ ,  $H = 1024$ , y  $A = 24$ .

En la entrada del modelo se usan incrustaciones de piezas de palabras “*WordPiece embeddings*” (segmentación de palabras para dividir palabras en subunidades más pequeñas llamadas subpalabras [35]) con un vocabulario de 30,000 tokens.

### 2.8.2. Pre-entrenamiento del modelo BERT

Durante el preentrenamiento, el modelo se entrena con datos no etiquetados (aprendizaje no supervisado).

En [5], el modelo de BERT realiza dos tareas. La primera es el uso del modelo de lenguaje enmascarado (MLM) para lograr representaciones bidireccionales. La segunda tarea del modelo BERT consiste en usar una tarea de predicción de la siguiente oración que preentrena conjuntamente las representaciones de pares de texto.

### Modelo de lenguaje enmascarado

De acuerdo con [5], el modelo de lenguaje enmascarado (MLM) consiste en enmascarar aleatoriamente un porcentaje del total de los tokens de entrada WordPiece (15 %) y luego predecir el término original enmascarado basándose en su contexto (también conocido como tarea Cloze).

En BERT, solo se predice las palabras enmascaradas en lugar de reconstruir la entrada completa. Sin embargo, esto genera un desajuste entre el preentrenamiento y el ajuste fino porque el token [MASK] no aparece en el ajuste fino. Para mitigar este problema, en realidad no siempre se reemplazan las palabras enmascaradas con el token [MASK]. El generador de datos de entrenamiento del 15 % de las palabras a enmascarar las reemplaza el 80 % del tiempo con [MASK], el 10 % del tiempo con un token aleatorio y el otro 10 % del tiempo con el token original.

### Predicción de la siguiente oración

El modelo de BERT original agrega el objetivo de clasificar si una oración sigue a otra, y así comprender las relaciones entre oraciones. El modelo tiene una entrada con el siguiente aspecto [19]:

$$[CLS][A1][A2]; \dots; [Am][SEP][B1][B2]; \dots; [Bn][SEP] \quad (2.26)$$

Donde SEP es un token separador y CLS es un token para marcar la clase. Si la oración B sigue a A, lo entrenamos para una tarea binarizada. Este tipo de entrenamiento previo puede ser útil para tareas de clasificación de pares de oraciones, como la implicación textual o la similitud textual [19].

Para determinar si una oración sigue a otra, la entrada al modelo se especifica usando la suma de tres incrustaciones diferentes: una por token, una para indicar el segmento (oración A o B), y una por ubicación (incrustación posicional aprendida). Luego, BERT aprende un mapeo de la secuencia de incrustación de entrada a una secuencia de incrustación de salida, que se decodifica en etiquetas de palabras (para las ubicaciones enmascaradas) o una etiqueta de clase (para la ubicación de CLS) [19].

#### 2.8.3. Ajuste fino del modelo BERT

Una vez se ha preentrenado el modelo BERT, se puede usar para varias tareas haciendo un ajuste fino de forma supervisada. Durante el ajuste fino, los pesos del modelo BERT preentrenado se ajustan para que se adapten mejor a la tarea en cuestión.

En la figura 2.7 se puede ver cómo abordar la tarea de similitud semántica entre pares de oraciones, las cuales se ingresan a la entrada con el formato de 2.26. Posteriormente, se implementa un modelo muy sencillo como la regresión a partir del token de clasificación CLS.

En la figura 2.7(a), se puede ver cómo realizar una clasificación a partir de una sentencia a través del token CLS. Un ejemplo de aplicación sería el análisis de sentimientos. En la figura 2.7(b), se puede ver la clasificación de pares de oraciones, las cuales se ingresan a la entrada, con el formato de 2.26 y se clasifican a partir del token CLS. Un ejemplo de uso sería la vinculación textual. La figura 2.7(c) muestra cómo abordar el etiquetado de una sola oración (asociación de una etiqueta con cada palabra), como ejemplo de aplicación sería el etiquetado de las partes del discurso (*part of speech tagging*), en el que se etiqueta cada palabra como un sustantivo, verbo, adjetivo, etc. Finalmente, en la figura 2.7(d), se muestra cómo abordar la respuesta a preguntas, donde la primera oración de entrada es la pregunta, la segunda es el texto de fondo y la salida se requiere para especificar las ubicaciones de inicio y final de la parte relevante del texto de fondo que contiene la respuesta [19].

#### 2.8.4. Antecedentes del modelo BERT

En marzo de 2018, [36] introduce una representación de palabras contextualizadas profundas conocida como *Embeddings from Language Models* (ELMo), que abstrae características complejas del lenguaje, como la sintaxis y la semántica. Estas representaciones se aprendieron de forma no supervisada, con una representación de entrada basada en caracteres que es procesada y contextualizada por una red neuronal convolucional (CNN) para luego alimentar los datos, junto con texto etiquetado, a una red LSTM bidireccional de 2 capas. Hasta entonces (2018), ya existían modelos de contexto en una dirección como Word2Vec y GloVe. Sin embargo, la innovación y el poder de ELMo consistieron en modelar contextos bidireccionales, es decir, abstraer un contexto de izquierda a derecha y de derecha a izquierda. Luego, se combinan sus representaciones profundas de estado oculto (la combinación de representaciones tomadas en capas internas) y se crea una incrustación para cada palabra tomando la concatenación de las representaciones en cada dirección [19]. Se probó el rendimiento de ELMo en 6 tareas de procesamiento de lenguaje natural, obteniendo entre un 6 % y 20 % de reducción de error, así como una mejora de entre 0.7 % y 4.7 % en las métricas F1 y precisión.

Para junio de 2018, [37] de OpenAI desarrollaron *Generative Pre-Training* conocido como GPT. En este trabajo se propone un procedimiento de entrenamiento en dos etapas: una primera etapa de preentrenamiento generativo no supervisado de un modelo de len-

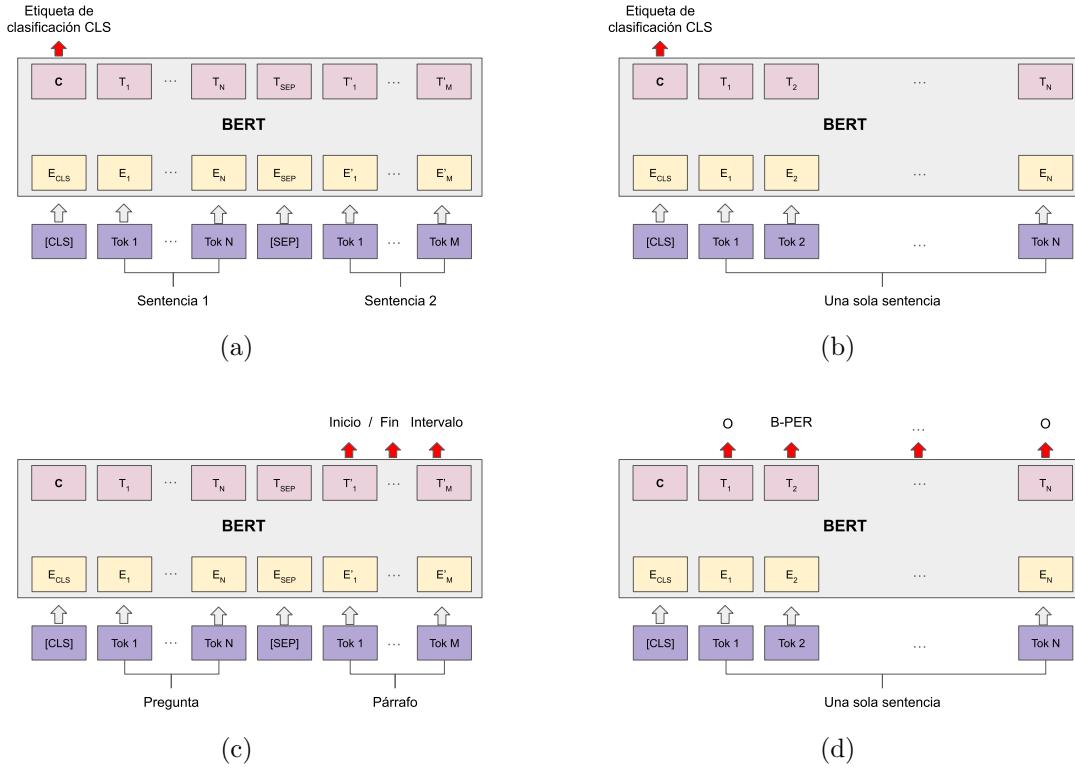


Figura 2.7: Ajuste fino de BERT para diferentes tareas de procesamiento de lenguaje natural de forma supervisada. Figura construida conforme [5].

guaje usando el corpus de texto *BooksCorpus* de forma no etiquetada, y una segunda etapa de ajuste fino *fine-tuning* discriminativo supervisado. La esencia es una arquitectura muy sencilla y con cambios mínimos que busca resolver una tarea objetivo de procesamiento de lenguaje natural. En el trabajo de [37], usaron una arquitectura basada en el modelo transformador propuesto por [4] pero usando solo la parte decodificadora. La arquitectura de GPT consiste en 12 bloques decodificadores que contienen 2 subcapas: la subcapa de autoatención enmascarada (con estados de 768 dimensiones y 12 cabezales de atención) y la subcapa de *feed forward position wise*. De todos los experimentos que se realizaron, lograron mejoras significativas en 9 de 12 conjuntos de datos para varias tareas de lenguaje natural (respuesta a preguntas, clasificación, similitud semántica, inferencia, etc.). En su análisis, observaron el impacto de la transferencia de aprendizaje con un número variable de capas, concluyendo que cada capa del modelo aportaba alguna funcionalidad útil para las tareas de lenguaje natural.

Unos meses después del desarrollo de ELMo y de GPT, se implementa el modelo

BERT de [5]. Al igual que ELMo, obtiene representaciones bidireccionales profundas (de izquierda a derecha y de derecha a izquierda) pero sin texto etiquetado. El trabajo de [5] junta lo mejor de ELMo y GPT de acuerdo con [38], ya que mantiene representaciones de contexto bidireccional y la idea del ajuste fino inspirado por GPT, requiriendo así pocos cambios de arquitectura para las tareas de procesamiento de lenguaje natural.

La arquitectura de BERT está basada en el transformador de [4], pero solo incorpora bloques codificadores. En el trabajo de [5] se propusieron 2 arquitecturas: la arquitectura base que consiste en 12 bloques codificadores, un tamaño oculto de 768 y 12 cabezales de autoatención, sumando 110 millones de parámetros. Por otro lado, está la arquitectura grande, la cual consiste en 24 bloques codificadores, tamaño oculto de 1024 y 16 cabezales de autoatención, con un total de 340 millones de parámetros. Ambas arquitecturas de BERT (base y grande) superaron todas las tareas de comprensión de lenguaje del benchmark *General Language Understanding Evaluation* (GLUE) con un aumento promedio del 4,5 % y 7,0 %.

# Capítulo 3

## Estado del arte

En la primera sección de este capítulo, se exponen trabajos relacionados con la resolución de la similitud semántica utilizando el modelo BERT y el token de clasificación CLS.

La segunda sección incluye trabajos que han examinado el modelo BERT mediante técnicas como la poda de cabezales o pesos poco significativos.

En la tercera sección, se abordan investigaciones que han buscado proporcionar interpretabilidad lingüística al modelo BERT, mayormente a través del enfoque de aprendizaje supervisado mediante el uso de clasificadores independientes conocidos como sondas. Estas sondas se utilizan para brindar interpretabilidad a través de tareas lingüísticas específicas, y es importante destacar que existen dos tipos de sondas: las basadas en rendimiento y las basadas en atención. Estas pruebas de sondeo comúnmente se realizan utilizando las representaciones internas de BERT a nivel de capa.

### 3.1. Similitud semántica a través del token CLS

Con la introducción de BERT, surgieron investigaciones que se esforzaron por presentar progresos en el campo de la similitud semántica. Estas investigaciones, en su mayoría, se centran en la utilización de la representación del token de clasificación (CLS).

Uno de los trabajos más influyentes fue el propuesto por [39]. En este trabajo se menciona que la estructura del modelo BERT no favorece la búsqueda de mayor similitud semántica entre pares de oraciones, ya que ambas oraciones se alimentan a la red. Encontrar el par de oraciones más similares requeriría realizar un barrido de posibles combinaciones y cálculos de inferencia, lo que resultaría en un alto costo computacional. Por ello, se propone obtener las incrustaciones de cada oración de forma separada a través de BERT, para después aplicar operaciones de agrupamiento, como la agrupación del tipo

promedio (*mean-pooling*) o el valor máximo (*max-pooling*). Una vez realizada la operación de agrupación a la salida del modelo, los vectores fijos se pasan a través de una red siamesa para obtener incrustaciones que capturen la semántica de las oraciones, cuyo resultado se compara con alguna medida como la similitud coseno. Cabe señalar que en este trabajo se plantearon tres funciones objetivo: para clasificación, regresión y tripleta, con el fin de adaptarse a diferentes tipos de tareas. Para mejorar el modelo y entender las relaciones semánticas entre oraciones, en el trabajo de [39] se realizó un ajuste fino con la ayuda del conjunto de datos NLI (usado para tareas de inferencia y relaciones semánticas).

Los resultados de [39] muestran que introducir sentencias individuales al modelo BERT y tomar directamente el vector CLS o promediar la salida del modelo produce incrustaciones deficientes, incluso peores que las incrustaciones de GloVe. Realizar un entrenamiento con el conjunto de datos NLI para después tomar directamente el vector CLS y el promedio de la salida mejoró considerablemente los resultados. Entrenar con el conjunto de datos NLI y con el conjunto STS-Benchmark brindó los mejores resultados.

En el trabajo de [40] se aborda un enfoque clínico de la similitud semántica mediante el uso de la aumentación de datos con el conjunto N2C2. Sin embargo, también mantienen experimentos en los conjuntos clásicos de STS-Benchmark y SICK-R.

Implementan un modelo del tipo jerárquico convolucional para evaluar la similitud textual semántica. La arquitectura incluye una convolución jerárquica para capturar interacciones globales. Esta implementación dio mejores resultados entre 0.1 y 0.6 puntos en las métricas de Spearman y Pearson, en comparación con solo evaluar los conjuntos mediante el token CLS. Otro experimento interesante fue la aumentación de datos mediante el reordenamiento de segmentos (SR) y traducción inversa (BT). En SR, se reorganizan segmentos de texto independientes en el orden original, mientras que en BT se utiliza la traducción automática del inglés al chino y luego al inglés nuevamente para crear variaciones adicionales. Estas variaciones lograron mejoras de hasta 2.5 puntos en la métrica de Spearman.

### 3.2. Interpretabilidad del modelo BERT a través de poda

En la investigación de [41], trabajaron con el transformador original observando que los roles que juegan las cabezas de atención son interpretables. Implementaron la poda de cabezales utilizando un método de compuertas estocásticas, norma L0 y relajación diferenciable. Los resultados mostraron que los cabezales que desempeñan un papel relevante fueron los últimos en ser podados. Dentro de las tareas principales, encontraron cabezales

posicionales (atención a palabras vecinas), cabezales sintácticos (tokens con relaciones de dependencia sintáctica específicas) y cabezales con atención a palabras raras.

De acuerdo con [41], se observó que incluso con pocos cabezales, el modelo continúa siendo efectivo (se podaron hasta 3/4 de los cabezales de atención del codificador y más de 1/3 de los cabezales del decodificador y del codificador-decodificador), y algunos cabezales incluso comienzan a desempeñar más de una función. Estos resultados se obtuvieron sobre un modelo que fue entrenado previamente con todos los cabezales. Se observó una mayor degradación al realizar una poda desde cero.

Por su parte, en [42], proponen un algoritmo greedy para la poda iterativa de cabezales de atención que contribuyen menos al modelo. Mediante pruebas de ablación, destacan que muchas de las cabezas son redundantes. Sus resultados indican que se pueden eliminar hasta un 40 % de los cabezales de BERT (para la tarea de inferencia de lenguaje) sin sufrir pérdidas significativas.

Otro enfoque de poda en el modelo BERT es el realizado por [43], mediante la eliminación de los pesos menos significativos. Este proceso de poda se aplicó en las matrices proyectadas (“keys”, “values” y “queries”), en la proyección lineal de salida y en las incrustaciones de palabras, mediante la definición de un umbral. En general, la poda se implementó de manera gradual, desde un 0 % hasta un 90 %, en incrementos de 10 %.

Para evaluar el impacto de la poda de [43], se utilizó el conjunto GLUE (*General Language Understanding Evaluation*). Mediante estos experimentos, se evidencia que la aplicación de bajos niveles de poda (30 - 40 %) no repercute en el rendimiento del pre-entrenamiento ni en la transferencia de conocimientos a tareas específicas. Además, se señala que ciertas tareas son más sensibles a la poda que otras. Por ejemplo, la tarea de similitud entre dos oraciones fue menos sensible que la tarea para determinar si una oración es gramaticalmente correcta; una posible explicación de esta disparidad podría ser la cantidad de ejemplos en el conjunto de datos.

### 3.3. Interpretabilidad de aspectos lingüísticos en el modelo BERT

En el análisis realizado por [10], se realizaron extracciones de fragmentos de oraciones, conocidos como “spans” o sintagmas, utilizando BERT base. Se empleó un conjunto con datos fragmentados, y se obtuvieron representaciones para cada capa del modelo. Se tomaron tanto el primer como el último vector oculto de los tokens del “span”, y se realizaron concatenaciones tanto del producto como de la diferencia entre ellos. Luego, aplicaron un algoritmo de reducción de dimensionalidad a las representaciones de los

“spans” y, utilizando el algoritmo de agrupamiento k-means, generaron visualizaciones de las agrupaciones resultantes, tales como frase proposicional (PP), frase verbal (VP), frase adjetival (ADJP), frase nominal (NP), frase adverbial (ADVP), frase subordinada (SBAR), partícula (PRT), frase conjuntiva (CONJP) y otras (O). A partir de este experimento, concluyeron que las capas inferiores capturan información a nivel de frase (agrupaciones más definidas), mientras que en las capas superiores, la información se disipa gradualmente (agrupaciones se superponen).

En un segundo experimento por [10], se emplea una sonda de rendimiento para predecir 10 tipos de tareas lingüísticas en 10 conjuntos de datos distintos, utilizando la herramienta SentEval. Las tareas lingüísticas se agruparon en tres categorías: superficial, sintáctica y semántica. Dentro de la categoría superficial se encuentran el tamaño de las oraciones (SentLen) y el contenido de las palabras en la oración (WC). En las tareas sintácticas se incluyen la sensibilidad al orden de las palabras (BShift), la profundidad del árbol sintáctico (TreeDepth) y la secuencia de los componentes de nivel superior en el árbol de sintaxis (TopConst). Las tareas semánticas engloban la evaluación del tiempo (Tense), el número del sujeto (SubjNum), el número del objeto (ObjNum), la sensibilidad a la sustitución aleatoria de un sustantivo/verbo (SOMO) y el intercambio aleatorio de conjunciones coordinadas de cláusulas (CoordInv). A partir de estos experimentos, observaron que las capas inferiores codifican información de superficie, mientras que las capas intermedias contienen información sintáctica y las capas superiores albergan información semántica.

A través de una prueba de concordancia de número y persona del verbo con el sujeto de la oración, se pudo observar que BERT necesita capas más profundas para abordar con éxito la dependencia a larga distancia en sentencias extensas. Finalmente, mediante una red de Descomposición de Productos Tensoriales (TPDN), descubrieron que BERT implementa un esquema basado en estructura de árbol. Esto significa que BERT es capaz de capturar información lingüística jerárquica en sus representaciones.

En el estudio realizado por [8], sostienen que el modelo BERT desempeña tareas convencionales de procesamiento del lenguaje natural en una forma interpretativa, localizable, en forma secuencial (tubería). Para evaluar el comportamiento del modelo BERT, implementan una sonda que accede a los vectores contextualizados por token y busca predecir etiquetas que incluyen aspectos como constituyentes y tipos de relaciones lingüísticas. Las etiquetas utilizadas en este estudio abarcan parte del discurso (POS), constituyentes, dependencias, entidades, roles semánticos en el etiquetado (SRL), coreferencia, roles semánticos prototípico (SPR) y clasificación de relaciones.

La sonda utilizada en [8] es basada en atención, por lo que cada representación contextual generada por una capa del modelo BERT es multiplicada por un peso escalar

específico. Estos pesos son entrenados en conjunto con la sonda, y después de entrenarla, extraen los pesos escalares para estimar la contribución de cada capa. A mayor peso, mayor importancia tiene una capa para una tarea en particular. Identifican los pesos más influyentes para determinar la capa más relevante. Encuentran que la tarea de etiquetado de partes del discurso (POS) es procesada en las etapas iniciales, seguida por la identificación de constituyentes, las dependencias sintácticas, los roles semánticos y, por último, la resolución de la coreferencia. Se ha destacado que la información sintáctica elemental emerge en las primeras capas del modelo BERT, mientras que los aspectos semánticos más avanzados se manifiestan en capas superiores.

En [8] se identificó que la información sintáctica tiende a concentrarse en un rango limitado de capas, mientras que la información semántica se extiende a lo largo de toda la estructura de la red. En la mayoría de los casos, la capacidad para clasificar los ejemplos de manera correcta se establece desde etapas tempranas. Se ha atribuido este fenómeno a ciertos atajos inherentes en BERT, donde los ejemplos más desafiantes se resuelven en etapas posteriores del proceso. También sostienen que el orden de la tubería tradicional de procesamiento de lenguaje natural se mantiene en BERT, aunque se menciona que la red es capaz de abordar problemas fuera de orden.

Luego, el estudio realizado por [6] no niega la existencia de un fundamento lingüístico intrínseco en el modelo BERT, pero difiere en la forma en que [10] y [8] sugieren que se manifiesta. En [6] se llevaron a cabo ajustes en los experimentos propuestos por [10], empleando la correlación de rango de Kendall y cálculo de suma lineal. A partir de estas adaptaciones, sostienen que las tareas semánticas alcanzan su máximo rendimiento en capas intermedias (6-9), que la tarea de superficie es distingible, mientras que las capas destinadas a las tareas de sintaxis y semántica resultan indistinguibles en términos de rendimiento.

En [6] se retoma el análisis de [8] y lo complementa empleando el coeficiente de correlación de rangos de Kendall y la divergencia de Kullback-Leibler (KL). Se calculan las puntuaciones de diferencia más altas, se ordenan los rangos de las puntuaciones de diferencia y se calcula el coeficiente de Kendall en relación a la secuencia ordenada. Los resultados obtenidos mediante este proceso demuestran que la información superficial, sintáctica y semántica se distribuye en paralelo a lo largo de las capas, siendo la información semántica la que se dispersa de manera más uniforme.

Por otro lado, [6] propone una sonda de atención denominada GridLoc para validar 10 tareas de lenguaje a través del conjunto SentEval. En GridLoc se aplicó un mecanismo de atención para asignar pesos a cada fuente de entrada y luego se utilizó un clasificador basado en estos pesos de atención asignados. La arquitectura de GridLoc involucró un modelo recurrente RNN. Este modelo utilizó las incrustaciones generadas por BERT de

una capa específica para generar un peso de atención individual para cada token en una oración. Posteriormente, la representación resultante se somete a otro proceso de agrupación mediante autoatención, lo que a su vez produce un peso de atención para la capa. Este proceso se repite para cada capa de representación de BERT. Finalmente, se entrena un clasificador de perceptrón multicapa (MLP) que utiliza las representaciones combinadas como entrada y genera las predicciones para la tarea en cuestión. En GridLoc, se exploró la variación de los pesos de atención con diferentes semillas aleatorias y en diversas iteraciones de entrenamiento.

GridLoc valida las afirmaciones de que las tareas de superficie se desarrollan en capas bajas o tempranas, mientras que las tareas sintácticas y semánticas están intrínsecamente relacionadas y se extienden a través de las capas intermedias y superiores, ver figura 3.1. Este hallazgo sugiere que las capas individuales de BERT no encapsulan completamente la supuesta tubería de procesamiento de [8].

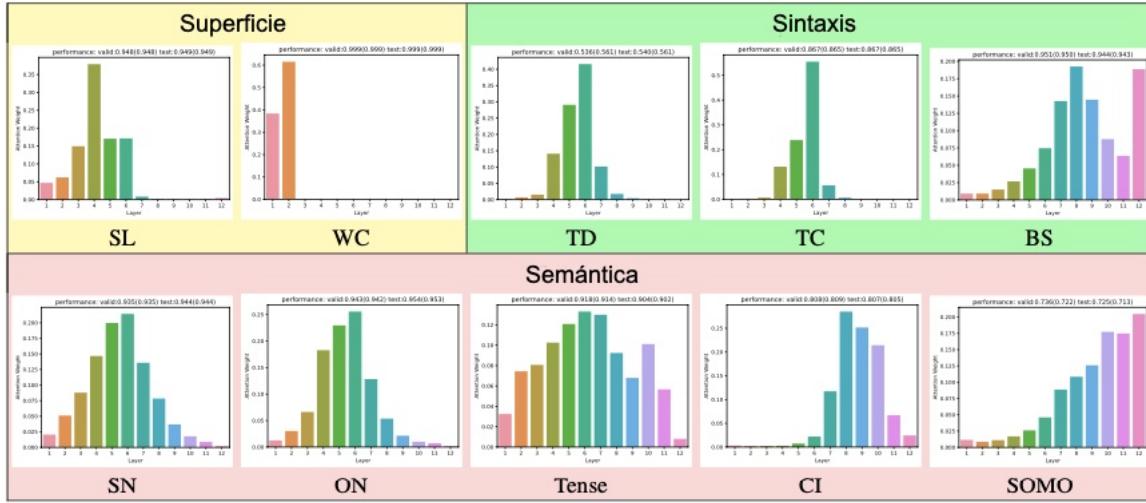


Figura 3.1: Distribución del peso de atención de la capa promedio de GridLoc para cada tarea de SentEval, demostrando que las tareas de superficie tienen capas distinguibles, pero las tareas sintácticas y semánticas son indistinguibles. Figura modificada de [6].

Por otro lado, en el trabajo de [7] propusieron una sonda estructural para demostrar que los árboles de sintaxis pueden estar incrustados implícitamente. La sonda estructural se basa en una transformación lineal de los vectores de palabras del modelo. Utiliza medidas de distancia y norma al cuadrado en el espacio de representación transformado para evaluar si se conservan las relaciones sintácticas entre las palabras. Se utilizaron los modelos de representación con pesos preentrenados de BERT con el conjunto de datos de Penn TreeBank. Para evaluar la reconstrucción de árboles, tomaron las distancias de

árbol de análisis sintáctico predichas para cada oración de prueba, calcularon el árbol de expansión mínima, utilizaron la puntuación de unión no dirigida (UUAS) y la correlación de Spearman entre las distancias verdaderas y las distancias predichas de las palabras en cada oración.

Según los resultados obtenidos en [7], se identificó una estructura sintáctica robusta incrustada en BERT. Esta estructura captura propiedades sintácticas fundamentales, como el orden de las palabras y las relaciones gramaticales. Se destacó un rendimiento óptimo en la capa 7-8 de BERT Base y en la capa 16 de BERT grande. Además, se observó que las palabras con funciones sintácticas similares tienden a agruparse de manera significativa en el espacio de representación.

En el trabajo de [11], también se encontró evidencia de representaciones geométricas detalladas de los sentidos de las palabras. Para esto, implementaron dos tipos de sondas: una del tipo binario para clasificar si existía una relación o no entre dos pares de palabras, y un segundo clasificador del tipo multiclasificación para determinar el tipo de relación entre palabras. La sonda binaria logró una precisión del 85,8%, y la sonda multiclasificación logró una precisión del 71,9%. Con este experimento, lograron determinar que la información sintáctica está codificada en los vectores de atención.

Para profundizar en este sentido, desarrollaron una herramienta de visualización que permite explorar representaciones geométricas de los sentidos de las palabras. Mediante un algoritmo de reducción dimensional, visualizaron cómo los sentidos de las palabras se encuentran separados espacialmente. Los resultados muestran que los sentidos de las palabras forman grupos claramente distinguibles. Por ejemplo, en la Figura 3.2, se muestra que la palabra en inglés “die” forma una agrupación cuando se utiliza como sustantivo (“dado”) y otra agrupación cuando se emplea como verbo (“morir”). Dentro de este último grupo, se pueden apreciar matices que permiten diferenciar el número asociado al verbo.

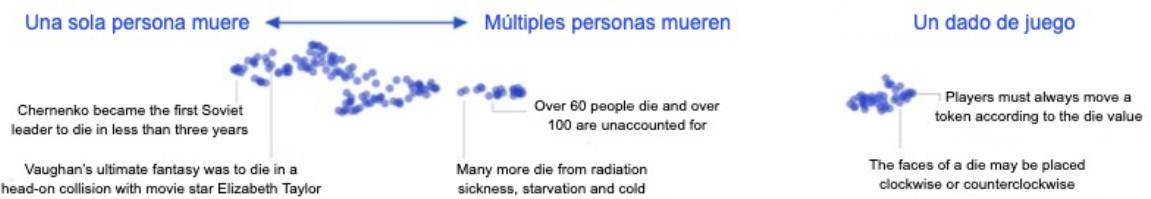


Figura 3.2: Incrustaciones en 2D de la palabra en inglés “*die*” en diferentes contextos. Figura modificada de 3.2.

Para probar la eficacia de las incrustaciones generadas por BERT en la desambiguación del sentido de las palabras, en [11] concatenaron oraciones que contenían una palabra con diferentes significados. Notaron que los tokens vecinos en la oración concatenada influían

Cuadro 3.1: Aspectos en los que se enfocan los cabezales de BERT base. Información recopilada de [9].

Enfoque	Número de cabezales
General en la oración	11
Token posterior	5 (capas 1,2,2,3 y 6)
Token previo	4 (capas 2,4,7 y 8)
CLS	1
SEP	19
Coreferencia	1
Objeto directo	1
Pronombres posesivos	1
Verbos auxiliares pasivos	1
Modificadores de sustantivos (determinantes)	1
Objeto de preposiciones	1
Puntuación (punto y coma)	18
Resto (sin clasificación)	80

en las incrustaciones del token de entrada, concluyendo que los tokens en el modelo BERT no respetan los límites semánticos, sino que absorben indiscriminadamente el significado de todos los tokens, lo que afecta la precisión en la interpretación del texto. De acuerdo con los experimentos, el trabajo de [11] demostró que la geometría interna de BERT puede dividirse en múltiples subespacios lineales, con espacios separados para la información sintáctica y semántica.

El trabajo de [9] se centra en analizar los mecanismos de atención en el modelo pre-entrenado de BERT. Analizan 144 cabezas de atención (modelo BERT base) y se dan cuenta de que hay cabezas especializadas que abstraen aspectos del lenguaje, como la sintaxis y la coreferencia. Algunas cabezas se enfocan en las relaciones de los verbos con sus objetos directos, los determinantes de los sustantivos, los objetos de las preposiciones y los objetos de los pronombres posesivos con una precisión superior al 75 %. En la Tabla 3.1, se desglosa la atención de los cabezales hacia algún aspecto lingüístico.

En [9], observaron que hasta un 90 % de la atención de BERT de un token cualquiera se enfoca en [SEP] (de la capa 5 a 9). Descubrieron que los gradientes de atención hacia el token separador se vuelven muy pequeños, lo que indica que aumentar o disminuir la atención en [SEP] no produce cambios significativos en las salidas de BERT. Por lo tanto, la atención hacia este token podría considerarse como una operación sin efecto en el rendimiento del modelo. Por otra parte, las capas 11 y 12 muestran una mayor atención en los signos de puntuación, mientras que en las primeras 4 capas observaron un mayor enfoque en el token de clasificación [CLS].

En [9], se emplearon sondas de atención para evaluar la capacidad sintáctica general de BERT. Estas sondas generan distribuciones de probabilidad sobre las palabras en una oración, indicando la probabilidad de que cada palabra sea la cabeza sintáctica de la palabra de entrada. Los experimentos concluyeron que BERT abstrae aspectos lingüísticos de su entrenamiento autosupervisado. Además, calcularon distancias entre pares de cabezas de atención usando la divergencia de Jensen-Shannon y las representaron en dos dimensiones mediante un escalado multidimensional. El análisis reveló que las distribuciones de atención en la misma capa son similares, y que las primeras capas exhiben alta entropía, disminuyendo gradualmente a lo largo de las capas.

Un estudio que empleó tanto un enfoque supervisado como uno no supervisado, combinando un clasificador y análisis de agrupamiento, se llevó a cabo en el trabajo de [44]. Este proyecto aborda el desafío de la polisemia en palabras. En el enfoque supervisado, introdujeron las incrustaciones de BERT en un clasificador lineal para predecir clases semánticas y evaluar la precisión del modelo. En cuanto al enfoque no supervisado, proyectaron las incrustaciones contextualizadas de BERT mediante PCA, seguido de un análisis de agrupamiento, con el objetivo de investigar en qué medida se forman regiones semánticas. Los resultados de sus experimentos revelaron que el subespacio de BERT no está exclusivamente determinado por la semántica, sino que también se entrelaza con conceptos como la sintaxis y el sentimiento.

Otro estudio que aborda el significado de las palabras es el de [45]. En este trabajo, emplean el clasificador K vecinos más cercanos para desambiguar palabras a partir de varias incrustaciones contextualizadas. Los resultados indican que BERT tiene la capacidad de identificar palabras polisémicas en regiones específicas de su espacio de incrustación, en contraste con otros tipos de incrustaciones como ELMo o Flair NLP.

Trabajos más recientes han optado por emplear clasificadores binarios con el propósito de mejorar la interpretabilidad de los modelos BERT. Un ejemplo de ello es el estudio llevado a cabo por [46], donde implementan un clasificador que define dos categorías: depresivo y no depresivo. Aunque la principal meta de este trabajo es la detección de la depresión mediante patrones lingüísticos específicos, indirectamente realiza un análisis de BERT al examinar dichos patrones. Sus conclusiones destacan la robustez del modelo frente a variaciones sensibles al género y al uso de pronombres. En una investigación adicional, [47] exploró el espacio vectorial de las representaciones de palabras del modelo BERT a través de grafos de conocimiento y clasificadores binarios para el reconocimiento de un concepto ontológico particular. Según sus hallazgos, las incrustaciones de BERT contienen información sobre la estructura de un grafo de conocimiento sin la necesidad de realizar ajustes finos.

# Capítulo 4

## Metodología

En este capítulo, se detallan los procedimientos, la arquitectura y las especificaciones utilizadas en el desarrollo de las pruebas experimentales. Se inicia describiendo los recursos utilizados a nivel de *hardware* y *software* para la realización de los experimentos. Posteriormente, se mencionan los conjuntos de datos utilizados para abordar la tarea de similitud semántica y el análisis de las autoatenciones.

En la sección 4.3, se proporcionan detalles sobre los experimentos realizados para resolver la tarea de similitud semántica mediante el token de clasificación CLS, así como los métodos de abstracción propuestos, que incluyen la red LSTM y los métodos de agregación promedio y máximo.

Por otro lado, en la sección 4.4, se detalla la metodología utilizada para abordar el problema de la interpretabilidad de las atenciones del modelo BERT a través del enfoque no supervisado.

Cabe señalar que el modelo utilizado para todos los experimentos es el modelo BERT base, que consta de 12 bloques codificadores con 12 cabezales de atención en cada capa.

### 4.1. Recursos de *hardware* y *software*

Los experimentos que no requirieron gran capacidad de cómputo se realizaron en una computadora con sistema operativo macOS Monterey versión 12.6.5 con las siguientes características:

- Procesador Intel Core i5 - 1.8 GHz de dos núcleos.
- Memoria RAM de 8 GB a 1600 MHz DDR3.

Para implementar los algoritmos de entrenamiento o aquellos experimentos en donde el poder de cómputo era mayor, se utilizó un ambiente que corre en la nube con las siguientes características:

- Acelerador de hardware GPU del tipo T4.
- 2 procesadores Intel(R) Xeon(R) CPU @ 2.20GHz.
- 12.7 GB de memoria RAM.

El lenguaje de programación utilizado fue Python en su versión 3.10.4, y se empleó la biblioteca de PyTorch con la versión 2.0.1 para la implementación de los algoritmos de aprendizaje automático.

## 4.2. Conjuntos de datos

### 4.2.1. STS-Benchmark

El conjunto de datos de similitud textual semántica STS-Benchmark es reconocido como un punto de referencia en la evaluación de la similitud semántica y ha sido utilizado en competencias anuales de evaluación semántica como SemEval.

Se utilizó el conjunto STS-Benchmark en su versión de idioma inglés. Este conjunto consiste de pares de oraciones, donde cada par ha sido anotado con una etiqueta que representa el grado de similitud semántica. La escala de similitud va de 0 a 5, donde 0 indica que no hay similitud semántica, 5 indica una equivalencia de significado y valores intermedios reflejan una superposición parcial de significado. La anotación humana de este corpus se llevó a cabo siguiendo las pautas establecidas por [48]. La tabla 4.1 proporciona una guía que describe el proceso de anotación realizado por humanos.

El conjunto de datos incluye conjuntos de entrenamiento, desarrollo/evaluación y prueba. La tabla 4.2 proporciona información sobre la cantidad de pares de oraciones en cada conjunto, así como el desglose de géneros de las oraciones. Estos datos se obtuvieron del recurso [49].

La escala de anotaciones está diseñada para ser accesible por jueces humanos sin experiencia formal en lingüística, a través de una colaboración colectiva en línea conocido como *crowdsourcing*. Los etiquetadores examinaron los pares de oraciones en inglés y asignaron etiquetas de similitud textual semántica a cada par. La etiqueta de oro se definió tomando cinco anotaciones (por par) y se calculó el promedio de los cinco puntajes. Las etiquetas se transfirieron a pares de conjuntos de datos multilingües y cruzados; sin embargo, las correlaciones más sólidas se lograron en el idioma inglés.

Cuadro 4.1: Criterios de anotación humana para el corpus STS-Benchmark. Información tomada de [48].

5	Las dos sentencias son completamente equivalentes, ya que significan lo mismo.
	<i>The bird is bathing in the sink.</i> <i>Birdie is washing itself in the water basin.</i>
4	Las dos sentencias son en su mayoría equivalentes, pero algunos detalles sin importancia difieren.
	<i>Two boys on a couch are playing video games.</i> <i>Two boys are playing a video game.</i>
3	Las dos sentencias son más o menos equivalentes pero falta o difiere alguna información importante.
	<i>John said he is considered a witness but not a suspect.</i> <i>“He is not a suspect anymore.” John said.</i>
2	Las dos sentencias no son equivalentes pero comparten algunos detalles.
	<i>They flew out of the nest in groups.</i> <i>They flew into the nest together.</i>
1	Las dos oraciones no son equivalentes pero están sobre el mismo tema.
	<i>The woman is playing the violin.</i> <i>The young lady enjoys listening to the guitar.</i>
0	Las dos oraciones son completamente diferentes.
	<i>The black dog is running through the snow.</i> <i>A race car driver is driving his car through the mud.</i>

Cuadro 4.2: Conjunto de datos STS-Benchmark

	Entrenamiento	Evaluación	Prueba	Total
Noticias	3299	500	500	4299
Títulos	2000	625	525	3250
Foro	450	375	254	1079
Total	5749	1500	1379	8628

### 4.2.2. SICK-R

De acuerdo con [50], el conjunto de datos SICK significa *Sentences Involving Compositional Knowledge* y contiene pares de oraciones en idioma inglés que incluyen fenómenos léxicos, sintácticos y semánticos. Este conjunto de datos es útil en dos tareas de procesamiento de lenguaje natural:

- Relacionado (SICK-R): predice el grado de similitud semántica en pares de oraciones. Contiene una etiqueta de valor continuo dentro de un rango de 1 a 5. Donde 1 representa similitud semántica nula y 5 es similitud semántica perfecta.
- Inferencia: detecta la relación de implicación entre las 2 oraciones. Contiene 3 etiquetas que indican inferencia, contradicción y neutro.

La anotación del conjunto de datos SICK se llevó a cabo a través de un extenso estudio de *crowdsourcing*, donde cada par de oraciones fue evaluado por 10 personas; posteriormente, la puntuación de similitud dorada se obtuvo promediando las 10 calificaciones asignadas por los participantes, según [50].

El conjunto SICK está compuesto por 9840 pares de oraciones; en la tabla 4.3 se puede ver su desglose.

Cuadro 4.3: Conjunto de datos SICK

Entrenamiento	Evaluación	Prueba	Total
4439	495	4906	9840

## 4.3. Similitud semántica a través del token CLS y otros métodos de abstracción.

Para abordar la tarea de similitud semántica en pares de oraciones en inglés, se han propuesto enfoques que implican abstracciones sobre la salida del modelo BERT base. Estos enfoques incluyen el uso de una red recurrente LSTM bidireccional y técnicas de agregación, como el *mean-pooling* y el *max-pooling*. Este intento de superar la abstracción del token de clasificación CLS es crucial, ya que este componente generalmente se utiliza para tareas de clasificación o regresión.

La Figura 4.1 proporciona una visión general de la metodología empleada para resolver la tarea de similitud semántica.

A continuación, se proporciona un detalle más exhaustivo de la metodología.

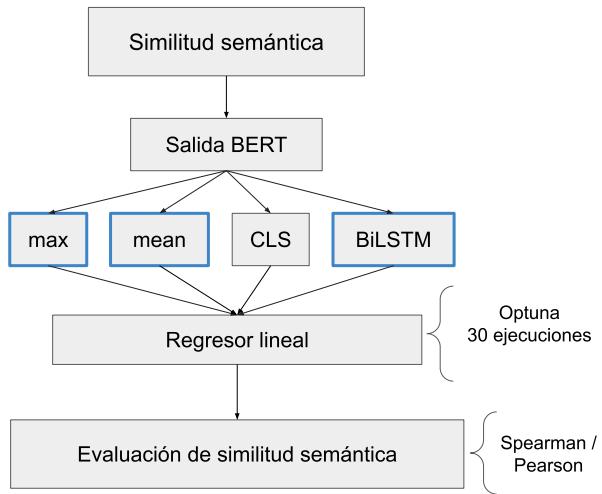


Figura 4.1: Metodología para resolver el problema de similitud semántica mediante métodos propuestos (mean, max y BiLSTM) y el token CLS.

#### 4.3.1. Preprocesamiento del corpus

Durante el preprocesamiento del conjunto de datos, se utilizó la función ‘tokenizer’ y el método ‘*encode\_plus*’ de la biblioteca *Hugging Face’s Transformers* en PyTorch para aplicar la tokenización especial requerida por BERT, conocida como “*WordPiece*”. Esto implicó asignar identificadores numéricos a cada token según el vocabulario, agregar tokens especiales como [CLS] y [SEP], y llenar las secuencias para que tuvieran la misma longitud (128 en este caso).

El método ‘*encode\_plus*’ generó los vectores que contenían los identificadores numéricos asignados a cada token, las máscaras de atención y el vector que identifica a la sentencia 1 y la sentencia 2. Estos vectores se convirtieron en tensores para su manejo eficiente en PyTorch. Para cargar los datos en lotes y acceder a ellos de manera eficiente, se utilizó la utilidad ‘*DataLoader*’ de PyTorch. Se crearon dataloaders separados para los conjuntos de entrenamiento, validación y prueba. El tamaño de lote se estableció en 32 secuencias para estos experimentos.

#### 4.3.2. Entrenamiento

La salida del modelo BERT tiene la forma [tamaño\_lote, long\_sentencia, dim\_incrustación]. Los métodos de agregación (operaciones de promedio o máximo) se aplicaron a lo largo de la longitud de la sentencia, generando vectores fijos con la forma [tamaño\_lote, dim\_incrustación].

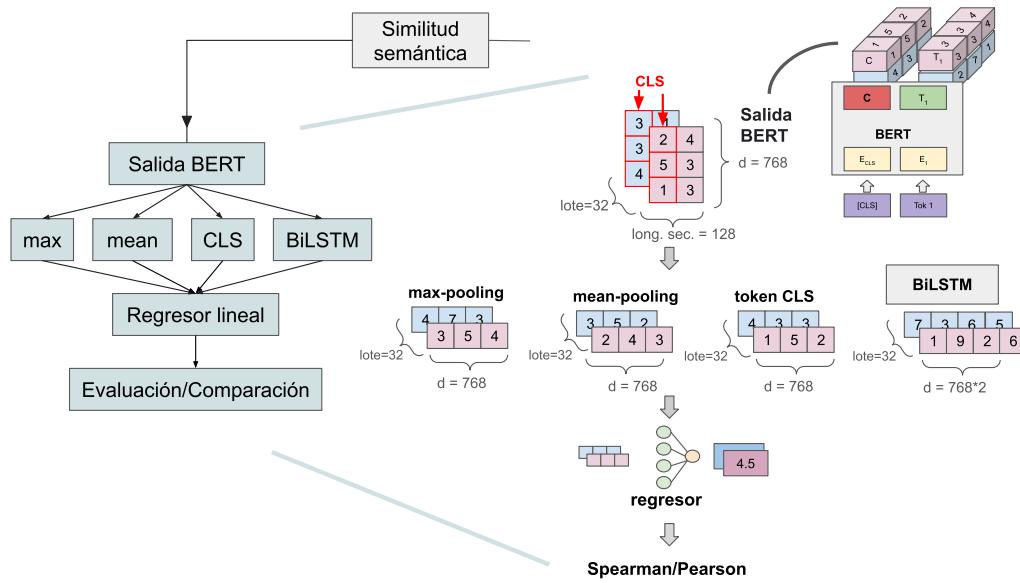


Figura 4.2: Detalle de los métodos de abstracción implementados para abordar la tarea de similitud semántica.

A continuación, estos vectores se introdujeron en un modelo regresor para evaluar la similitud semántica. En el caso de la red BiLSTM, la salida del modelo BERT se utilizó como entrada para la red BiLSTM, y la salida de esta fue el último estado oculto final. Dado que es una red bidireccional, la salida del modelo tuvo la forma [tamaño\_lote, dim\_incrustación \* 2]. Al igual que en los métodos de agregación, esta salida se dirigió directamente al modelo regresor. En cuanto al token CLS, se seleccionó la representación de la primera posición en la longitud de secuencia de la salida de BERT, la cual se introdujo directamente en el regresor. La descripción de cada método se ilustra en la figura 4.2.

#### 4.3.3. Validación y optimización de hiperparámetros

Para llevar a cabo la validación y optimización de hiperparámetros, se empleó el *framework* Optuna. En el caso del modelo BERT base, se realizaron aproximadamente 30 pruebas para cada uno de los métodos propuestos.

En el proceso de optimización, se seleccionó el valor de la pérdida o error del conjunto de validación y entrenamiento como el objetivo a minimizar. Para gestionar y coordinar el proceso de optimización de hiperparámetros, se creó un estudio individual para cada uno de los métodos propuestos.

Los parámetros optimizados por el *framework* Optuna incluyeron la tasa de aprendizaje (desde 1e-5 hasta 1e-4), el optimizador (Adam, RMSprop y SGD), el número de

capas densas (desde 1 hasta 3 capas), y el dropout para la red LSTM y las capas densas (desde 0 hasta 0.8).

En la tabla 4.4, se presenta la configuración determinada para cada uno de los métodos propuestos, para ambos conjuntos de datos y mediante el *framework* de Optuna. Los cuatro métodos propuestos comparten una arquitectura de ajuste fino similar, diferenciándose en los valores asignados a los parámetros de tasa de aprendizaje, dropout y cantidad de capas densas en el regresor.

Cuadro 4.4: Hiperparámetros de los métodos propuestos

Parámetro	CLS	MEAN	MAX	LSTM
Tamaño de lote			32	
Long. sentencia			128	
Épocas			5	
Función pérdida	Error Cuadrático Medio (MSELoss)			
Optimizador	RMSprop ( <i>Root Mean Square Propagation</i> )			
<b>STS-Benchmark</b>				
Tasa aprendizaje	2.45e-5	3.1e-5	2.8e-5	3.63e-5
Dropout	0.0681	0.06	0.04	0.8
Dropout(LSTM)	No aplica			0.6
Capas densas	2			
<b>SICK-R</b>				
Tasa aprendizaje	2.89e-5	2.85e-5	3.7e-5	1.94e-5
Dropout	0.132	0.48	0.464	0.32
Dropout(LSTM)	No aplica			0.51
Capas densas	3	2	2	3

#### 4.3.4. Medición de desempeño

Para realizar la medición de desempeño del regresor lineal al abordar la tarea de similitud semántica, se utilizaron las siguientes métricas:

- **Coeficiente de correlación de Pearson:** Describe una relación lineal entre la variable independiente (predictora  $\hat{y}$ ) y la variable dependiente (respuesta  $y$ ). Puede

tener valores entre -1 y +1, donde el signo indica el tipo de relación lineal (negativa o positiva). Mientras más cerca esté de -1 o +1, mayor es la fuerza en la relación, mientras que un valor cercano a cero indicaría que no hay una correlación lineal ([51]). La fórmula para calcular el coeficiente de correlación de Pearson es la siguiente:

$$\rho_p = \frac{\sum_{i=1}^N \hat{y}_i y_i}{\sqrt{\sum_{i=1}^N \hat{y}_i^2 \sum_{i=1}^N y_i^2}} \quad (4.1)$$

- **Coeficiente de correlación de Spearman:** A diferencia del coeficiente de correlación de Pearson, la correlación de Spearman evalúa la correlación monotónica u ordenada, lo que significa que puede detectar relaciones no lineales que siguen un patrón específico. El coeficiente de correlación de Spearman se puede calcular utilizando la fórmula del coeficiente de correlación de Pearson 4.1, pero primero se deben convertir los datos originales en rangos (ordinales), con valores entre 1 y  $N$  ([52]). La fórmula para calcular el coeficiente de correlación de Spearman se muestra en 4.2.

$$\rho_s = \frac{\sum_{i=1}^N \hat{y}_{i,r} y_{i,r}}{\sqrt{\sum_{i=1}^N \hat{y}_{i,r}^2 \sum_{i=1}^N y_{i,r}^2}} \quad (4.2)$$

Es importante señalar que tanto  $y$  y  $\hat{y}$  de las fórmulas definidas en 4.1 y 4.2 están centrados alrededor de la media.

- **Error cuadrático medio:** Métrica conocida en inglés como *Mean Squared Error* (MSE), es comúnmente utilizada para evaluar la calidad de un modelo de regresión. Se utiliza para medir la diferencia cuadrada promedio entre los valores predichos por el modelo y los valores reales. Cuanto menor sea el valor del MSE, mejor será el rendimiento del modelo, [28]. La fórmula del MSE se expresa de la siguiente manera:

$$MSE = \frac{1}{n} \sum_{i=1}^n (Y_i - \hat{Y}_i)^2 \quad (4.3)$$

En 4.3, el valor  $n$  representa el número de observaciones del conjunto de datos,  $Y_i$  son los valores reales para la observación  $i$  y  $\hat{Y}_i$  representan los valores predichos por el modelo para la observación  $i$ .

Estas métricas se utilizaron para evaluar el conjunto de entrenamiento, de validación y prueba del corpus STS Benchmark y del conjunto SICK-R.

## 4.4. Análisis de autoatenciones

Para llevar a cabo el análisis de las autoatenciones del modelo BERT base en cada una de sus capas, se implementó un autoencoder que facilitara la reducción dimensional de las atenciones a un vector de tamaño fijo. Después de obtener la representación, se realizó un análisis exploratorio utilizando algoritmos de agrupamiento. Se implementaron métricas intrínsecas para evaluar la calidad de los agrupamientos. Este análisis abordó tres aspectos del lenguaje: longitud de sentencia (LS), similitud semántica (SS) y similitud de estructura gramatical (SEG). En la Figura 4.3 se presentan las secciones principales para llevar a cabo el análisis de las autoatenciones, las cuales se detallarán a lo largo de esta sección.

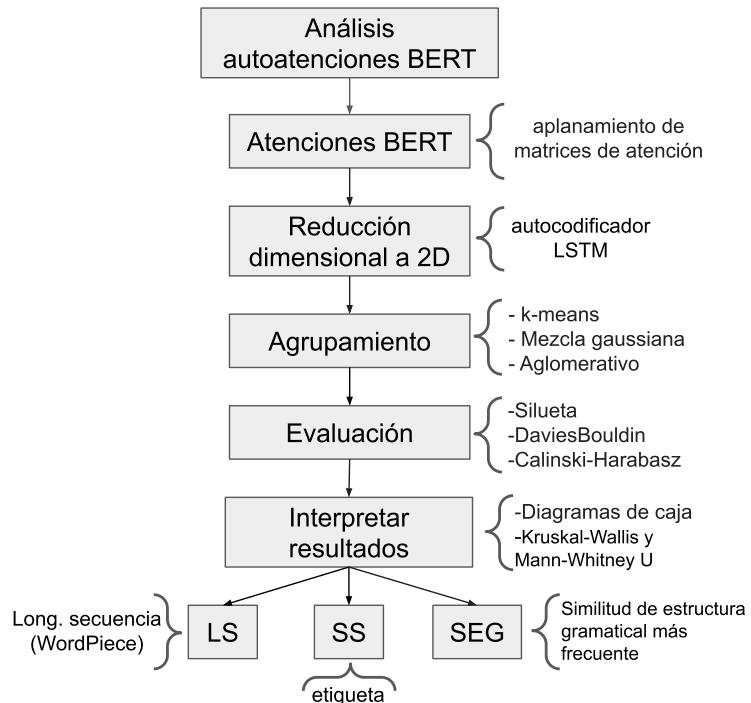


Figura 4.3: Metodología para analizar las autoatenciones del modelo BERT.

Cabe señalar que se implementó una herramienta visual que permitió un mayor entendimiento del comportamiento de los datos, así como automatizar y facilitar el análisis exploratorio, que incluye la fase de agrupamiento, evaluación y la interpretación de los resultados para cada aspecto lingüístico.

#### 4.4.1. Manejo de las representaciones de atención

La biblioteca de *Hugging Face* permite extraer y tener acceso a las representaciones del modelo transformador a nivel de capa. Para este apartado, se extrajeron las atenciones de cada cabezal de todas las capas del modelo BERT base, para los conjuntos de prueba STS-Benchmark y SICK-R.

Para analizar las atenciones y extraer información relevante, fue necesario realizar un recorte del relleno de las atenciones. Se obtuvieron las matrices cuadradas de atención con tamaños diversos, dependiendo de la longitud de la tokenización “*WordPiece*” realizada para BERT.

Las atenciones se abstrajeron para cada capa, lo que significa que para cada ejemplo de los conjuntos de prueba, se guardaron 144 matrices de atención (12 capas x 12 cabezales), lo que permitió un análisis detallado de la información de atención en diferentes capas del modelo.

Los tensores de las atenciones de cada cabezal se apilaron a lo largo de una sola dimensión. Es decir, que la representación de una capa consistía del aplanamiento de las atenciones de sus 12 cabezales. De la figura 4.4, el primer subíndice de la matriz de atención  $A$  denota la capa  $l$ , el segundo subíndice representa el número de cabezal  $h$ , y el tercer subíndice se refiere a la posición específica de un peso de atención  $(i, j)$ , es decir,  $A_{l,h,(i,j)}$ , y sea  $n$  el número de tokens que conforman una secuencia.

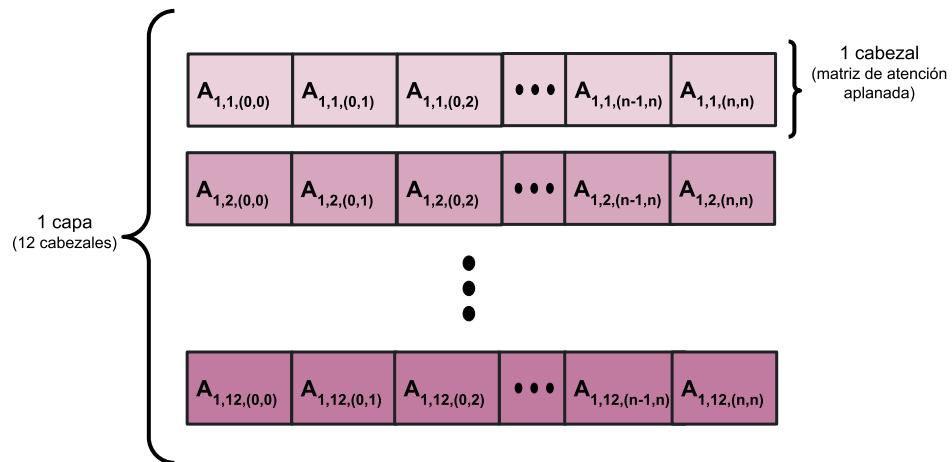


Figura 4.4: Atenciones de cabezales apilados para cada capa del modelo BERT. Cada capa contiene los 12 cabezales con matrices de atención aplanadas.

#### 4.4.2. Reducción de dimensionalidad mediante autocodificador

Se hizo uso del aprendizaje no supervisado al implementar un autocodificador o *autoencoder* basado en redes recurrentes, específicamente una red LSTM, con el propósito de llevar a cabo una reducción de dimensionalidad. Esta reducción se configuró para producir una representación de 2 dimensiones, con el objetivo de facilitar el análisis y la visualización de las abstracciones generadas por el modelo.

La implementación del autocodificador involucró la definición de su entrada de datos, así como la creación de los bloques codificador y decodificador. A continuación, se proporciona una descripción más detallada de cada una de estas secciones.

##### Entrada de datos del autocodificador

Para cada ejemplo de los conjuntos de prueba de STS-Benchmark y SICK-R, se definieron 12 representaciones, una para cada capa específica del modelo. Estas representaciones se diseñaron de manera que el autocodificador pudiera trabajar con lotes de datos. Dado que las secuencias de texto tenían longitudes variables, sus representaciones de atención también varíanan en tamaño. Se optó por establecer un tamaño de lote de 12 con el objetivo de que este tamaño reflejara las 12 capas del modelo BERT base. Esto aseguraba que todos los lotes tuvieran el mismo tamaño, lo que mejoraba la eficiencia del modelo en PyTorch. Como resultado, se generaron un total de 1379 lotes para el conjunto STS-Benchmark y un total de 4906 lotes para el conjunto SICK-R.

Una vez aplanadas las autoatenciones para cada capa, se configuró el tamaño del lote en 12, que corresponde al número de capas en el modelo BERT base. La dimensión de la secuencia se definió conforme a la longitud de la atención aplanada de cada cabezal, mientras que el número de características que se esperan en cada paso de tiempo de la red LSTM es igual al número de cabezales presentes en una capa del modelo BERT base, que es 12. En la figura 4.5 se ilustra la estructura de los datos de entrada.

##### Arquitectura y entrenamiento del autocodificador

El diseño del autocodificador se compone de dos bloques: un codificador (*encoder*) y un decodificador (*decoder*), ambos basados en redes neuronales recurrentes conocidas como “*Long Short-Term Memory*” o LSTM.

El bloque del codificador, denotado como  $e_\theta(x)$ , toma como entrada los datos previamente definidos y genera una representación latente  $z$  de menor dimensión, comúnmente referida como “estado oculto” o “*hidden state*”. Se optó por una dimensión de dos unidades (2 dimensiones) para el estado oculto, con el propósito de facilitar la visualización de las representaciones al representarlas gráficamente.

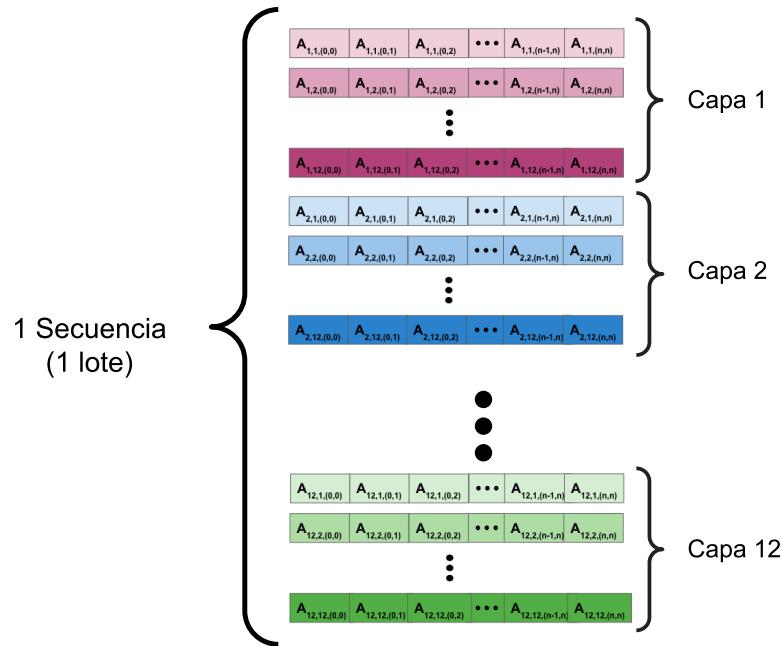


Figura 4.5: Estructura de las representaciones formadas para la entrada al modelo autocodificador.

Por otro lado, el bloque del decodificador, representado como  $d_\varphi(z)$ , recibe la representación latente como entrada y la utiliza para reconstruir los datos originales. La salida del decodificador se compara con la entrada original del codificador para calcular el error del modelo y guiar el proceso de entrenamiento. En la figura 4.6, se proporciona una representación visual de alto nivel del modelo.

Es importante destacar que tanto la red LSTM en el bloque codificador como en el bloque decodificador mantienen la misma configuración. En este caso, se utiliza la forma más simple de una red LSTM, que consta de una sola capa y procesa la secuencia en una sola dirección, es decir, hacia adelante en el tiempo, sin emplear bidireccionalidad. En la tabla 4.5 se presenta un resumen de los parámetros principales que definen la arquitectura del modelo para ambos conjuntos de datos.

#### 4.4.3. Algoritmos de agrupamiento y evaluación

Para explorar patrones semánticos en las representaciones de BERT a través de los vectores bidimensionales generados por el modelo autocodificador, se graficaron mediante diagramas de dispersión los 1379 vectores del conjunto de prueba STS-Benchmark y los 4906 vectores del conjunto de prueba SICK-R para cada capa del modelo BERT.

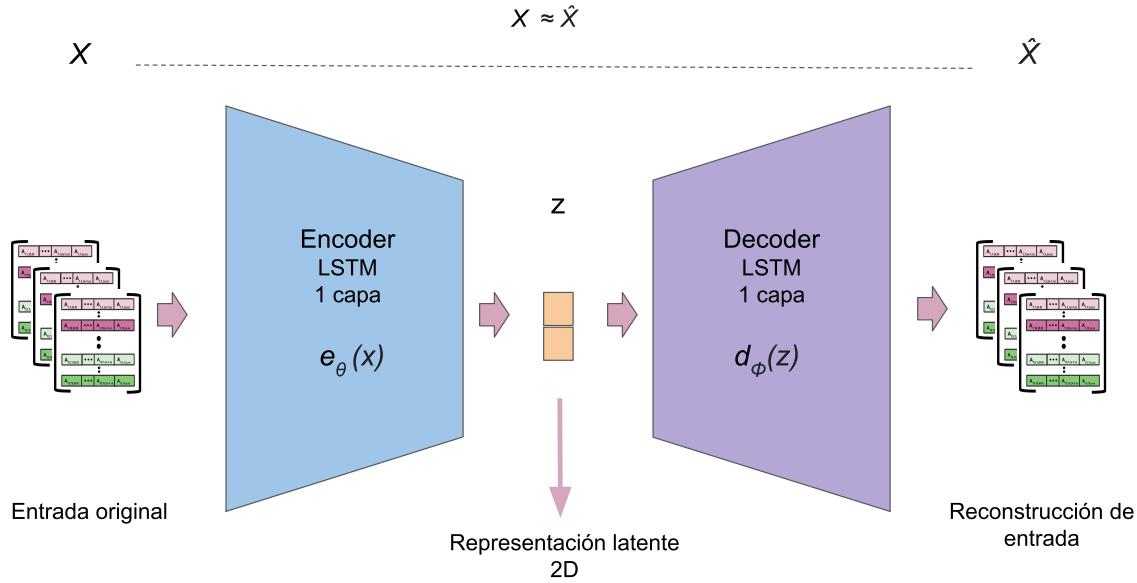


Figura 4.6: Autocodificador propuesto basado en redes LSTM para reducción de dimensionalidad.

Para cada capa, se realizó un análisis exploratorio mediante diferentes algoritmos de agrupamiento, como k-means, DBSCAN, jerárquico aglomerativo, espectral y mezcla gaussiana. Mediante la biblioteca de aprendizaje automático de scikit-learn y SciPy, fue posible desarrollar una herramienta visual que facilitó la configuración de los respectivos hiperparámetros de cada algoritmo. Para evaluar la calidad de los agrupamientos, se implementaron algunas métricas intrínsecas, como el índice de silueta, el índice de Davies-Bouldin y el índice de Calinski Harabasz. De manera particular, para el algoritmo de k-means se implementó el cálculo de la inercia y para el algoritmo de mezcla gaussiana se implementó el Criterio de Información Bayesiano (BIC) y el Criterio de Información de Akaike (AIC).

Determinar el número óptimo de agrupamientos representó un desafío significativo, por lo que en la herramienta visual se implementaron el método del codo y la representación gráfica de todas las métricas de evaluación para un número de agrupamientos  $k$ , definido por el usuario. Las gráficas para cada métrica indicaban el valor óptimo a lo largo del barrido de  $k$ . La decisión sobre cuántos agrupamientos considerar para cada capa se basó en dos criterios. El primero fue el consenso y la votación de todas las métricas, mientras que el segundo dependió de la exploración de los datos, definiendo el valor de  $k$  que resultara más útil para la interpretabilidad del análisis.

Cuadro 4.5: Hiperparámetros del modelo autocodificador.

Parámetro	Valor
Tasa aprendizaje	1e-3
Tamaño de lote	12
Tamaño de entrada	12
Estado oculto	2
Épocas	4
Función pérdida	Error cuadrático medio (MSELoss)
Optimizador	Estimación adaptativa de momentos (Adam)

#### 4.4.4. Interpretación de aspectos lingüísticos

Para interpretar los resultados, se emplearon herramientas de análisis, como diagramas de caja, junto con la aplicación de pruebas estadísticas, como la prueba de Mann-Whitney U para analizar diferencias significativas entre dos grupos y la prueba de Kruskal-Wallis cuando se contaba con más de dos agrupamientos. La elección de estas pruebas se basó en un análisis previo de normalidad de los datos mediante Shapiro-Wilk y una evaluación de la igualdad de varianzas mediante Levene. Estos análisis revelaron desigualdad de varianzas y una distribución no normal de los datos (datos no paramétricos). La implementación de estas pruebas estadísticas tenía como objetivo calcular la estadística y el valor p, sin definir un umbral de significancia específico; simplemente se buscaba evaluar y analizar los valores p.

En cuanto a la formulación de la hipótesis nula y alternativa, estas podrían definirse de la siguiente manera:

- Hipótesis nula  $H_0$ : No hay diferencias significativas entre los grupos identificados por los algoritmos de agrupamiento.
- Hipótesis alternativa  $H_1$ : Hay diferencias significativas entre al menos uno de los grupo identificados por el algoritmo de agrupamiento.

La interpretación de los resultados, así como la visualización mediante diagramas de caja y el cálculo de estadísticas y valores  $p$  para las pruebas de Kruskal-Wallis y Mann-Whitney U, se vieron facilitados gracias a la herramienta visual que se desarrolló.

La interpretación de los resultados se definió para 3 aspectos del lenguaje: longitud de secuencia (LS), similitud semántica (SS) y similitud de estructura gramatical (SEG). A continuación se detalla el análisis para cada aspecto del lenguaje.

### Análisis de longitud de secuencia

Se extrajeron las dimensiones de cada secuencia de los conjuntos de prueba de STS-Benchmark y SICK-R en función de la longitud de secuencia definida por la tokenización *WordPiece* utilizada por BERT. La distribución de la longitud de secuencia para el conjunto STS-Benchmark se muestra en la figura 4.7(a) y para el conjunto SICK-R en la figura 4.7(b).

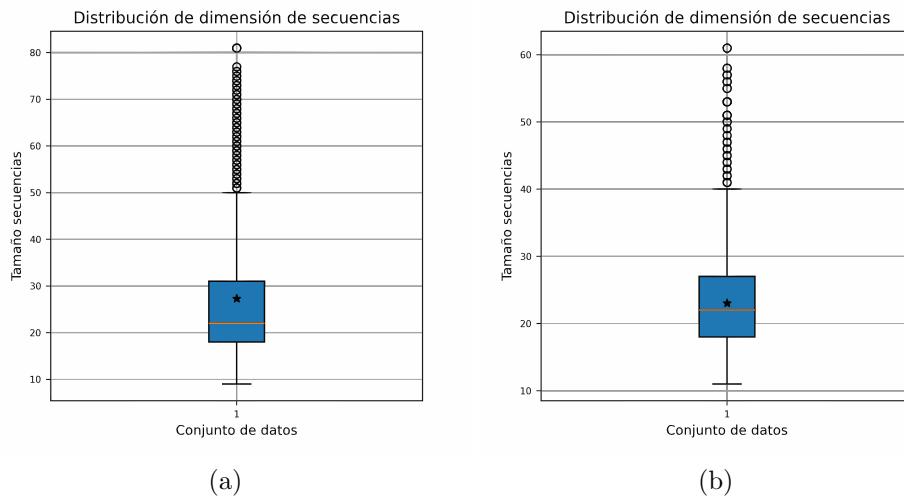


Figura 4.7: Distribución de los datos con respecto a la longitud de secuencia, en el conjunto de prueba STS-Benchmark (a) y en el conjunto de prueba SICK-R (b).

Para ambos conjuntos de datos se muestra una asimetría positiva, que se atribuye a la presencia de secuencias con dimensiones notablemente altas. También se muestra que el conjunto de datos STS-Benchmark muestra longitudes más grandes.

Por otro lado, la herramienta visual diseñada para este aspecto del lenguaje posibilita la introducción de rangos de longitud de secuencia, permitiendo además la visualización de las muestras que concuerdan con el rango especificado por el usuario.

### Análisis de similitud semántica

En este aspecto, se analizaron los patrones generados por el modelo en relación con la similitud semántica de las muestras, es decir, conforme a las etiquetas de similitud proporcionadas por los conjuntos de datos. La distribución de las etiquetas de similitud en los conjuntos de datos se presenta en el diagrama de la figura 4.8(a) para el conjunto STS-Benchmark y en 4.8(b) para el conjunto SICK-R.

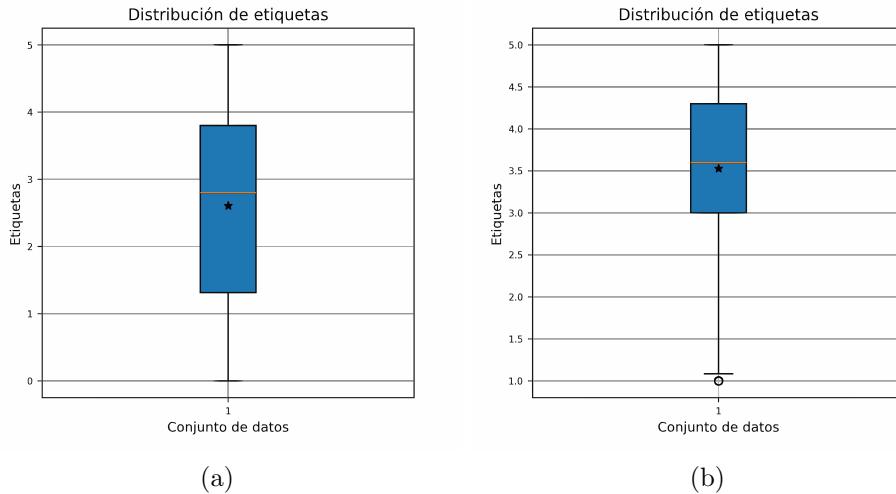


Figura 4.8: Distribución de las etiquetas de similitud semántica en el conjunto de prueba STS-Benchmark (a) y el conjunto de prueba SICK-R (b).

Las distribuciones de las etiquetas del conjunto STS-Benchmark no presentan valores atípicos, pero exhiben un ligero sesgo hacia la izquierda, lo que indica que hay más etiquetas con valores de similitud semántica más altos. En cuanto al conjunto de datos SICK-R, se observa una mayor dispersión de los datos para valores bajos en similitud semántica.

La herramienta visual implementada complementa el análisis exploratorio de los datos al facilitar la introducción de rangos de similitud semántica y su visualización, lo que contribuyó a una mejor comprensión del comportamiento de los datos.

### Análisis de similitud de estructura gramatical

En esta sección, se procedió a identificar las estructuras gramaticales presentes en todas las secuencias de los conjuntos de prueba STS-Benchmark y SICK-R. Es importante destacar que la estructura de la secuencia es, en realidad, la combinación de las estructuras gramaticales de los dos pares de oraciones.

Se determinó la estructura gramatical más frecuente en cada conjunto de datos y, posteriormente, se calculó la similitud coseno entre esta y el resto de las estructuras. Los puntajes resultantes de la similitud estructural se emplearon como variables de análisis.

La estructura gramatical más recurrente en el conjunto STS-Benchmark se encuentra en 32 secuencias del conjunto de prueba, mientras que la estructura gramatical más frecuente en el conjunto SICK-R se repite 82 veces. Es relevante señalar que la estructura más común en el conjunto de datos STS-Benchmark es idéntica a la del conjunto SICK-R.

Las dos oraciones que forman la secuencia más popular comparten las mismas etiquetas de dependencia, que incluyen:

```
[‘dep’, ‘det’, ‘nsubj’, ‘aux’, ‘ROOT’, ‘det’, ‘dobj’, ‘punct’]
```

La herramienta visual desarrollada simplificó el análisis de la similitud de estructura gramatical. Facilita la identificación de las estructuras más recurrentes en el conjunto de datos y proporciona visualización de aquellas estructuras que son más similares entre sí, ajustando un umbral de similitud según las preferencias del usuario.

# Capítulo 5

## Resultados experimentales

### 5.1. Similitud semántica mediante el token CLS y métodos propuestos

Ninguno de los métodos propuestos (agregación promedio, agregación máxima y red LSTM bidireccional) logró superar la capacidad de abstracción de información de lenguaje que el token de clasificación CLS realiza sobre toda la secuencia al evaluar la similitud semántica en los conjuntos de datos STS-Benchmark y SICK-R. En la tabla 5.1 se presentan los resultados obtenidos para ambos conjuntos de datos a través de las métricas de coeficiente de correlación de Spearman y Pearson.

En la tabla 5.1, se presentan los resultados obtenidos por los métodos propuestos en comparación con el estado del arte, representado por los trabajos de [39] y [40]. Estos estudios calcularon la similitud semántica utilizando el token de clasificación CLS mediante un ajuste fino sencillo a través de un regresor lineal. Aunque implementaron métodos más elaborados o técnicas adicionales, no se reportan aquí, ya que el objetivo es proporcionar únicamente un valor de referencia para comparar y analizar los resultados de la abstracción del token CLS con respecto a los métodos propuestos en este trabajo.

La abstracción del token CLS es ligeramente mejor que la generada por los métodos propuestos. Sin embargo, surge la curiosidad de analizar estos resultados.

#### 5.1.1. Análisis y discusión sobre métodos propuestos y token CLS

Dado que el modelo BERT comparte la misma arquitectura del bloque codificador que el modelo transformador propuesto por [4], este trabajo trata de comprender cómo los

Cuadro 5.1: Resultados de similitud semántica mediante token CLS, métodos de agregación y red BiLSTM

Metodo	Spearman	Pearson
<b>CLS</b> (STS-B) [39]	84.30 $\pm$ 0.76	-
<b>CLS</b> (SICK-R) [40]	82.9	88.6
<b>STS-Benchmark</b>		
<b>CLS</b>	<b>84.8</b>	<b>86.1</b>
MAX	84.2	84.2
MEAN	84.4	85.7
LSTM	84	85.4
<b>SICK-R</b>		
<b>CLS</b>	<b>83.2</b>	<b>88.6</b>
MAX	82.6	82.6
MEAN	82.5	88
LSTM	82.8	<b>88.6</b>

mecanismos de autoatención y la subcapa *feed forward position-wise*, que forman parte de este bloque codificador, influyen en la abstracción de información del token de clasificación CLS.

Cuando transitamos por el mecanismo de autoatención, consideremos la primera capa y un solo cabezal. En esta etapa, se realiza el producto punto de los vectores  $Q$  y  $K^T$ , proporcionando así una medida de relevancia y relación entre las consultas y las claves. En otras palabras, esta operación calcula la relación entre dos pares de palabras. El resultado de la autoatención (considerando que aún no se multiplica por  $V$ ) es una matriz cuadrada en la que la primera fila y la primera columna corresponden al token CLS. Esto significa que obtenemos una medida de la relación entre el token CLS y las demás palabras de la secuencia, mientras que el resto de la matriz contiene las relaciones entre las palabras restantes.

La forma en que se gestiona la atención nos proporciona una medida de relación entre pares de palabras que sugiere la aparición de la sintaxis. Sin embargo, la representación del token CLS en el mecanismo de atención mantiene un enfoque individual, es decir, establece relaciones entre el token CLS y las palabras de la secuencia, pero hasta ese momento, la abstracción del token CLS no considera las relaciones semánticas que podrían existir entre otros pares de palabras en la secuencia. A pesar de esto, sí puede obtenerse una perspectiva

global sobre las relaciones gramaticales en la secuencia, lo cual puede ser muy beneficioso en el contexto de tareas relacionadas con la sintaxis.

El modelo BERT es capaz de capturar y aprender relaciones cada vez más complejas a medida que se entrena a lo largo de las capas codificadoras. A medida que la información relevante se abstrae y se propaga gradualmente a través de las capas superiores, el modelo tiene la capacidad de capturar relaciones más complejas entre las palabras.

A partir de la interpretación anterior, el resultado de los métodos de agregación aplicados a la representación de la última capa, en comparación con el token CLS, se debe a que las relaciones entre las palabras en el resto de la secuencia tienen diferentes niveles de relevancia. Por ejemplo, al realizar una operación de *mean-pooling* en la representación generada a la salida del último codificador, se considera igualmente relevante cada par de palabras de la secuencia, asumiendo que todas las relaciones tienen la misma importancia. Esto no necesariamente es correcto, especialmente considerando la presencia de palabras vacías o *stopwords* que pueden establecer relaciones poco relevantes en el análisis semántico.

En el caso del método *max-pooling*, que solo considera el valor máximo en la representación, se pierde una parte significativa de la información contextual, la cual es crucial en tareas de similitud semántica.

La red recurrente bidireccional LSTM tampoco mostró un rendimiento superior al token CLS. Esto podría deberse a que el modelado de lenguaje realizado por BERT para capturar la estructura y relaciones entre palabras es efectivo y ha capturado información de alta calidad, limitando la capacidad de la red recurrente LSTM para extraer información adicional. También es posible que la red recurrente tenga dificultades para capturar relaciones a largo plazo en la secuencia, lo cual podría tener sentido, ya que los peores resultados registrados en el método LSTM fueron para el conjunto de datos STS-Benchmark, el cual contiene secuencias de dimensionalidad más larga que el conjunto SICK-R.

## 5.2. Interpretabilidad lingüística en autoatenciones

Los resultados de interpretabilidad de las autoatenciones revelan que el aspecto de longitud de secuencia muestra su mejor desempeño en capas inferiores o tempranas, aunque puede extenderse hasta capas intermedias con una significativa degradación. En cuanto a la similitud semántica, se observa una consistencia destacada en ambas capas de datos, especialmente en las capas 6, 8, 9 y 10. Por otro lado, el aspecto de similitud de estructura gramatical proporciona resultados más variables y menos consistentes en comparación con los otros dos aspectos; no obstante, se aprecia una tendencia hacia el agrupamiento de

estructuras similares.

En general, en la mitad inferior de las capas, se observa una mayor dispersión en las autoatenciones, mientras que en la mitad superior de las capas se evidencia una menor dispersión. Estos resultados se comparan con los hallazgos de los trabajos del estado del arte y se resumen de manera concisa en la tabla 5.2.

Cuadro 5.2: Resumen de resultados de interpretabilidad lingüística

Aspecto	Estado del arte	Resultado
Superficie	Inferiores [10], [6]	Inferiores(max) - intermedias
Semántica	Intermedias (6-9) [6], Superiores [10],[6]	Capas 6, 8, 9 y 10
Sintaxis	Intermedias [10],[6] Inferiores [8]	En la mayoría de las capas se agrupan estructuras similares
Entropía	Alta en capas inferiores [9]	Alta en capas inferiores

En las subsecciones siguientes se proporcionan detalles específicos sobre los resultados obtenidos para cada aspecto del lenguaje analizado.

### 5.2.1. Resultados de reducción dimensional y agrupación

La reducción dimensional realizada por el autocodificador LSTM generó una pérdida de entrenamiento de 0.045 en el conjunto STS-Benchmark y de 0.05 en el conjunto SICK-R. En el contexto de una tarea textual, estos valores de pérdida de entrenamiento podrían considerarse bajos. Esto sugiere que el modelo ha logrado aprender una representación latente de baja dimensionalidad de manera efectiva, especialmente dado que la reducción a solo 2 dimensiones implica una dimensionalidad bastante pequeña.

Una vez obtenidos los vectores bidimensionales para ambos conjuntos de datos, se procedió a llevar a cabo el análisis de agrupamiento. En la tabla 5.3 se presenta un desglose de los algoritmos empleados para cada capa, junto con los hiperparámetros específicos utilizados para el conjunto de datos STS-Benchmark y el conjunto SICK-R. De acuerdo con la tabla 5.3, los algoritmos de agrupamiento más utilizados fueron el algoritmo k-means, mezcla gaussiana y jerárquico aglomerativo.

Los agrupamientos para cada capa del modelo, tal como se especifica en la tabla 5.3, se exhiben visualmente a través de diagramas de dispersión en la figura 5.1 para el conjunto de prueba STS-Benchmark y en la figura 5.2 para el conjunto de prueba SICK-R. Se pueden apreciar similitudes en ambas figuras.

Cuadro 5.3: Algoritmos de agrupamiento usados para el análisis de cada capa para el conjunto de datos STS-Benchmark y conjunto SICK-R para el análisis de atención.

<b>Capa</b>	<b>Algoritmo</b>	<b>Hiperparámetros</b>	<b>K</b>
<b>STS-Benchmark</b>			
1	K-means	Distancia euclidiana	15
2	Mezcla gausiana	Covarianza esférica	6
3	Aglomerativo	Distancia euclidiana, enlace ward	3
4	Aglomerativo	Distancia euclidiana, enlace ward	4
5	K-means	Distancia euclidiana	13
6	K-means	Distancia euclidiana	6
7	Mezcla gausiana	Covarianza ligada	2
8	Mezcla gausiana	Covarianza diagonal	3
9	Aglomerativo	Distancia euclidiana, enlace completo	2
10	Mezcla gausiana	Covarianza esférica	6
11	Aglomerativo	Distancia euclidiana, enlace completo	2
12	Mezcla gausiana	Covarianza esférica	2
<b>SICK-R</b>			
1	Mezcla gausiana	Covarianza esférica	5
2	Aglomerativo	Distancia euclidiana, enlace ward	2
3	Mezcla gausiana	Covarianza ligada	3
4	Mezcla gausiana	Covarianza ligada	2
5	Mezcla gausiana	Covarianza esférica	3
6	Kmeans	Distancia euclidiana	4
7	Mezcla gausiana	Covarianza ligada	2
8	Mezcla gausiana	Covarianza completa	3
9	Mezcla gausiana	Covarianza ligada	2
10	Mezcla gausiana	Covarianza esférica	2
11	Mezcla gausiana	Covarianza completa	2
12	Mezcla gausiana	Covarianza completa	2

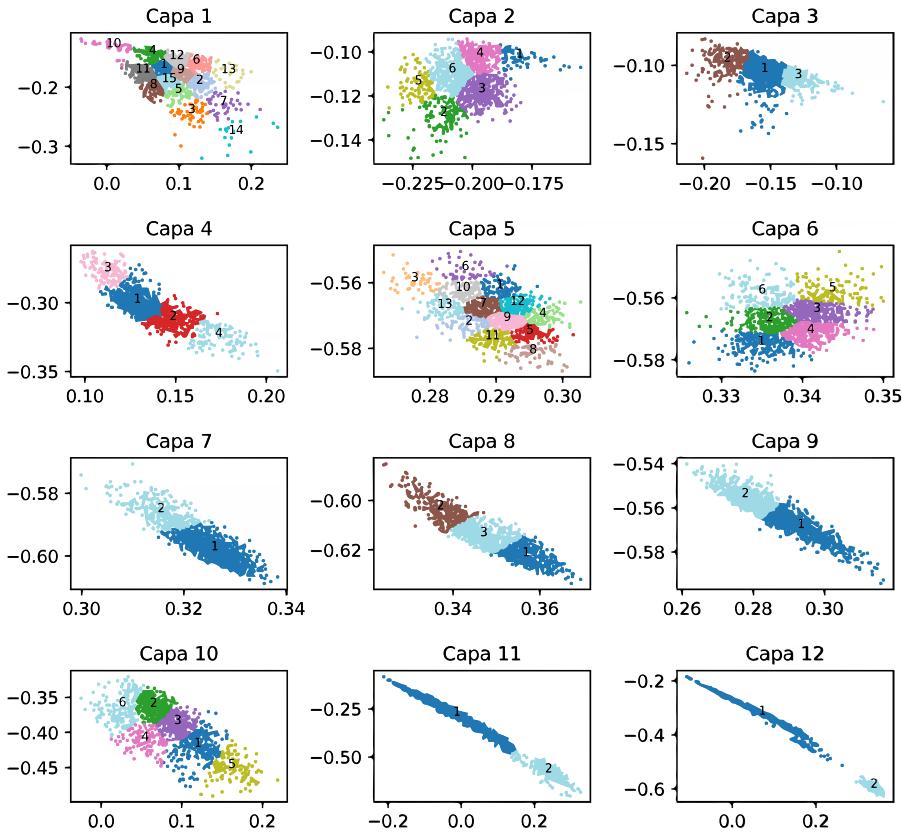


Figura 5.1: Diagramas de dispersión con agrupamientos del conjunto STS-Benchmark.

En ambas figuras (5.1 y 5.2), se destaca una dispersión más pronunciada en las atenciones de las capas iniciales del modelo (capas 1, 2, 3, 5 y 6). Por otro lado, se aprecia una menor dispersión en la mitad superior, especialmente en las últimas dos capas del modelo, corroborando las observaciones de [9]. Estos resultados se mantienen consistentes en ambos conjuntos de datos.

Es interesante señalar que las capas 7, 8, 9 y 10 exhiben una notable similitud entre sí, manteniendo una tendencia uniforme en ambos conjuntos con la misma inclinación y forma. Asimismo, las capas 11 y 12 siguen esta tendencia, pero presentan una apariencia más compacta.

Se observa que las capas 2, 3, 5 y 6 muestran patrones o formas distintas en los conjuntos STS-Benchmark y SICK-R. Es relevante destacar que las capas con una mayor

dispersión requirieron un número más elevado de agrupamientos, mientras que aquellas con menor dispersión en su mayoría se caracterizan por contar solo con dos agrupamientos.

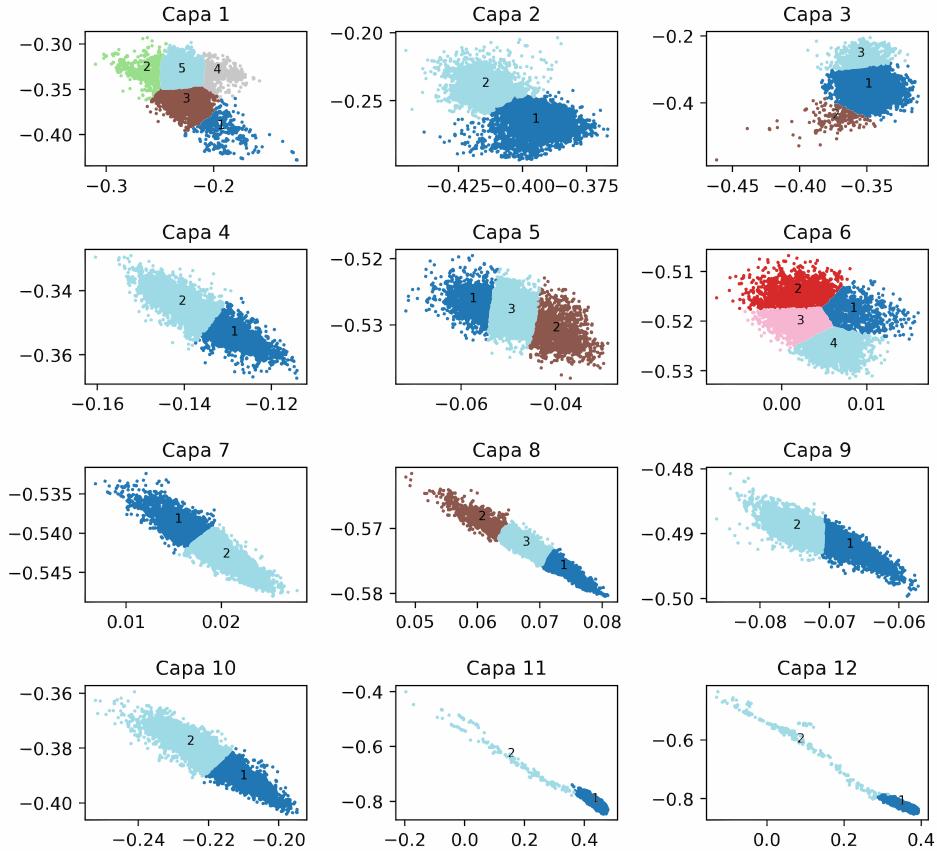


Figura 5.2: Diagramas de dispersión con agrupamientos del conjunto SICK-R.

Los agrupamientos fueron evaluados en términos de calidad mediante métricas intrínsecas, que incluyen el índice de silueta, índice de Davies-Bouldin, índice de Calinski Harabasz, inercia (utilizada específicamente para evaluar el algoritmo de k-means), así como el criterio de información bayesiano (BIC) y el criterio de información de Akaike (AIC) exclusivamente para el algoritmo de mezcla gaussiana. Los resultados de estas métricas para los conjuntos STS-Benchmark y SICK-R se detallan en la tabla 5.4.

En la tabla 5.4 se evidencia la dispersión de los datos en las primeras capas. Se destaca que las últimas dos capas en ambos modelos reciben las evaluaciones más sobresalientes.

Este fenómeno se atribuye a que los agrupamientos en estas capas exhiben una mayor cohesión entre sus elementos y una separación más pronunciada entre los grupos, contribuyendo así a una calidad superior en los resultados.

Cuadro 5.4: Resultados de métricas intrínsecas del conjunto STS-Benchmark y SICK-R

Capa	Silueta	Bouldin	Calinski	Inercia	BIC	AIC
<b>STS-Benchmark</b>						
1	0.387	0.7616	1093	0.23	-	-
2	0.357	0.784	932	-	-17559	-17679
3	0.435	0.695	1394	-	-	-
4	0.4312	0.6736	1909	-	-	-
5	0.3471	0.7919	1072	0.008	-	-
6	0.3493	0.8452	1076	0.015	-	-
7	0.5484	0.5935	2057	-	-22405	-22447
8	0.5019	0.6184	2917	-	-19799	-19872
9	0.5345	0.6285	2428	-	-	-
10	0.3765	0.8431	1704	-	-11646	-11767
11	0.678	0.3308	3760	-	-	-
12	0.7838	0.201	5959	-	-8200	-8236
<b>SICK-R</b>						
1	0.377	0.787	3788	-	-48958	-49082
2	0.501	0.705	7162	-	-	-
3	0.418	0.556	2596	-	-46068	-46139
4	0.514	0.662	7690	-	-76871	-76923
5	0.4541	0.706	9009	-	-79925	-79996
6	0.385	0.802	4322	0.04	-	-
7	0.523	0.648	7983	-	-94259	-94311
8	0.514	0.589	10895	-	-91960	-92071
9	0.522	0.652	8346	-	-89760	-89812
10	0.537	0.625	9157	-	-69640	-69685
11	0.896	0.465	10957	-	-56938	-57010
12	0.91	0.407	17662	-	-60878	-60950

### 5.2.2. Análisis de longitud de secuencia (LS)

El modelo muestra una atención significativa hacia la longitud de secuencia, especialmente en las capas iniciales. En el diagrama de caja para el conjunto de datos STS-Benchmark se puede visualizar en la figura 5.3 mientras que los diagramas de caja para el conjunto SICK-R se pueden observar en la figura 5.4.

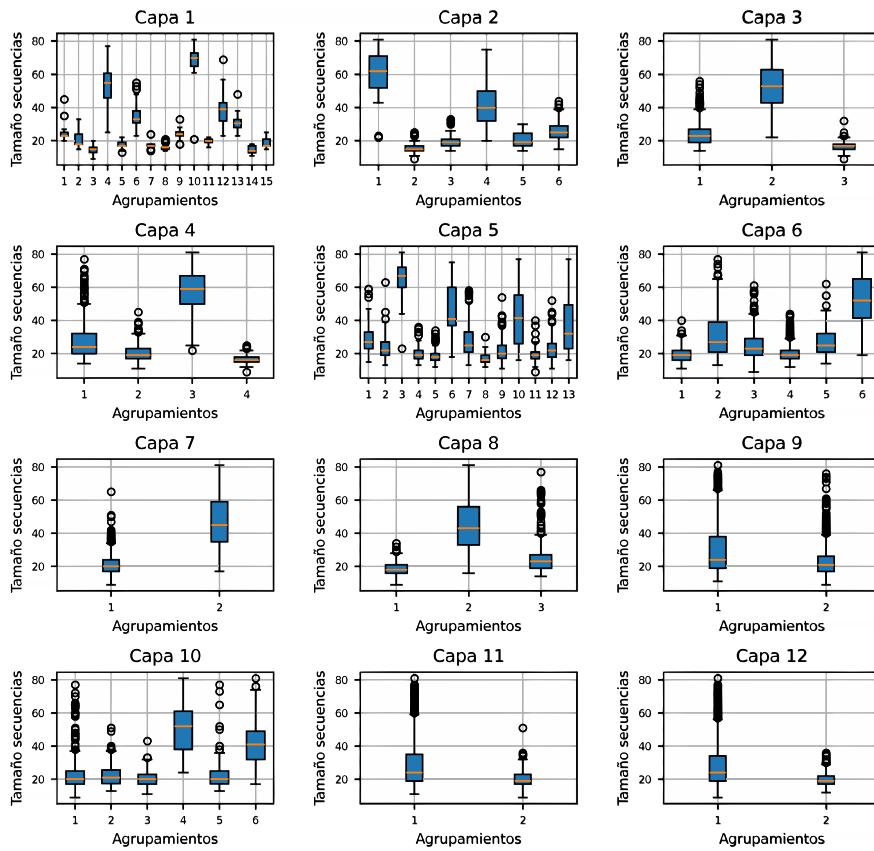


Figura 5.3: Diagramas de caja para análisis de longitud de secuencia de STS-Benchmark.

Según el análisis del conjunto STS-Benchmark en la figura 5.3, se evidencia que las primeras 8 capas del modelo exhiben grupos con rangos intercuartiles a niveles distintos, sugiriendo un enfoque del modelo en la longitud de secuencia. En contraste, en las últimas 4 capas, los rangos intercuartiles se mantienen en un mismo nivel, indicando que el modelo combina elementos con diferentes valores de longitud de secuencia. En relación al conjunto

SICK-R, ilustrado en la figura 5.4, se visualiza una tendencia similar a la del conjunto STS-Benchmark. No obstante, se observa que las capas 11 y 12 muestran un ligero enfoque en la longitud de secuencia.

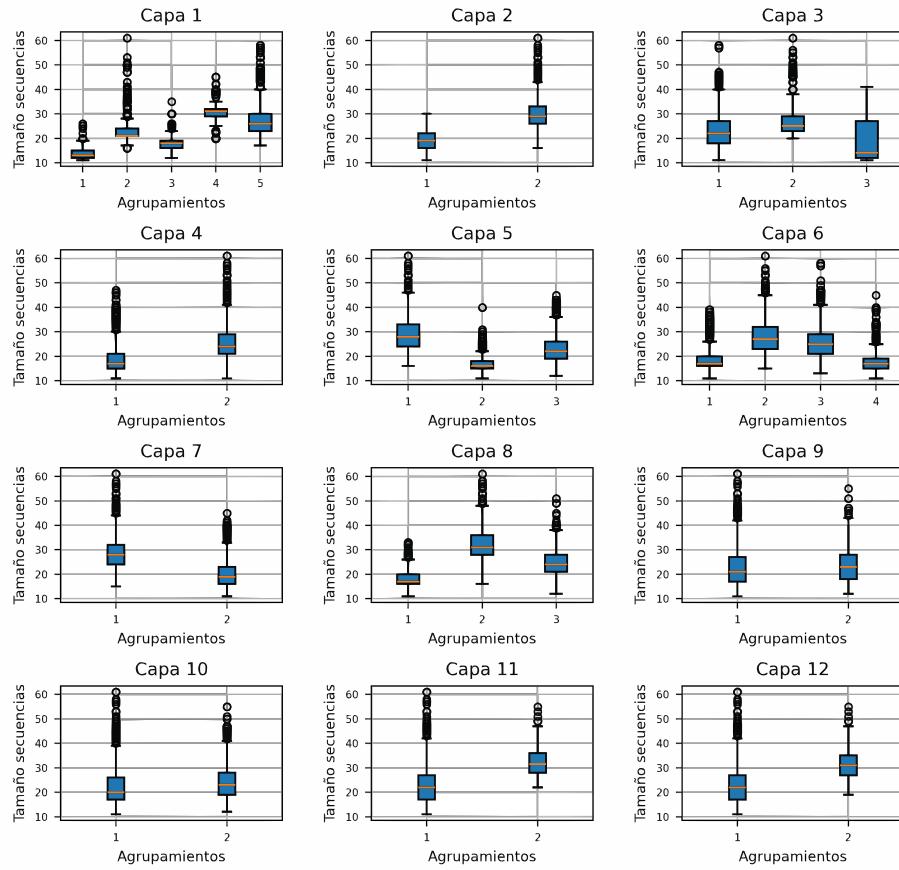


Figura 5.4: Diagramas de caja para análisis de longitud de secuencia de SICK-R.

Por otro lado, al analizar los valores p en las pruebas Kruskal-Wallis y Mann-Whitney U (consultar tabla 5.5), se revela gradualidad. Como recordatorio, cuanto más pequeño sea el valor p, más evidencia sugiere que los agrupamientos son diferentes, ya que implica el rechazo de la hipótesis nula. Por el contrario, un valor p grande refuerza la hipótesis nula. Tanto en el conjunto de datos STS-Benchmark como en SICK-R, se observa la tendencia de tener valores p muy pequeños, llegando incluso a 0, en las primeras capas.

En el caso del conjunto STS-Benchmark, se percibe una degradación a medida que

se avanza a lo largo de sus capas; las capas intermedias muestran una pérdida gradual de enfoque, mientras que las últimas 4 capas reflejan los valores más altos de p. En el conjunto SICK-R, se observa un patrón similar, con la excepción de la capa 3 que muestra un enfoque más débil en la longitud de secuencia. Aunque, al igual que en el conjunto STS-Benchmark, las últimas 4 capas en el modelo SICK-R presentan una disminución en la fuerza del enfoque en la longitud de secuencia. No obstante, el análisis de los valores p indica que el modelo intenta mantener el enfoque en la longitud de secuencia, aunque este es mucho más débil en las últimas capas.

Cuadro 5.5: Resultados del valor p de Kruskal-Wallis y Mann-Whitney U para longitud de secuencia (LS).

Capa	STS-B LS(p)	SICK-R LS(p)
1	<b>3.36e-243</b>	0.0
2	<b>1.06e-206</b>	0.0
3	<b>9.769e-168</b>	4.7e-84
4	7.57e-121	4.3e-292
5	6.207e-120	<b>0.0</b>
6	2.96e-105	<b>0.0</b>
7	1.34e-140	<b>0.0</b>
8	1.32e-162	<b>0.0</b>
9	3.69e-20	4.2e-10
10	3.94e-113	3.3e-36
11	8.87e-29	4.3e-44
12	5.06e-26	1.3e-58

Por otro lado, la herramienta visual no solo facilita el análisis exploratorio necesario para el análisis no supervisado, sino que también ofrece la capacidad de visualizar el comportamiento de los datos en términos de su longitud de secuencia. En esta herramienta, es posible ingresar varios rangos de valores de tamaño de secuencia y visualizar su comportamiento mediante el resultado de las muestras que se encuentran dentro de dichos rangos.

Como ejemplo, en la figura 5.5, se presentan los diagramas de dispersión de las capas 1, 4, 8 y 12 del conjunto de datos STS-Benchmark. En estos gráficos, se resaltan, a través de la herramienta visual, los rangos de longitud de secuencia de 9-15 (color rojo), 20-22 (color rosa) y 60-81 (azul claro), mientras que las secuencias que no cumplen con las dimensiones especificadas en los rangos se mantienen en azul oscuro. En la capa 1, se observa una marcada segmentación de los datos en función de su longitud, en la capa 4

se aprecia cómo comienzan a combinarse valores pequeños e intermedios de longitud de secuencia. En la capa 8, la combinación de valores bajos e intermedios es más evidente. Por último, en la capa 12, se observa una mezcla de valores intermedios y bajos, así como algunos valores del rango grande. Sin embargo, el modelo tiende a mantener los valores más grandes en un extremo.

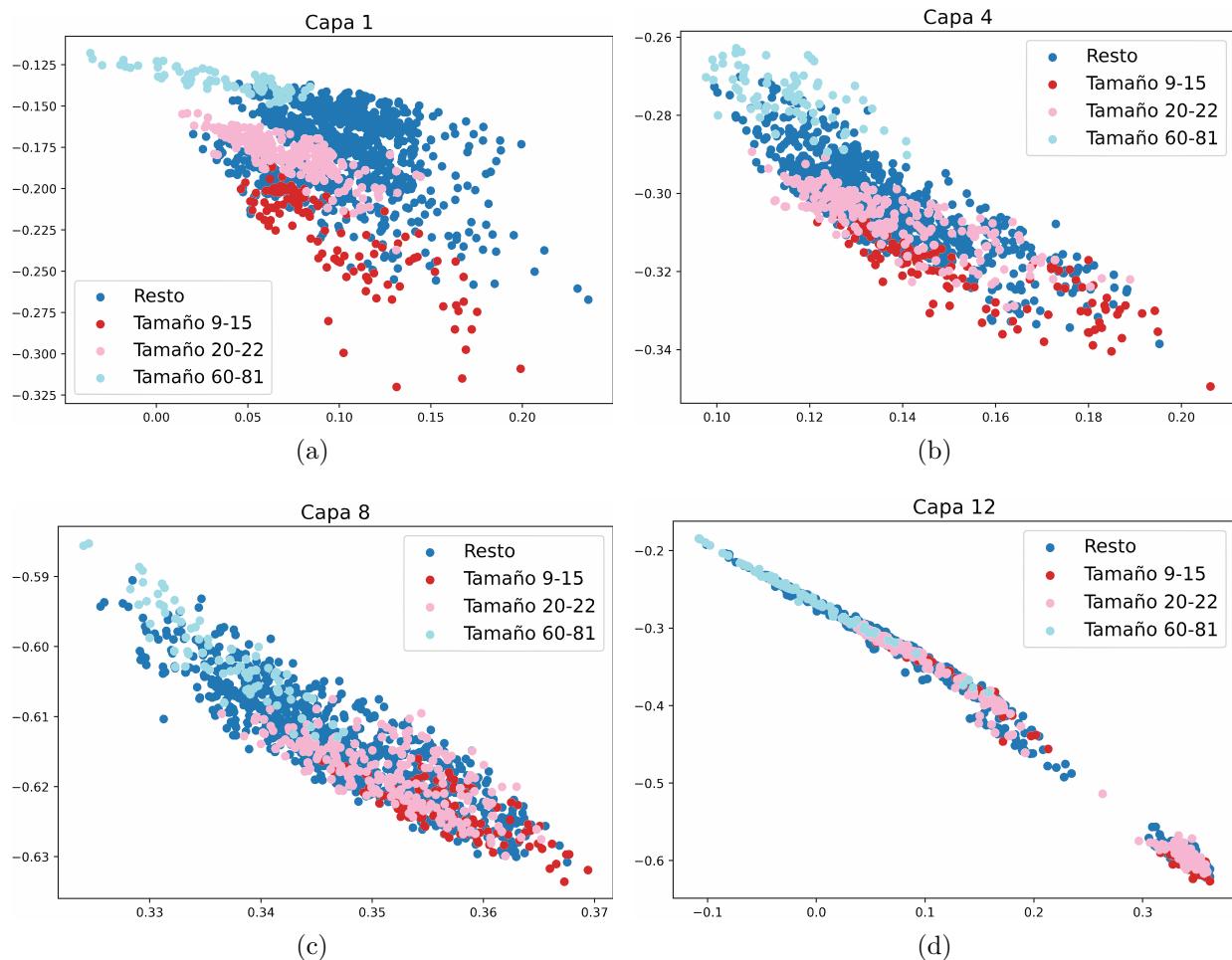


Figura 5.5: Diagramas de dispersión de las capas 1, 4, 8 y 12 señalando rangos de tamaño de secuencias para el conjunto STS-Benchmark.

En la figura 5.6, se presentan los diagramas de dispersión de las capas 1, 5, 9 y 11 del conjunto SICK-R. Se resaltan los rangos de longitud de secuencia entre 11-15 (color rojo), 20 (color rosa) y 40-61 (color azul claro), mientras que en azul oscuro se mantienen aquellos valores que no se encuentran en los rangos definidos. La capa 1 del conjunto SICK-R muestra una similitud notable con la capa 1 del conjunto STS-Benchmark, manteniendo

la misma tendencia al segmentar las longitudes de las secuencias. En la capa 5, comienza a observarse una mezcla de valores intermedios y bajos. Por otro lado, en las capas 9 y 11 se aprecia una combinación de valores pequeños, intermedios y grandes de longitud de secuencia.

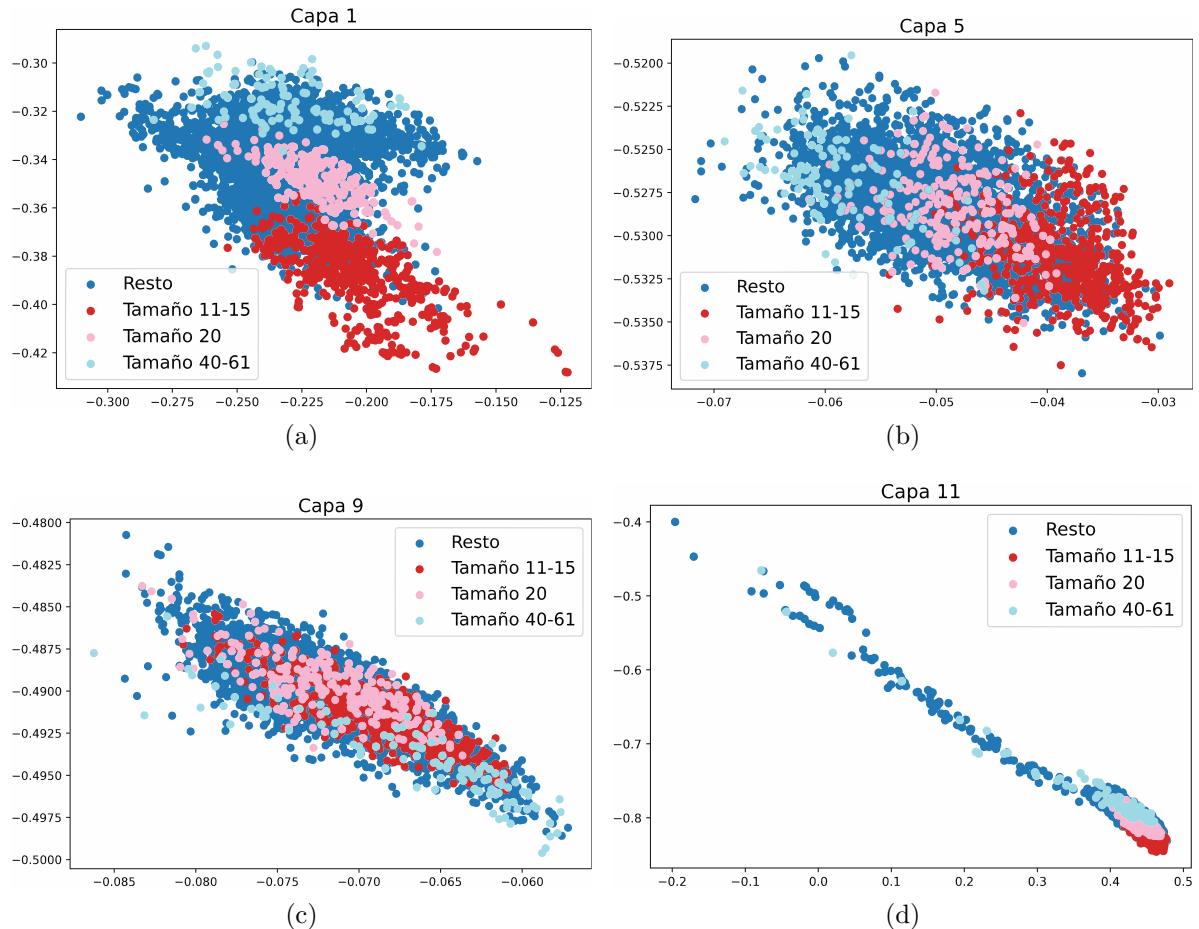


Figura 5.6: Diagramas de dispersión de las capas 1,5,9 y 11 señalando rangos de tamaño de secuencias para el conjunto SICK-R.

### 5.2.3. Análisis de similitud semántica (SS)

La similitud semántica se evidenció claramente a partir de las capas intermedias, extendiéndose hacia las capas superiores, aunque no se observó de manera tan marcada en las dos capas finales del modelo. Estas observaciones son consistentes y se pueden visualizar en los diagramas de caja, en los resultados de las pruebas estadísticas, así como

mediante la exploración de los datos con el respaldo de la herramienta visual.

El análisis a través de diagramas de caja de cada capa del modelo BERT para el conjunto de datos STS-Benchmark se presenta en la figura 5.7, mientras que para el conjunto SICK-R se puede visualizar en la figura 5.8.

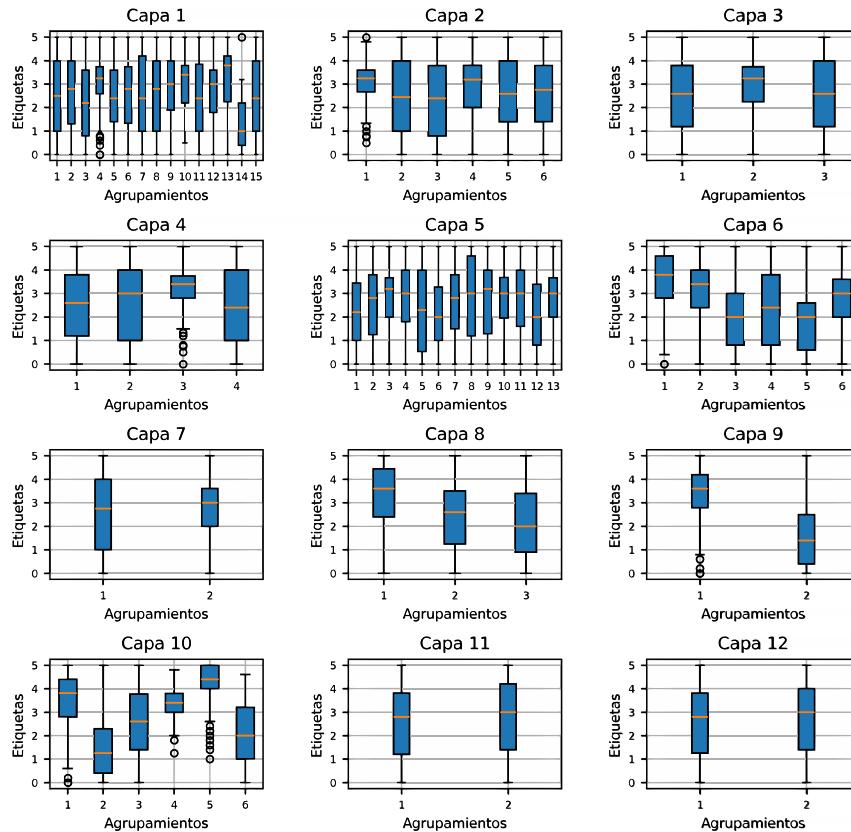


Figura 5.7: Diagramas de caja para analizar la similitud semántica en STS-Benchmark.

En el diagrama de caja correspondiente al conjunto de datos STS-Benchmark, destaca fácilmente el rendimiento sobresaliente de la capa 9. Se aprecia claramente cómo en esta capa, los rangos intercuartiles concentran valores diferentes. El grupo 1 tiende a mantener elementos con alta similitud semántica, mientras que el agrupamiento 2 mantiene elementos con similitud semántica baja. En la capa 10, se observa que el agrupamiento 5 exhibe valores de similitud semántica notablemente altos, mientras que el agrupamiento 2 concentra más del 50 % de sus datos en un rango de valores entre 0.4 y 2.2 de similitud.

Además, los agrupamientos 1, 3 y 5 de la capa 6 también presentan rangos intercuartiles diferentes, mostrando indicios de enfoque en la similitud semántica desde esta capa. Estas observaciones sugieren que en las capas 9 y 10 del conjunto STS-Benchmark, la similitud semántica adquiere relevancia, mostrando indicios de enfoque desde la capa 6.

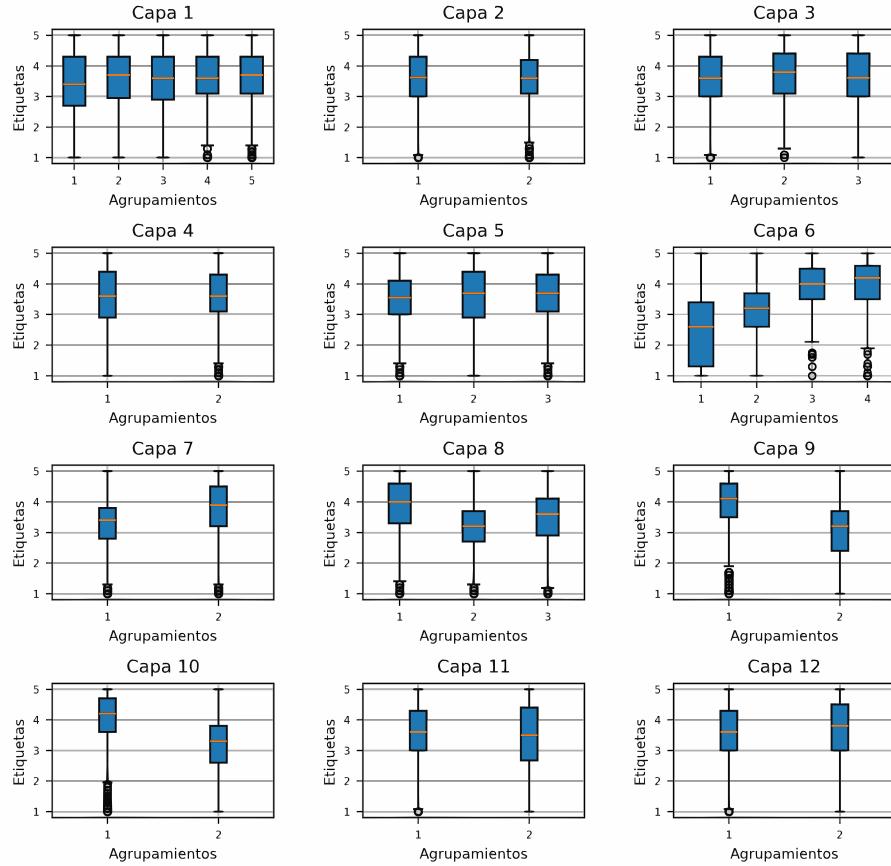


Figura 5.8: Diagramas de caja para analizar la similitud semántica en SICK-R.

En cuanto al conjunto de datos SICK-R, se observan resultados más notables en la capa 6, ya que los grupos 1, 3 y 4 tienen rangos de similitud semántica diferentes. El enfoque en la similitud semántica también es claro en las capas 9 y 10, donde los rangos intercuartiles exhiben alturas distintas.

En ambos conjuntos de datos, se evidencia la consistente tendencia de que, tanto en las primeras como en las últimas dos capas, los rangos intercuartiles para la característica de

similitud semántica se mantienen al mismo nivel. Este patrón indica que en estas capas, el modelo BERT no centra su atención en el aspecto de similitud semántica.

Los resultados de las pruebas Kruskal-Wallis y Mann-Whitney U se pueden observar en la tabla 5.6. En esta tabla, las capas 6, 8, 9 y 10 muestran un valor p muy pequeño, indicando que los agrupamientos son significativamente diferentes, es decir, que el modelo en estas capas se enfoca en la similitud semántica en ambos conjuntos de datos de forma consistente. En contraste para el resto de las capas, el valor p es significativamente mayor indicando que los grupos tienen muestras con valores de similitud semántica diversos.

Cuadro 5.6: Resultados del valor p de Kruskal-Wallis y Mann-Whitney U para similitud semántica (SS).

Capa	STS-B SS(p)	SICK-R SS(p)
1	0.00188	0.0048
2	0.00239	0.45
3	0.00058	0.26
4	0.00086	0.511
5	0.00101	3.9e-5
6	<b>2.31e-47</b>	<b>0.0</b>
7	0.1377	1.8e-76
8	<b>4.545e-34</b>	<b>5.8e-92</b>
9	<b>1.21e-107</b>	<b>1.6e-259</b>
10	<b>1.94e-113</b>	<b>6.6e-279</b>
11	0.00075	0.237
12	0.0353	0.0798

El comportamiento de los datos, según la herramienta visual, se ilustra en la figura 5.9. En esta representación, se exhiben los diagramas de dispersión de las capas 2, 7, 10 y 12, considerando similitudes semánticas de 0 (color café) y 5 (color azul claro). Las capas 2 y 12 muestran una notable dispersión o combinación de las muestras con respecto a las etiquetas de similitud semántica. En el caso de la capa 7, no se observa tanta combinación de las muestras, apreciándose una ligera segmentación entre las etiquetas. En contraste, la capa 10 presenta una clara separación entre las etiquetas con similitud 0 (ubicadas en la parte superior) y las etiquetas con similitud 5 (ubicadas en la parte inferior).

En la figura 5.10, se presentan los diagramas de dispersión que destacan la similitud semántica para el conjunto de datos SICK-R, considerando las etiquetas de similitud 1

(color café) y 5 (color azul claro). De acuerdo con la herramienta visual, se observa cómo en las capas 2 y 12 las muestras están combinadas en relación con la etiqueta de similitud semántica. Sin embargo, en las capas 6 y 10, se aprecia una clara separación entre las etiquetas de similitud 1 y 5, indicando que en estas capas el modelo se enfoca en la semántica.

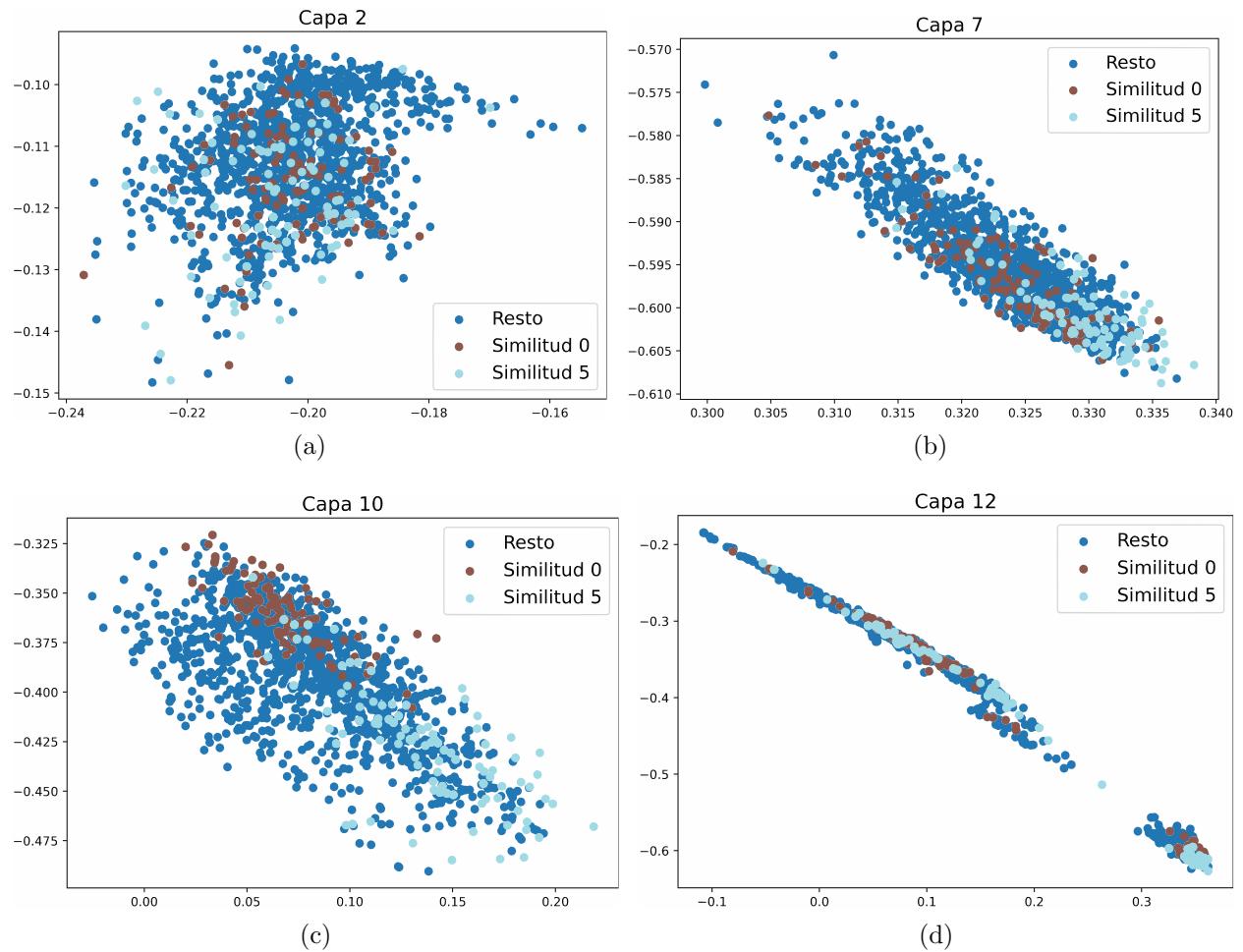


Figura 5.9: Diagramas de dispersión resaltando las muestras con similitudes semánticas de 0 y 5 para el conjunto STS-Benchmark.

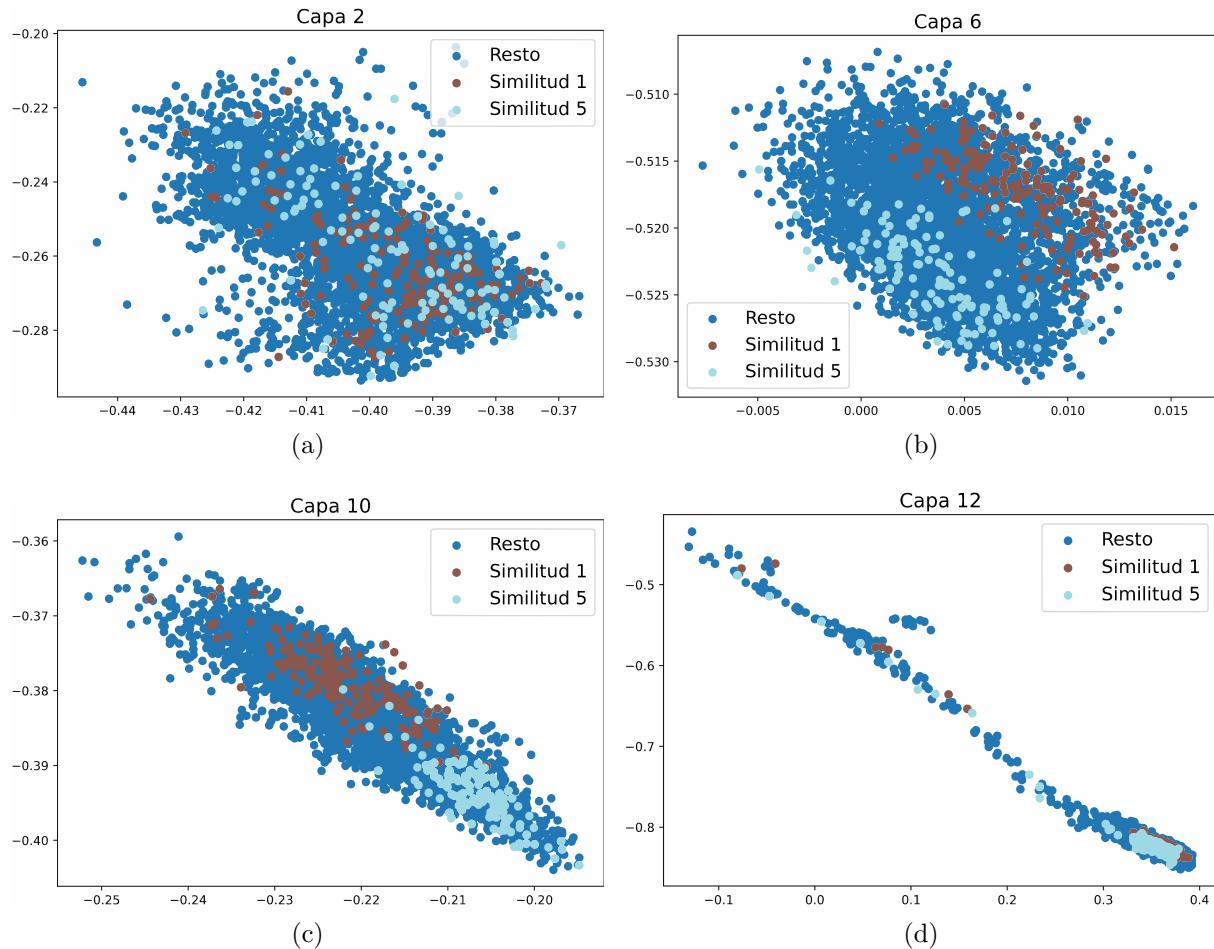


Figura 5.10: Diagramas de dispersión resaltando las muestras con similitudes semánticas de 1 y 5 para el conjunto SICK-R

#### 5.2.4. Análisis de similitud de estructura grammatical (SEG)

El análisis de similitud en la estructura grammatical exhibió mayor variabilidad y menos consistencia en comparación con los aspectos de similitud semántica y longitud de secuencia. A pesar de esto, se observa una tendencia general en la mayoría de las capas a agrupar las estructuras gramaticales similares.

El análisis, respaldado por diagramas de caja, para el conjunto de datos STS-Benchmark se presenta en la figura 5.11. En este gráfico, se destaca que en las capas 3, 7 y 8 los rangos intercuartiles tienen alturas similares, indicando que el modelo no mantiene un enfoque claro en la similitud de estructura grammatical en estas capas. Por otro lado, el resto de las capas muestran, en mayor o menor medida, un enfoque en este aspecto.

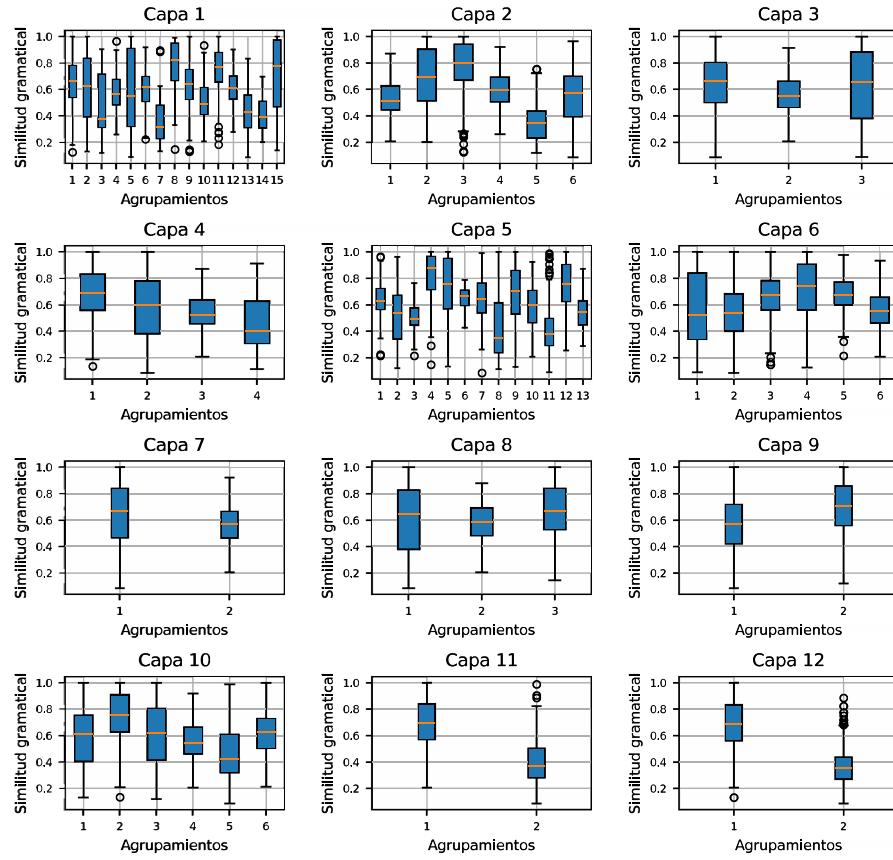


Figura 5.11: Diagramas de caja para analizar la similitud de estructura gramatical de la secuencia más frecuente con respecto a otras estructuras del conjunto STS-Benchmark.

En relación con el conjunto SICK-R, los diagramas de caja se visualizan en la figura 5.12. Se observa que, para este conjunto, las capas 3, 9 y 10 presentan rangos intercuartiles a la misma altura, lo que sugiere que el modelo no parece centrarse en la similitud de estructura gramatical en estas capas. En comparación con el conjunto STS-Benchmark, las capas 11 y 12 muestran un menor enfoque en esta característica, mientras que la capa 7 y 8 presentan un mayor enfoque. Sin embargo, en general, el análisis de caja para la similitud de estructura gramatical no revela un patrón claro en esta característica.

Los resultados de las pruebas estadísticas de Kruskal-Wallis y Mann-Whitney U, específicamente para la similitud de estructura gramatical, se presentan en la tabla 5.7. De acuerdo con estos resultados, se evidencia consistencia en las capas 2 y 5 en ambos

conjuntos de datos, indicando una atención destacada del modelo hacia esta característica. Asimismo, la capa 3 muestra consistencia al indicar que, en este nivel, el modelo no presta atención particular a la similitud de estructura gramatical. No obstante, se observa una pérdida de consistencia en el resto de las capas, ya que algunas muestran un mayor enfoque en un conjunto, pero no en el otro conjunto de datos.

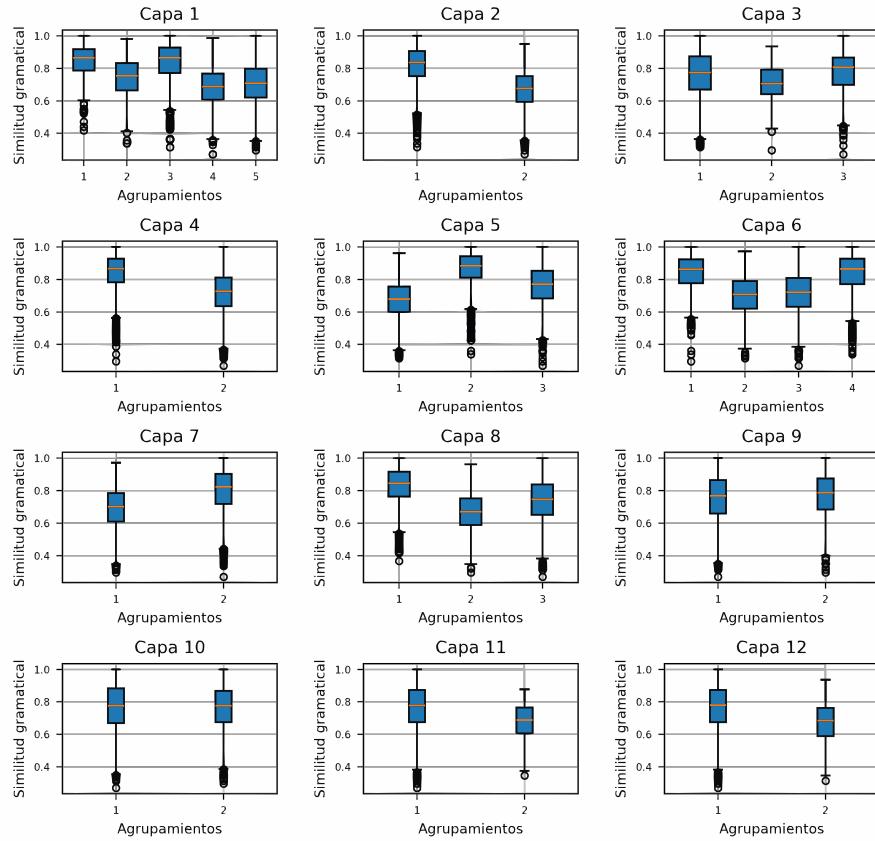


Figura 5.12: Diagramas de caja para analizar la similitud de estructura gramatical de la secuencia más frecuente con respecto a otras estructuras del conjunto SICK-R.

El comportamiento de los datos con respecto a la similitud de estructura gramatical se muestra en la figura 5.13. En esta representación, se presentan los diagramas de dispersión correspondientes a las capas 2, 3, 8 y 12 del conjunto STS-Benchmark. A través de la herramienta visual, fue posible destacar en color azul claro las estructuras con similitud coseno igual o mayor a 0.95 en comparación con la estructura más frecuente del conjunto

STS-Benchmark.

Cuadro 5.7: Resultados de los valores p obtenidos mediante las pruebas de Kruskal-Wallis y Mann-Whitney U para la similitud de estructura gramatical (SEG) en los conjuntos STS-Benchmark y SICK-R.

Capa	STS-B SEG(p)	SICK-R SEG(p)
1	3.38e-46	<b>4.1e-268</b>
2	<b>4.62e-85</b>	<b>0.0</b>
3	1.079e-08	2.5e-11
4	1.63e-34	2e-210
5	<b>9.22e-64</b>	<b>0.0</b>
6	3.75e-24	<b>1.1e-244</b>
7	1.58e-12	2.8e-167
8	1.99e-09	4.6e-221
9	1.23e-22	3.1e-05
10	2.48e-47	0.198
11	<b>8.82e-95</b>	1.7e-13
12	<b>8.82e-93</b>	1.8e-21

De acuerdo con la figura 5.13, las estructuras similares tienden a mantenerse cercanas entre sí, con la diferencia de que algunas capas muestran mayor o menor dispersión. Por ejemplo, la capa 8 o la capa 2 presentan una mayor dispersión en comparación con la capa 12, aunque en todas estas capas se observa la tendencia a agrupar estructuras similares.

Los diagramas de dispersión que destacan el aspecto de similitud de estructura gramatical para el conjunto de datos SICK-R se pueden apreciar en la figura 5.14. Similar al conjunto STS-Benchmark, con el respaldo de la herramienta visual, se observa que las estructuras gramaticales similares tienden a mantenerse cercanas entre sí. Por ejemplo, en la figura se aprecia que las capas 3 y 10 exhiben una mayor dispersión de las estructuras similares en comparación con las capas 2 o 12. En particular, la capa 12 muestra una mayor cohesión de las muestras con estructuras gramaticales similares.

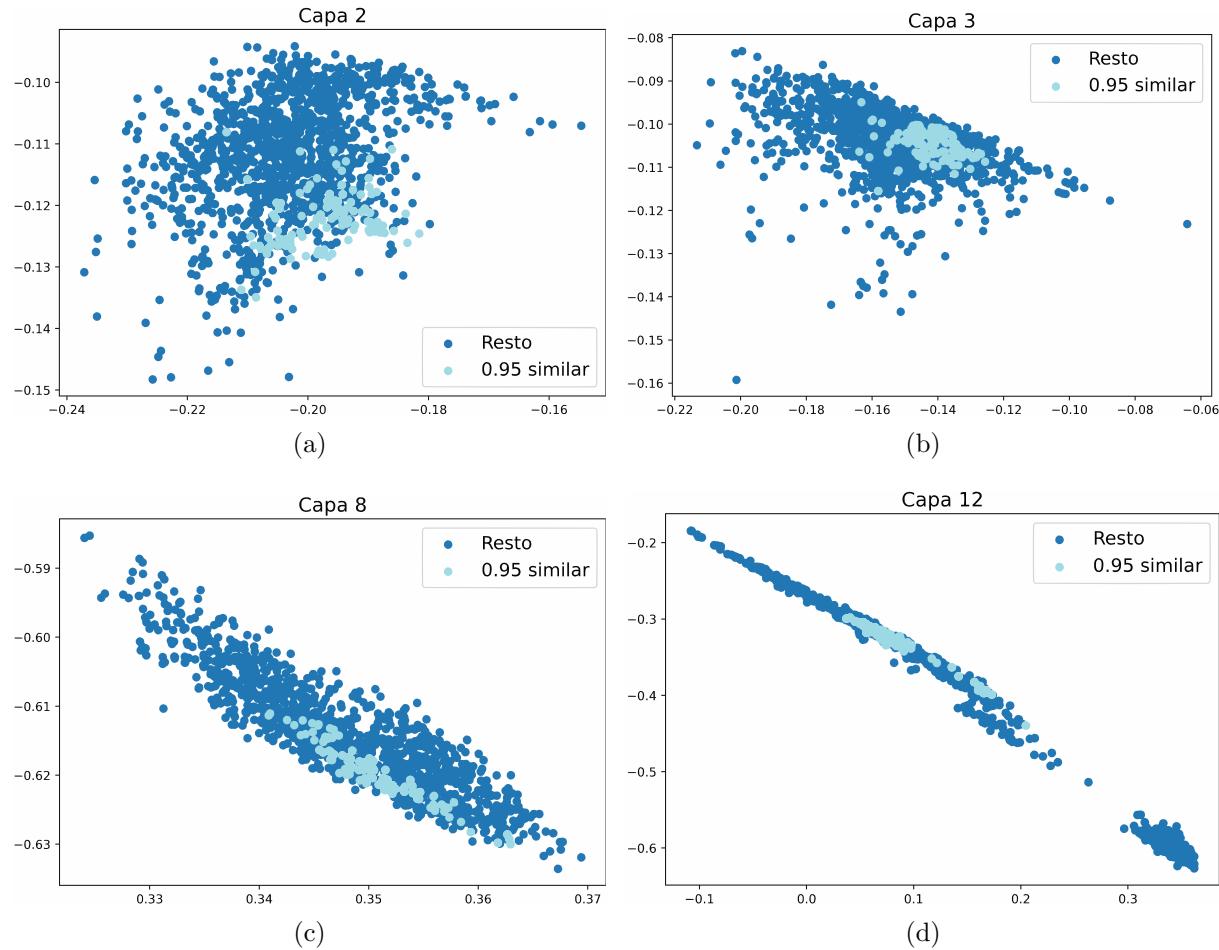


Figura 5.13: Diagramas de dispersión de las capas 2, 3, 8 y 12, resaltando estructuras con similitud mayor o igual a 0.95 para la estructura más frecuente del conjunto STS-Benchmark.

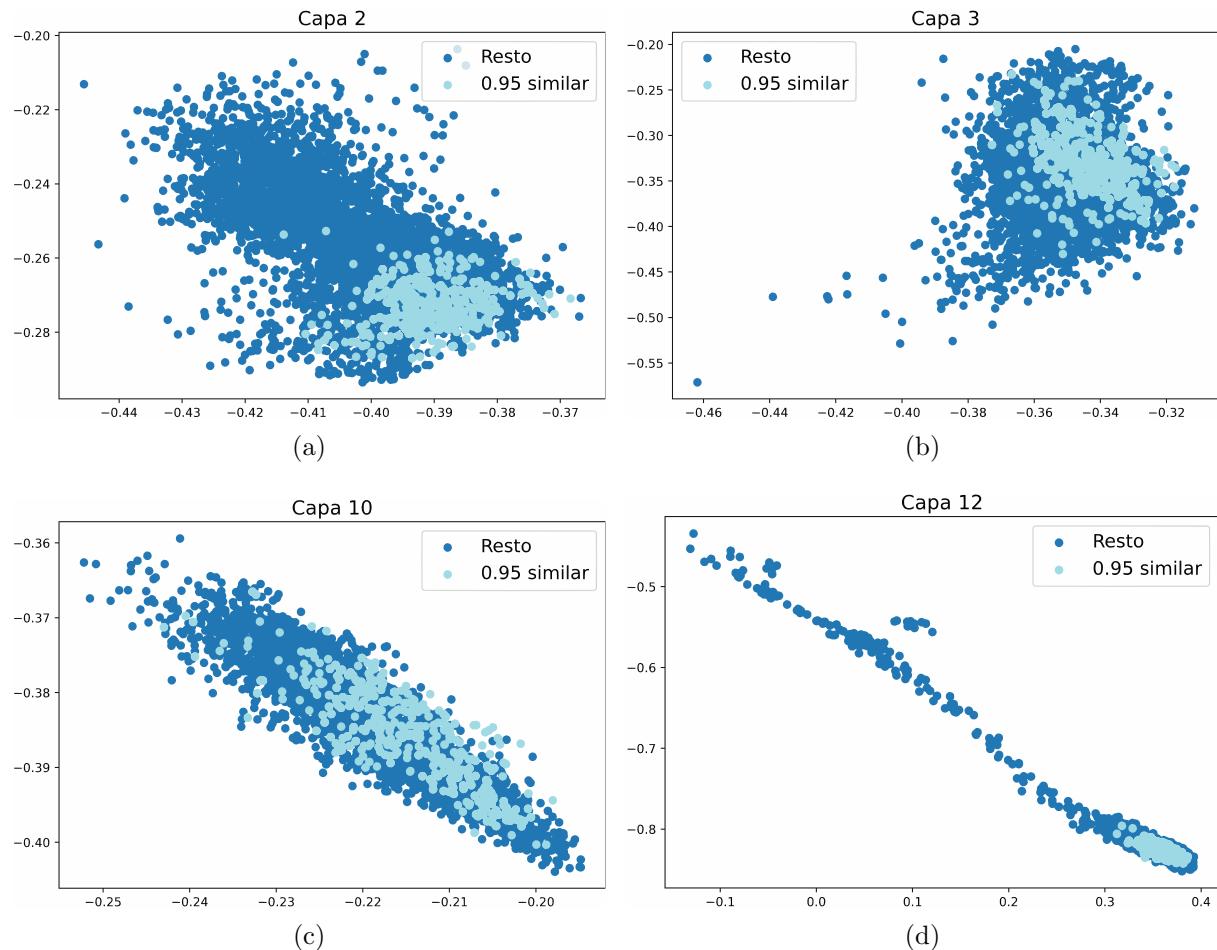


Figura 5.14: Diagramas de dispersión de las capas 2, 3, 10 y 12, resaltando estructuras con similitud mayor o igual a 0.95 para la estructura más frecuente del conjunto SICK-R.

# Capítulo 6

## Conclusiones y trabajo futuro

### 6.1. Conclusiones

De acuerdo con la hipótesis planteada, se concluye que ha diferencia de lo establecido en la hipótesis planteada en este trabajo, el token de clasificación CLS del modelo BERT generó una mejor abstracción lingüística que el resto de los métodos propuestos (métodos de agregación promedio y máximo e incluso la red recurrente LSTM), en este sentido la hipótesis planteada fue incorrecta.

Por otra parte, se ha evidenciado que las autoatenciones del modelo BERT logran abstractar aspectos lingüísticos. Además, la exploración de la interpretabilidad lingüística de BERT mediante el enfoque de aprendizaje no supervisado se revela como una herramienta confiable para dicho propósito. De esta manera, se confirman como correctos estos dos puntos planteados en la hipótesis del presente trabajo. La afirmación de que el aprendizaje no supervisado constituye una herramienta fiable para abordar la interpretabilidad de BERT se respalda al observar que los resultados obtenidos concuerdan con las conclusiones establecidas por trabajos del estado del arte, que adoptaron el enfoque supervisado, mediante el uso de clasificadores o sondas.

En este trabajo se observaron matices distintos con respecto a los resultados de la literatura. Por ejemplo, se notó que la longitud de las sentencias recibe una atención destacada en las capas iniciales del modelo, manteniendo esta tendencia hasta capas intermedias, aunque con mucha menor intensidad. Asimismo, el modelo mostró enfoque en la similitud semántica en las capas 6, 8, 9 y 10, es decir, desde capas intermedias y superiores, pero no específicamente en las últimas capas. En cuanto a la similitud de estructura gramatical, los resultados no mostraron patrones tan claros como en los otros dos aspectos, aunque se observó que las estructuras gramaticales similares tienden a agruparse en mayor o menor medida a lo largo de las capas.

Se identificó consistencia con la literatura en el comportamiento de las atenciones del modelo BERT, al notar que las capas iniciales del modelo BERT tienden a tener mayor entropía que las capas finales, tal y como se evidencia en los diagramas de dispersión de este trabajo.

También, en la mayoría de los experimentos realizados, se observó una notable consistencia entre los resultados de ambos conjuntos de datos (STS-Benchmark y SICK-R), mostrando solo diferencias en el aspecto de similitud de estructura gramatical.

Finalmente, se concluye que la metodología utilizada para generar abstracciones de 2 dimensiones a partir de las atenciones del modelo BERT, así como el diseño del modelo autocodificador, resultaron efectivos para lograr una buena abstracción de la información a pesar de la considerable reducción en la dimensionalidad. La metodología para el análisis de agrupamientos también se considera exitosa, a pesar de los desafíos iniciales para determinar el número óptimo de agrupamientos, los cuales se contrarrestaron eficazmente mediante la herramienta visual, que proporcionó una visión y entendimiento de los datos para cada aspecto lingüístico analizado.

## 6.2. Aportaciones

El estudio resalta la eficacia del token de clasificación CLS en comparación con los métodos propuestos para abordar la tarea de similitud semántica. Este hallazgo conduce a una reflexión detallada sobre las posibles razones subyacentes, enriqueciendo así la investigación.

La contribución de este trabajo en la interpretabilidad del modelo BERT se manifiesta al analizar diversos aspectos lingüísticos mediante el enfoque del aprendizaje no supervisado. Este enfoque, se ha utilizado muy poco en la literatura para abordar la interpretabilidad de BERT, sin embargo, mostró ser tan confiable y valioso como el enfoque supervisado con sondas o clasificadores. Además, se proporciona una visión del comportamiento de las atenciones a lo largo de las capas, un aspecto poco explorado en el estado del arte.

Se introduce una herramienta visual diseñada para facilitar el análisis exploratorio del aprendizaje no supervisado, el cual a menudo podría resultar exhaustivo. Esta herramienta integra un autocodificador LSTM que reduce las atenciones a la dimensionalidad seleccionada por el usuario. Facilita la elección del número de agrupamientos para algoritmos específicos y ofrece herramientas analíticas, como diagramas de caja y evaluaciones estadísticas. Aunque no se utilizó la sección de análisis de componentes principales en este trabajo debido a limitaciones de tiempo, la herramienta abarca la comprensión de los datos en diversos aspectos lingüísticos, como análisis de tópicos, longitud de secuencia,

similitud semántica y similitud de estructura gramatical. Es destacable que esta herramienta esté desarrollada en Python, lo que le confiere mayor versatilidad y agilidad en el desarrollo.

### 6.3. Trabajo a Futuro

A partir de este trabajo surgen diversas ideas que podrían ser objeto de desarrollo en futuras investigaciones. A continuación, se enumeran:

- La herramienta visual desarrollada como apoyo para el análisis exploratorio podría expandirse para incorporar algoritmos como análisis de componentes principales (PCA) o la incrustación estocástica de vecinos distribuidos  $t$  (T-SNE). Esta ampliación permitiría realizar pruebas experimentales con una reducción dimensional más alta y aplicar PCA o T-SNE para análisis y visualización.
- Sería interesante aplicar el mismo análisis de interpretabilidad realizado en las atenciones del modelo BERT a otros componentes, como el token de clasificación  $CLS$  o las representaciones de salida de cada bloque codificador. Esta tarea sería sencilla de realizar, ya que la herramienta visual está preparada para ello; solo sería necesario obtener las representaciones en 2 dimensiones.
- Todos los análisis de interpretabilidad de las autoatenciones del modelo se llevaron a cabo mediante la implementación de un ajuste fino. No obstante, sería interesante contrastar los resultados del presente trabajo con un análisis sin ajuste fino. Al igual que en el punto anterior, la herramienta visual está lista para este análisis; solo sería necesario obtener los vectores en 2 dimensiones.
- Actualmente, la herramienta visual cuenta con una sección de análisis exploratorio para tópicos, que debido a limitaciones de tiempo no pudo incluirse en el presente trabajo. Sin embargo, valdría la pena concluir este análisis. También sería enriquecedor llevar a cabo otros tipos de análisis sobre aspectos lingüísticos, como análisis de sentimientos, homonimia, sinonimia, etc.
- Dada la inconsistencia en los resultados de la similitud de estructura gramatical, sería útil profundizar en el ámbito de la sintaxis, quizás proponiendo otro enfoque o tarea lingüística asociada a este aspecto.
- Finalmente, durante esta investigación, se observaron varios caminos de estudio que podrían aportar significativamente a la comunidad científica. Por ejemplo, aunque

la mayoría de los trabajos se centran en brindar explicabilidad mediante un análisis por capa del modelo BERT, sería interesante realizar un análisis de interdependencia entre capas. Esto implica examinar cómo una capa influye en las capas posteriores a ella. Una forma de abordar este problema podría ser a través de la programación dinámica.

Para ampliar la investigación en relación con cualquiera de los puntos mencionados, se invita a utilizar como base el código de la herramienta visual desarrollada para facilitar el análisis exploratorio y comprender el comportamiento de los datos en cada aspecto lingüístico analizado en este trabajo. El repositorio que alberga el código de la herramienta visual para el análisis de interpretabilidad está disponible en el siguiente enlace: [[https://github.com/amsedel/interpretability\\_BERT](https://github.com/amsedel/interpretability_BERT)].

De manera similar, el repositorio asociado a la evaluación de la similitud semántica mediante los métodos propuestos y el token CLS se encuentra en: [[https://github.com/amsedel/semantic\\_similarity\\_BERT](https://github.com/amsedel/semantic_similarity_BERT)].

# Bibliografía

- [1] R. Xu and D. Wunsch, *Clustering*. John Wiley & Sons, 2008.
- [2] C. C. Aggarwal and C. K. Reddy, *Data Clustering: Algorithms and Applications*. Chapman and Hall/CRC, 2014.
- [3] Y. Bengio, I. Goodfellow, and A. Courville, *Deep learning*, vol. 1. MIT press Cambridge, MA, USA, 2017.
- [4] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, “Attention is all you need,” *Advances in neural information processing systems*, vol. 30, 2017.
- [5] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, “Bert: Pre-training of deep bidirectional transformers for language understanding,” *arXiv preprint arXiv:1810.04805*, 2018.
- [6] J. Niu, W. Lu, and G. Penn, “Does bert rediscover a classical nlp pipeline?,” in *Proceedings of the 29th International Conference on Computational Linguistics*, pp. 3143–3153, 2022.
- [7] J. Hewitt and C. D. Manning, “A structural probe for finding syntax in word representations,” in *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pp. 4129–4138, 2019.
- [8] I. Tenney, D. Das, and E. Pavlick, “Bert rediscovers the classical nlp pipeline,” *arXiv preprint arXiv:1905.05950*, 2019.
- [9] K. Clark, U. Khandelwal, O. Levy, and C. D. Manning, “What does bert look at? an analysis of bert’s attention,” *arXiv preprint arXiv:1906.04341*, 2019.

- [10] G. Jawahar, B. Sagot, and D. Seddah, “What does bert learn about the structure of language?,” in *ACL 2019-57th Annual Meeting of the Association for Computational Linguistics*, 2019.
- [11] E. Reif, A. Yuan, M. Wattenberg, F. B. Viegas, A. Coenen, A. Pearce, and B. Kim, “Visualizing and measuring the geometry of bert,” *Advances in Neural Information Processing Systems*, vol. 32, 2019.
- [12] S. Jain and B. C. Wallace, “Attention is not explanation,” *arXiv preprint arXiv:1902.10186*, 2019.
- [13] V. W. Anelli, G. M. Biancofiore, A. De Bellis, T. Di Noia, and E. Di Sciascio, “Interpretability of bert latent space through knowledge graphs,” in *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*, pp. 3806–3810, 2022.
- [14] Y. Gupta, “Chat gpt and gpt 3 detailed architecture study-deep nlp horse,” 2023. Medium article.
- [15] P. P. Ray, “Chatgpt: A comprehensive review on background, applications, key challenges, bias, ethics, limitations and future scope,” *Internet of Things and Cyber-Physical Systems*, 2023.
- [16] D. Hidalgo, “La ambigüedad lingüística,” *Revista de Filología y Lingüística de la Universidad de Costa Rica*, vol. 4, no. 2, pp. 39–46, 1978.
- [17] O. Kolesnikova, “Exposición del laboratorio de procesamiento de lenguaje natural,” (Ciudad de México, México), Centro de Investigación en Computación, 2022.
- [18] C. Zapata, K. Palomino, and R. Rosero, “A method for coordinative and prepositional syntactic disambiguation,” *Dyna*, vol. 75, no. 156, pp. 29–42, 2008.
- [19] K. P. Murphy, *Probabilistic machine learning: an introduction*. MIT press, 2022.
- [20] E. Grimson, J. Guttag, and A. Bell, “Clustering in ’introduction to computational thinking and data science’,” 2016. MIT OpenCourseWare video.
- [21] G. Gan, C. Ma, and J. Wu, *Data clustering: theory, algorithms, and applications*. SIAM, 2020.
- [22] H.-H. Bock, “On some significance tests in cluster analysis,” *Journal of classification*, vol. 2, pp. 77–108, 1985.

- [23] J. Carmichael and R. Julius, “Finding natural clusters,” *Systematic Biology*, vol. 17, no. 2, pp. 144–150, 1968.
- [24] B. S. Everitt, S. Landau, M. Leese, and D. Stahl, *Cluster analysis*. John Wiley & Sons, 2011.
- [25] S. Rogers and M. Girolami, *A first course in machine learning*. Chapman and Hall/CRC, 2016.
- [26] F. Bach and M. Jordan, “Learning spectral clustering,” *Advances in neural information processing systems*, vol. 16, 2003.
- [27] A. Géron, *Hands-on machine learning with Scikit-Learn, Keras, and TensorFlow*. O'Reilly Media, Inc., 2022.
- [28] R. E. Walpole, R. H. Myers, S. L. Myers, and K. Ye, “Probability and statistics for engineering and science,” *Pearson Education*, pp. 430–435, 2012.
- [29] O. Bello, “Análisis multivariado - anova y kruskal wallis”, 2023. Maestría en Ciencias Sociomédicas.
- [30] G. W. Corder and D. I. Foreman, *Nonparametric statistics: A step-by-step approach*. John Wiley & Sons, 2014.
- [31] D. Rothman, *Transformers for Natural Language Processing: Build innovative deep neural network architectures for NLP with Python, PyTorch, TensorFlow, BERT, RoBERTa, and more*. Packt Publishing Ltd, 2021.
- [32] W. Wong, “What is teacher forcing?,” *Towards Data Science*, Oct 2019. Consultado el 17 de marzo de 2023.
- [33] S. Ioffe and C. Szegedy, “Batch normalization: Accelerating deep network training by reducing internal covariate shift,” in *International conference on machine learning*, pp. 448–456, pmlr, 2015.
- [34] J. L. Ba, J. R. Kiros, and G. E. Hinton, “Layer normalization,” *arXiv preprint arXiv:1607.06450*, 2016.
- [35] R. Sennrich, B. Haddow, and A. Birch, “Neural machine translation of rare words with subword units,” *arXiv preprint arXiv:1508.07909*, 2015.

- [36] M. E. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee, and L. Zettlemoyer, “Deep contextualized word representations,” in *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, (New Orleans, Louisiana), pp. 2227–2237, Association for Computational Linguistics, June 2018.
- [37] A. Radford, K. Narasimhan, T. Salimans, I. Sutskever, *et al.*, “Improving language understanding by generative pre-training,” 2018.
- [38] A. Zhang, Z. C. Lipton, M. Li, and A. J. Smola, “Dive into deep learning,” *arXiv preprint arXiv:2106.11342*, 2021.
- [39] N. Reimers and I. Gurevych, “Sentence-bert: Sentence embeddings using siamese bert-networks,” *arXiv preprint arXiv:1908.10084*, 2019.
- [40] Y. Wang, F. Liu, K. Verspoor, and T. Baldwin, “Evaluating the utility of model configurations and data augmentation on clinical semantic textual similarity,” in *Proceedings of the 19th SIGBioMed workshop on biomedical language processing*, pp. 105–111, 2020.
- [41] E. Voita, D. Talbot, F. Moiseev, R. Sennrich, and I. Titov, “Analyzing multi-head self-attention: Specialized heads do the heavy lifting, the rest can be pruned,” *arXiv preprint arXiv:1905.09418*, 2019.
- [42] P. Michel, O. Levy, and G. Neubig, “Are sixteen heads really better than one?,” *Advances in neural information processing systems*, vol. 32, 2019.
- [43] M. A. Gordon, K. Duh, and N. Andrews, “Compressing bert: Studying the effects of weight pruning on transfer learning,” *arXiv preprint arXiv:2002.08307*, 2020.
- [44] D. Yenicelik, F. Schmidt, and Y. Kilcher, “How does bert capture semantics? a closer look at polysemous words,” in *Proceedings of the Third BlackboxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP*, pp. 156–162, 2020.
- [45] G. Wiedemann, S. Remus, A. Chawla, and C. Biemann, “Does bert make any sense? interpretable word sense disambiguation with contextualized embeddings,” *arXiv preprint arXiv:1909.10430*, 2019.
- [46] J. Novikova and K. Shkaruta, “Deck: Behavioral tests to improve interpretability and generalizability of bert models detecting depression from text,” *arXiv preprint arXiv:2209.05286*, 2022.

- [47] V. W. Anelli, G. M. Biancofiore, A. De Bellis, T. Di Noia, and E. Di Sciascio, “Interpretability of bert latent space through knowledge graphs,” in *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*, pp. 3806–3810, 2022.
- [48] D. Cer, M. Diab, E. Agirre, I. Lopez-Gazpio, and L. Specia, “Semeval-2017 task 1: Semantic textual similarity-multilingual and cross-lingual focused evaluation,” *arXiv preprint arXiv:1708.00055*, 2017.
- [49] D. Cer, M. Diab, E. Agirre, and I. n. Lopez-Gazpio, “A multi-perspective benchmark for evaluating semantic textual similarity.” <http://ixa2.si.ehu.eus/stswiki/index.php/STSbenchmark>, 2017.
- [50] M. Marelli, L. Bentivogli, M. Baroni, R. Bernardi, S. Menini, and R. Zamparelli, “Semeval-2014 task 1: Evaluation of compositional distributional semantic models on full sentences through semantic relatedness and textual entailment,” in *Proceedings of the 8th international workshop on semantic evaluation (SemEval 2014)*, pp. 1–8, 2014.
- [51] R. S. Witte and J. S. Witte, *Statistics*. John Wiley & Sons, 2017.
- [52] J. C. De Winter, S. D. Gosling, and J. Potter, “Comparing the pearson and spearman correlation coefficients across distributions and sample sizes: A tutorial using simulations and empirical data.,” *Psychological methods*, vol. 21, no. 3, p. 273, 2016.