Universidad de Buenos Aires
Facultad de Ciencias Exactas y Naturales

Análisis de similaridad entre representaciones de palabras generadas por modelos de audio y de texto

Tesis de Licenciatura en Ciencias de Datos

Cecilia Micaela Bolaños

Director: Pablo Brusco
Codirector: Leonardo Pepino
Buenos Aires, 2024

ANÁLISIS DE SIMILARIDAD ENTRE REPRESENTACIONES DE PALABRAS GENERADAS POR MODELOS DE AUDIO Y TEXTO

En el campo del aprendizaje automático, es muy común el uso de modelos pre-entrenados. Estos modelos, creados para resolver tareas generales y, generalmente entrenados sobre grandes cantidades de datos, se utilizan o adaptan posteriormente para tareas específicas. Un factor común de estos modelos es la capacidad de generar representaciones de los datos (*embeddings*). El poder expresivo de estas representaciones dependen en gran medida del diseño del modelo, del proceso de entrenamiento, y de los datos con los que fueron entrenados.

Una pregunta de interés en este campo es el estudio de la convergencia hacia *representaciones universales*, incluso en modelos entrenados en instancias provenientes de diferentes modalidades (audio, texto, imágenes, etc.). Entender qué información capturan estas representaciones y cómo se relacionan entre sí es esencial para aprovechar al máximo estos modelos.

En este trabajo, nos concentramos en el mundo de las **representaciones de palabras habladas**. Para ello, nos proponemos replicar y extender parte del trabajo realizado en Pasad et al. [2021]. En este artículo, los autores utilizan el *Análisis de Correlación Canónica* (CCA) para examinar similitudes y diferencias entre representaciones de palabras habladas provenientes de un modelo de habla (*Wav2Vec 2.0*), en contraste con representaciones generadas por un modelo de texto (*GloVe*).

Como extensión al trabajo proponemos la utilización de dos métricas adicionales: (a) *Linear Centered Kernel Alignment (Linear CKA)*, como alternativa a CCA; y (b) *ASIF*, como métrica complementaria con una visión más localizada de las representaciones, lo cual proporciona una mayor interpretabilidad. En cuanto a los modelos, proponemos la inclusión de modelos modernos que incluyen *BEATs* y *EncodecMAE* para audio, y *BERT* para texto.

Nuestros resultados indican que tanto Linear CKA como CCA son igualmente efectivos para evaluar la similitud entre representaciones, aunque Linear CKA es más eficiente computacionalmente. Además, mostramos cómo Wav2Vec 2.0 presenta una mayor similitud en las representaciones generadas con el modelo BERT que con las generadas con el modelo GloVe, alcanzando una similitud de 0.30 con BERT y 0.18 con GloVe bajo la métrica Linear CKA. Si bien se espera que Wav2Vec 2.0 aprenda del contexto del audio debido a su arquitectura, este hallazgo indicaría que además captura información contextual de las palabras y su significado, lo cual no es tan evidente. En este trabajo analizamos esta y otras hipótesis que pueden explicar la mayor cercanía entre estos espacios de representación. Por último, la comparación entre modelos de audio general (EncoderMAE) y de audio provenientes únicamente de habla (Wav2Vec 2.0) reveló diferencias en cómo los modelos de audio tienden a ser más fonéticos que semánticos.

Palabras claves: Modelos de Audio, Modelos de Texto, Similitud entre Representaciones, Representaciones de Palabras, Linear Centered Kernel Alignment, ASIF, Análisis de Correlación Canónica.

SIMILARITY ANALYSIS OF WORD REPRESENTATIONS FROM AUDIO AND TEXT MODELS

In the field of machine learning, the use of pre-trained models is widespread. These models, designed to solve general tasks, and generally trained on large amounts of data, are subsequently used or adapted for more specific tasks. A common pattern of these models is the ability to generate data representations (*embeddings*). The effectiveness of these representations depends on the model design, the training process, and the data used for training.

A question of interest in this field is the study of convergence towards *universal representations*, even in models trained on instances from different modalities (audio, text, images, etc.). Understanding what information these representations capture and how they relate to each other is essential to make the most of these models.

In this work, we focus on *spoken word representations*. To this end, we aim to replicate and extend part of the work carried out in Pasad et al. [2021]. In this article, the authors use *Canonical Correlation Analysis* (CCA) to examine similarities and differences between spoken word representations from a speech model (*Wav2Vec 2.0*) in contrast to representations generated by a text model (*GloVe*).

As an extension, we propose the use of two additional metrics: (a) *Linear Centered Kernel Alignment (Linear CKA)*, as an alternative to CCA; and (b) *ASIF*, as a complementary metric with a more localized view of the representations, providing greater interpretability. Regarding the models, we propose the inclusion of modern models such as *BEATs* and *EncodecMAE* for audio, and *BERT* for text.

Our results indicate that both Linear CKA and CCA are equally effective for evaluating the similarity between representations, although Linear CKA is more computationally efficient. Additionally, we show how Wav2Vec 2.0 presents a greater similarity in the representations generated with the BERT model than with those generated with the GloVe model, achieving a similarity of 0.30 with BERT and 0.18 with GloVe under the Linear CKA metric. While it is expected that Wav2Vec 2.0 learns from the context of the audio signal due to its architecture, this finding suggests that it also captures contextual information about the words and their meanings, which is not as evident. In this work, we analyze this and other hypotheses that may explain the closer relationship between these representation spaces. Lastly, the comparison between general audio models (EncodecMAE) and models derived solely from speech (Wav2Vec 2.0) revealed differences in how general audio models tend to be more phonetic than semantic.

Keywords: Audio Models, Text Models, Similarity Between Representations, Word Representations, Linear Centered Kernel Alignment, ASIF, Canonical Correlation Analysis.

AGRADECIMIENTOS

Quiero expresar mi más profundo agradecimiento a Leo, mi director. No podría haber soñado con un mejor director. Gracias por estar siempre para mí, en cualquier momento, explicando las cosas con paciencia, buen humor y entendiendo exactamente lo que necesitaba. Leo, gracias por soportar mi intensidad y por hacer estos cuatro meses de trabajo tan disfrutables. Tu apoyo fue clave para arrancar mi vida académica y me dio más ganas todavía de seguir en este camino. Valoro especialmente cómo compartiste tu pasión conmigo, porque cada charla que tuvimos fue clave para mantenerme motivada y con más ganas de aprender.

Pablo, mi otro director, realmente le diste ese toque especial al proyecto. Sin vos esta tesis no hubiera sido lo mismo. Gracias por recordarme la importancia de contar la historia, no solo de presentar resultados, y por estar siempre ahí para ver si necesitaba algo o cómo podías ayudarme. Pero mi agradecimiento va más allá de este trabajo. Gracias, Pablo, por ser el mejor profe del mundo, por la energía y dedicación que pones en cada clase y en cada uno de nosotros, tus alumnos. Me contagiaste las ganas de enseñar y me inspiras a seguir un camino como el tuyo.

Leo y Pablo, gracias por ser el mejor equipo que podría haber imaginado, en serio. Les quiero agradecer de corazón por haberme acompañado en todo este camino. Su generosidad, sus consejos y el estar ahí siempre significó un montón para mí. Los admiro no solo por lo cracks que son en lo que hacen, sino también como personas. Estoy eternamente agradecida por todo lo que me dieron.

A mi familia, les debo todo lo que soy. Mamá y papá, gracias por criarme con libertad, por confiar en mis decisiones y por estar ahí en cada paso, incluso yendo a buscarme a la parada cuando llegaba tarde a la noche. A mis hermanos, gracias por mostrarme con su ejemplo que todo es posible.

A mis amigos de la facultad, el grupo de *Intros*, ¿qué les puedo decir? Sin ustedes, la carrera no hubiera sido lo mismo. Las horas interminables de estudio, las clases compartidas, los almuerzos donde a veces hablábamos de la facu y otras veces no... Gracias por estar siempre ahí para motivarnos y apoyarnos, y por esa competencia amistosa entre nosotros que nos impulsa a mejorar y a querer siempre más.

Por último, quiero agradecer a la universidad pública, a la Facultad de Ciencias Exactas y Naturales, y a todos sus docentes. Gracias por todos estos años llenos de aprendizajes y momentos lindos. Asimismo, mi agradecimiento al Instituto de Cálculo, especialmente a Guille y Constanza, por cedernos el lugar para poder realizar nuestra tesis.

A mis viejos, por educarme con libertad y por creer en mí, acompañándome en cada decisión. Gracias por su apoyo incondicional.

Índice general

1. INTRODUCCIÓN

1.1. Motivación

En la actualidad, se desarrollan modelos de inteligencia artificial que son capaces de resolver no solo una tarea específica, sino un amplio espectro de estas. Esto se debe, en gran medida, al aprendizaje de **representaciones**: formas de abstraer patrones subyacentes a los datos que permiten una manipulación eficiente de su información. Por ejemplo, modelos de texto como *BERT* o *GPT* aprenden representaciones que les permiten realizar tareas específicas, como completar textos de manera coherente o traducir entre diferentes idiomas. La dependencia de estas tareas en las representaciones aprendidas genera una pregunta esencial: *¿cómo representan los datos estos modelos generales?*

Las representaciones de estos modelos son generalmente vectores de números llamados **embeddings** – vectores que encapsulan la información aprendida de los datos. A través de estas representaciones numéricas, los modelos pueden diferenciar y clasificar conceptos, aunque la forma exacta en que lo hacen puede variar. Los embeddings aprendidos dependen del entrenamiento del modelo y, por lo tanto, de los datos con los cuales se lo entrena, la arquitectura del modelo, el objetivo de entrenamiento, entre otros aspectos de diseño. Una pregunta interesante que surge es si distintos modelos tienden a converger hacia una representación común o "universal". A su vez, si esta convergencia sucede en modelos entrenados en distintas modalidades (audio, texto, imágenes, etc). Esto sugeriría que estas representaciones capturan aspectos fundamentales de la realidad, de manera análoga a las generadas por el cerebro humano.

Por ejemplo, en Huh et al. [2024] estudian si los modelos de texto e imágenes representan de manera similar conceptos equivalentes (imágenes junto a sus descripciones). Los autores proponen que, independientemente de la arquitectura, el objetivo de entrenamiento, o la modalidad, los modelos tienden a converger hacia una representación universal, la cual denominan "representación platónica". Esto indicaría que las representaciones generadas por diferentes modalidades están alineadas, sugiriendo que las modalidades son perspectivas de una única realidad subyacente y que los modelos son capaces de capturar ese espacio en común. La convergencia de representaciones facilitaría la transición de una modalidad a otra, además, datos de diferentes modalidades podrían contribuir al entrenamiento para obtener modelos más robustos.

Particularmente, en el área de procesamiento del audio, modelos como *BEATs* o *EncodecMAE* desarrollan representaciones que capturan características esenciales de los sonidos, las cuales sirven para reconocer aspectos tales como la melodía de una canción o la fuente de un sonido específico. Otros modelos de audio, especializados en la tarea de reconocimiento automático del habla, tales como *Wav2Vec 2.0*, aprenden representaciones específicas para la conversión de señales acústicas a texto estructurado. En esta dirección, el objetivo central de este trabajo es entender si las representaciones de palabras generadas por modelos entrenados a partir de texto y audio (específico de habla) convergen hacia representaciones comunes.

Para ello, replicaremos una serie de experimentos realizados en Pasad et al. [2021], en donde buscan medir la cercanía de distintos espacios de representación del habla a partir de métricas diseñadas para dicho objetivo. A su vez, extenderemos estos análisis

a modelos generales de audio que no fueron diseñados específicamente para la tarea del reconocimiento automático del habla, e incorporaremos otras técnicas para entender qué aspectos de las palabras están siendo capturados por los modelos de audio. En concreto, utilizamos representaciones de instancias replicadas en dos modalidades distintas – 40.000 palabras en forma escrita y su señal de acústica correspondiente – y desarrollamos experimentos que miden (1) si existe alguna correlación entre los espacios (2) si la cercanía entre conceptos en un espacio se mantiene en el otro.

1.2. Trabajos previos

La investigación de las representaciones que generan los modelos de inteligencia artificial constituye un campo de creciente interés, lo cual es clave para entender qué conceptos aprenden los modelos y cómo. Este análisis abarca diversas modalidades de entrada – generalmente imágenes, audio o texto –, y se enfoca en explorar distintos aspectos de la arquitectura de los modelos, tales como la influencia de las funciones de pérdida durante el entrenamiento.

Un enfoque en este campo es analizar cómo las decisiones de arquitectura y los objetivos de entrenamiento afectan las representaciones generadas. En Chung et al. [2021], los investigadores comparan diversas arquitecturas (redes recurrentes, redes convolucionales, *Transformer*) y direccionalidades (uni/bidireccional) con diferentes objetivos de entrenamiento (codificación predictiva contrastiva, codificación predictiva autorregresiva y codificación predictiva enmascarada). Para esto, emplean una técnica conocida como *Linear Centered Kernel Alignment (Linear CKA)* [Kornblith et al., 2019], un enfoque popular que se ha utilizado para comparar representaciones de las diferentes capas de una red neuronal [Matsoukas et al., 2022, Hao et al., 2023, Chen et al., 2023, Xie et al., 2022, Vulić et al., 2020]. Sus resultados muestran que la elección del objetivo de entrenamiento tiene el mayor impacto en las representaciones y que, bajo el mismo objetivo, la direccionalidad del modelo influye más que la propia arquitectura. Este tipo de análisis es crucial para entender las representaciones que estudiamos y sus diferencias evolutivas.

En línea con el análisis de las decisiones de arquitectura, también es relevante evaluar la utilidad de las representaciones generadas. En Ethayarajh [2019] muestran que las representaciones de palabras de modelos de texto como *ELMo* y *BERT* son altamente contextuales y varían significativamente dependiendo del contexto en el que se usa la palabra, especialmente en las capas superiores (las más cercanas a la salida de la red). En Choi et al. [2024] exploran representaciones de palabras generadas por modelos de habla, centrándose en un conjunto de datos diseñado específicamente con pares de palabras casi homófonas (fonéticamente similares) y sinónimos (semánticamente similares). Al medir las similitudes entre las representaciones con estos pares en diversos modelos, descubrieron que las similitudes fonéticas predominan sobre las semánticas. Estos análisis aportan una comprensión de las representaciones de palabras generadas por modelos de texto y habla.

Otro enfoque que se encuentra en la literatura consiste en analizar si modelos de diferentes modalidades representan conceptos similares, explorando la posibilidad de una representación universal. Estudios como Maniparambil et al. [2024] investigan el alineamiento entre modelos de visión y texto, en donde encuentran altas similitudes semánticas independientes al paradigma de entrenamiento. También proponen una métrica local basada en *CKA* para recuperar descripciones a partir de imágenes.

En nuestro estudio, nos proponemos investigar si los modelos de audio, tanto generales

como específicos de habla, representan a las palabras de manera similar a ciertos modelos de texto y si esto sugiere la existencia de una representación universal de palabras. El estudio que inspira nuestro trabajo es Pasad et al. [2021]. En él, los autores examinan cómo las representaciones en el modelo de habla *Wav2Vec 2.0* evolucionan a lo largo de sus distintas capas. Para ello utilizan diferentes configuraciones, y exploran dónde se localiza la información acústica y lingüística. Además, exploran si el modelo capta el significado de las palabras al comparar sus representaciones con las producidas por *GloVe* (un modelo basado en texto). Descubren que Wav2Vec 2.0 muestra una progresión parecida a la de un *autoencoder* (un tipo de red neuronal diseñada para comprimir información en pocas dimensiones), revelando cómo se codifican inicialmente características acústicas, seguidas por características fonéticas, luego la identidad de la palabra y finalmente su significado.

Nuestro trabajo ampliará el estudio de Pasad et al. [2021]. Para ello, replicaremos algunos de los experimentos utilizando los mismos datos, expandiremos las métricas utilizadas agregando *Linear CKA* y *ASIF* – una novedosa técnica propuesta en Norelli et al. [2023] –, y extenderemos el análisis a modelos de audio generales, lo que representa una novedad en la literatura. También actualizaremos los resultados a un modelo de texto más moderno, *BERT*.

Todos estos estudios fundamentan nuestro análisis, en el que buscamos analizar la evolución de las representaciones de palabras dentro de un mismo modelo y explorar la existencia de una representación universal. Para ello, medimos la correlación entre los espacios generados por distintos modelos y evaluamos su eficacia para recuperar representaciones de una modalidad utilizando otra. Este enfoque nos permite determinar no solo la similitud entre los modelos, sino también la utilidad de estos espacios representativos en tareas transmodales.

1.3. Estructura de la tesis

Considerando este capítulo introductorio, esta tesis se encuentra organizada en cinco capítulos que describiremos a continuación.

En el Capítulo 2, “Introducción teórica y métodos”, presentamos una definición del problema. Explicamos las diferentes formas de aprender las representaciones y mostramos cómo lo hacen los modelos que vamos a utilizar para el análisis. Además, introducimos las métricas que utilizaremos para comparar los espacios de representación.

En el Capítulo 3, “Metodología”, detallamos las decisiones clave adoptadas para realizar los experimentos. Esto incluye la selección de conjuntos de audios, los métodos para extraer sus representaciones y la implementación de las métricas.

En el Capítulo 4, “Experimentación”, exploramos, qué tanto se diferencian los distintos espacios de representación.

El Capítulo 5, “Conclusiones”, resume las mayores contribuciones de esta tesis, expone sus limitaciones y plantea posibles direcciones de trabajo futuro en las líneas de investigación propuestas en esta tesis.

2. INTRODUCCIÓN TEÓRICA Y MÉTODOS

2.1. Aprendizaje de representaciones

En este trabajo, analizaremos el desempeño de modelos entrenados para aprender representaciones, tanto de texto como de audio, a partir de datos. El objetivo principal de este aprendizaje es que dichas representaciones puedan ser reutilizadas en nuevas tareas. Las representaciones aprendidas encapsulan características generales que son transferibles a tareas específicas, un proceso que se conoce como *aprendizaje por transferencia* (transfer learning). La efectividad de este proceso depende de la capacidad de las representaciones para captar información relevante y descartar lo irrelevante. Lo que se considera relevante varía según la tarea, por lo que es ideal desarrollar representaciones mediante tareas similares a las que se pretenden realizar. Por ejemplo, si se busca transcribir habla, es mejor usar representaciones basadas en señales de voz y tareas que se centren en el contenido fonético, en lugar de usar señales de música o tareas que identifiquen al hablante.

Distinguimos dos enfoques generales para el aprendizaje de representaciones:

- **Enfoque supervisado:** Este enfoque entrena un modelo para resolver una tarea en la que se posee una gran cantidad de datos etiquetados, y luego se utilizan sus representaciones internas para resolver tareas de un dominio similar donde se poseen pocos datos etiquetados. Por ejemplo, si tenemos una gran base de datos de imágenes como ImageNet, podemos entrenar un modelo para clasificar las imágenes en diferentes clases. Este modelo aprenderá a reconocer características universales tales como bordes, texturas, patrones de colores y formas. Posteriormente, transfiere su conocimiento para tareas específicas usando bases de datos más chicas, tal como lo hacen en Heravi et al. [2017], donde utilizan modelos preentrenados para la clasificación automática de alimentos.
- **Enfoque no supervisado:** Este enfoque entrena un modelo para resolver una tarea sin necesidad de datos etiquetados. Un ejemplo de esto es un autoencoder, que se entrena para reconstruir la entrada a partir de una representación de menor dimensionalidad. Otro ejemplo son los modelos de lenguaje enmascarado, que buscan predecir las partes de la entrada que han sido enmascaradas [Devlin et al., 2019]. Este enfoque obliga al modelo a capturar las características más importantes de la entrada en las representaciones generadas. Posteriormente, estas representaciones pueden utilizarse como atributos para resolver otras tareas.

En esta tesis, nos centraremos en las representaciones generadas por modelos que emplean un enfoque no supervisado.

2.2. Modelos

En las siguientes secciones, analizaremos en detalle los modelos utilizados en nuestro estudio. Estos incluyen modelos de texto como GloVe y BERT, y modelos de audio como Wav2Vec 2.0, BEATs y EncodecMAE. Cada modelo se caracteriza por su tipo de entrada

específico, su arquitectura y sus objetivos y funciones de optimización particulares. Examinaremos estas diferencias, ya que las decisiones de diseño y entrenamiento influyen en las representaciones aprendidas.

La Tabla 2.1 presenta una comparación de las características clave de estos modelos, incluyendo el tipo de entrada, el objetivo de entrenamiento y la función de pérdida.

Tab. 2.1: Tabla comparativa de los diferentes modelos considerados para este trabajo diferenciando la entrada que considera, el objetivo de entrenamiento y la función de pérdida.

Modelo	**Entrada**	**Objetivo de Entrenamiento**	**Función de pérdida**
GloVe [Pennington et al., 2014]	Palabras	Global log-bilinear regression	Mínimos cuadrados ponderados
BERT [Devlin et al., 2019]	Subword tokens	Modelo de Lenguaje Enmascarado	Entropía cruzada
Wav2Vec 2.0 [Baevski et al., 2020]	Audio crudo	Modelo de Audio Enmascarado	Contrastiva
BEATs [Chen et al., 2022]	Parches de EspectogramaMel	Modelo de Audio Enmascarado (Teacher-student)	Entropía cruzada
EnCodecMAE [Pepino et al., 2024]	Audio crudo o ventanas de EspectogramaMel	Modelo de Audio Enmascarado	Entropía cruzada ponderada

2.2.1. Modelos de Texto: GloVe

GloVe [Pennington et al., 2014] (Global Vectors for Word Representation) es un método de aprendizaje no supervisado que tiene como objetivo aprender representaciones de palabras que capturen regularidades semánticas[1] y sintácticas[2]. Este modelo se basa en la utilización de estadísticas globales de coocurrencia dentro de un corpus.

En primer lugar, el algoritmo construye una matriz de coocurrencia de palabras X, donde cada elemento X_{ij} representa la cantidad de veces que la palabra j ocurre en el contexto de la palabra i – definiendo contexto como una ventana, típicamente simétrica, alrededor de cada palabra.

En segundo lugar, la contribución a los recuentos de coocurrencia se ve modificada por una función de ponderación $f(X_{ij})$ que otorga más importancia a las coocurrencias frecuentes mientras que disminuye la influencia de las que suceden en menor medida.

[1] Regularidad Semántica: Capacidad para capturar relaciones entre palabras basadas en su significado. Por ejemplo, en GloVe, la relación entre las capitales y sus respectivos países se puede capturar con la siguiente operación vectorial entre las representaciones: Madrid − España + Francia ≈ París.

[2] Regularidad Sintáctica: Capacidad de capturar relaciones gramaticales entre palabras. Por ejemplo, la relación entre diferentes formas de un verbo se puede capturar con la siguiente operación vectorial entre las representaciones: correr − corriendo + comiendo ≈ comer.

Tercero, se inicializan de manera aleatoria vectores para cada palabra y, mediante métodos de descenso de gradiente estocástico, se busca minimizar una función objetivo. Esta función es un modelo de mínimos cuadrados ponderados:

$$J = \sum_{i,j=1}^{V} f(X_{ij})(w_i^T \tilde{w}_j + b_i + \tilde{b}_j - \log(X_{ij}))^2$$

donde w_i y $\tilde{w}_j$ son los vectores de palabras y de contexto, respectivamente, b_i y $\tilde{b}_j$ son los términos de sesgo, y $f(X_{ij})$ es la función de ponderación.

Tras el entrenamiento, el modelo genera dos conjuntos de vectores: vectores de palabras (w) y vectores de contexto ($\tilde{w}$), los cuales son sumados para obtener la representación final de cada palabra. De esta manera, cada palabra tiene una representación vectorial consistente, que codifica detalles específicos del contexto en cada aparición individual de la palabra.

2.2.2. Modelos de Texto: BERT

BERT [Devlin et al., 2019] (Bidirectional Encoder Representations from Transformers), es un modelo de procesamiento de lenguaje natural que utiliza la arquitectura de Transformers [Vaswani et al., 2023] para preentrenar representaciones de lenguaje bidireccionales a partir de texto no etiquetado. A diferencia de GloVe, BERT produce vectores que codifican a palabras junto a su contexto. Por ejemplo, en la frase "el banco del parque" y "el banco financiero", la palabra "banco" tendrá diferentes representaciones en BERT debido a los contextos distintos, mientras que en GloVe, "banco" tendría una única representación independiente de su uso en la oración.

El entrenamiento de BERT consiste en modificar el codificador de un Transformer y conectarlo a cabezas especializadas para dos tareas principales que se realizan de manera simultanea: el Modelo de Lenguaje Enmascarado (MLM) y la Predicción de Siguiente Oración (NSP). En MLM, el 15 % de los tokens[3] en cada secuencia de entrada se enmascaran aleatoriamente con el objetivo es predecir el token original basado únicamente en su contexto. En NSP, el modelo aprende a predecir si una oración B es o no la que continúa a una oración A, lo cual ayuda al modelo a aprender la relación entre oraciones consecutivas. La función de pérdida de BERT es la suma de las pérdidas de las tareas MLM y NSP, lo que permite medir qué tan bien el modelo predice las palabras enmascaradas y si una oración sigue lógicamente a otra.

Finalmente, las cabezas adicionales son ignoradas y las representaciones de las distintas capas del codificador son utilizadas como la representación de una oración de entrada. Exploraremos más adelante en este trabajo cómo las distintas capas producen representaciones con distinto nivel de utilidad para distintas tareas.

2.2.3. Modelos de Audio: Wav2Vec 2.0

Wav2vec 2.0 [Baevski et al., 2020] es un modelo de reconocimiento del habla basado en la arquitectura de Transformers y entrenado mediante aprendizaje no supervisado. Este modelo está diseñado para ser preentrenado y posteriormente ajustado en tareas específicas, lo que lo hace especialmente útil para lenguajes con pocos recursos. Por ejemplo, se

[3] Un token refiere a la unidad mínima en la cual se divide la secuencia de entrada para su análisis.

lo aplicó en lenguas como el Quechua y el Guaraní [Romero et al., 2024], donde solo hay disponibles unos minutos u horas de habla transcrita.

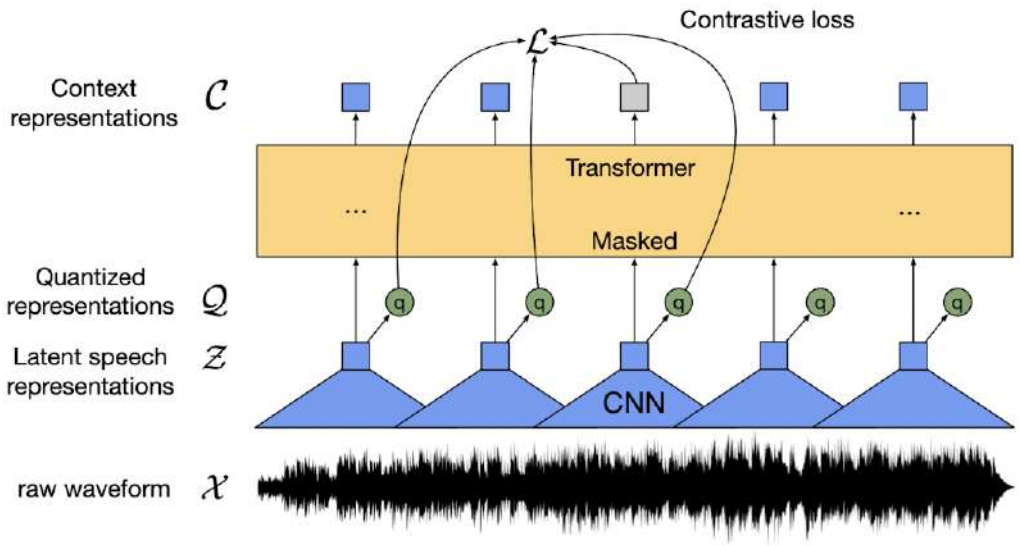

Fig. 2.1: Arquitectura del modelo Wav2vec 2.0. Fuente: Baevski et al. [2020].

La Figura 2.1 representa la arquitectura del modelo Wav2vec 2.0. A continuación, desglosamos los componentes y procesos principales mostrados en la figura:

- **Señal de Audio Cruda (χ)**: La señal de audio de entrada, denotada como χ, se representa con una frecuencia de muestreo de 16kHz, comúnmente utilizada en tareas de reconocimiento de habla.
- **Representaciones Latentes del Habla (Z)**: La señal cruda pasa a través de una red neuronal convolucional (CNN), la cual extrae representaciones latentes. Dado que se aplica una convolución con un cierto *campo receptivo* y un *stride* definido, estas representaciones latentes corresponden a ventanas de tiempo específicas del audio original. Después de aplicar todos los strides, la tasa de muestreo original de 16 kHz se reduce a 50 Hz. Esta reducción significa que, en lugar de tener 160,000 muestras para 10 segundos de audio, terminamos con solo 500 muestras (50 muestras/segundo * 10 segundos). Estas secuencias de datos reducidas son mucho más manejables para arquitecturas de tipo Transformer.
- **Módulo de Cuantización (Q)**: El módulo de cuantización convierte las salidas vectoriales continuas en representaciones discretas. Este proceso de cuantización se realiza mediante la *cuantización por producto*[4]. Para este modelo, los autores uti-

[4] La cuantización por producto es una técnica que divide un espacio de alta dimensión en varios subespacios de menor dimensión, permitiendo manejar datos de manera más eficiente. Utiliza "libros de códigos" – conjuntos de posibles valores cuantizados para cada subespacio. Para un vector dado, se elige una representación cuantizada de cada subespacio utilizando los libros de códigos correspondientes y luego se concatenan estas representaciones para formar una representación final del vector. Por ejemplo, si un vector de dimensión 100 se divide en 5 subespacios de dimensión 20, se selecciona un código de cada uno de los 5 libros de códigos y se concatenan para formar un nuevo vector de dimensión 100.

lizaron este módulo ya que encontraron que cuantizar las representaciones objetivo daba mejores resultados que usar la señal continua. Una posible razón que nombran, es que, al convertir las representaciones continuas en discretas, el modelo aprende unidades del habla que son más robustas y representativas de los patrones fonéticos en el audio.

- **Representaciones de Contexto (C)**: Las representaciones latentes se procesan mediante una serie de capas de Transformers. Las capas de Transformers generan las representaciones de contexto, denotadas como C, que capturan la información contextual de los segmentos de audio.

Durante el preentrenamiento, el modelo Wav2vec 2.0 enmascara ciertas porciones de los pasos de tiempo en las representaciones latentes del habla Z. Estas porciones corresponden a segmentos específicos de la secuencia de audio que se seleccionan y ocultan. El objetivo es que el modelo aprenda a predecir la representación latente cuantizada correcta (q_t) para un paso de tiempo enmascarado dentro de un conjunto de distractores ($\tilde{Q}_t$). Este conjunto incluye la representación correcta y K representaciones distractoras.

La función de pérdida a minimizar es la suma de la pérdida contrastiva (L_m) y la pérdida de diversidad (L_d) en $L = L_m + \alpha L_d$.

- **Función de pérdida Contrastiva (L_m)**: La pérdida contrastiva es el rendimiento en la tarea no supervisada y se define como:

$$L_m = -\log \frac{\exp(\text{sim}(c_t, q_t)/\kappa)}{\sum_{\tilde{q}\in\tilde{Q}_t} \exp(\text{sim}(c_t, \tilde{q})/\kappa)}$$

  donde c_t es la salida de las capas de Transformers centrada en el paso de tiempo enmascarado t, y $\text{sim}(a, b)$ es la similitud coseno entre a y b:

$$\text{sim}(a, b) = \frac{a^T b}{\|a\|\|b\|}$$

- **Función de pérdida de Diversidad (L_d)**: La pérdida de diversidad está diseñada para aumentar el uso de todas las representaciones del libro de códigos cuantizado, y no solo de un subconjunto selecto.

Originalmente, el modelo Wav2Vec 2.0 fue desarrollado para abordar el reconocimiento automático de habla. Sin embargo, estudios recientes demostraron su aplicabilidad para resolver otras tareas de habla, como verificación del hablante [Fan et al., 2021], traducción del habla [Wang et al., 2021] y reconocimiento de emociones [Pepino et al., 2021].

2.2.4. Modelos de Audio: BEATs

BEATs [Chen et al., 2022] (Bidirectional Encoder representation from Audio Transformers) representa un enfoque para el preentrenamiento de audio, combinando un tokenizador acústico con un modelo de aprendizaje no supervisado de audio (SSL). A diferencia de *Wav2Vec 2.0*, que se centra específicamente en aprender representaciones para el reconocimiento del habla, *BEATs* busca aprender representaciones generales de sonido. Esta distinción se refleja en la elección del conjunto de datos para el preentrenamiento: mientras que *Wav2Vec 2.0* utiliza audiolibros, *BEATs* emplea un dataset que incluye clips diseñados para identificar una variedad de sonidos.

El modelo BEATs es preentrenado utilizando un marco iterativo, donde tanto el tokenizador acústico como el modelo SSL son optimizados de manera conjunta. A continuación describimos los detalles de este proceso:

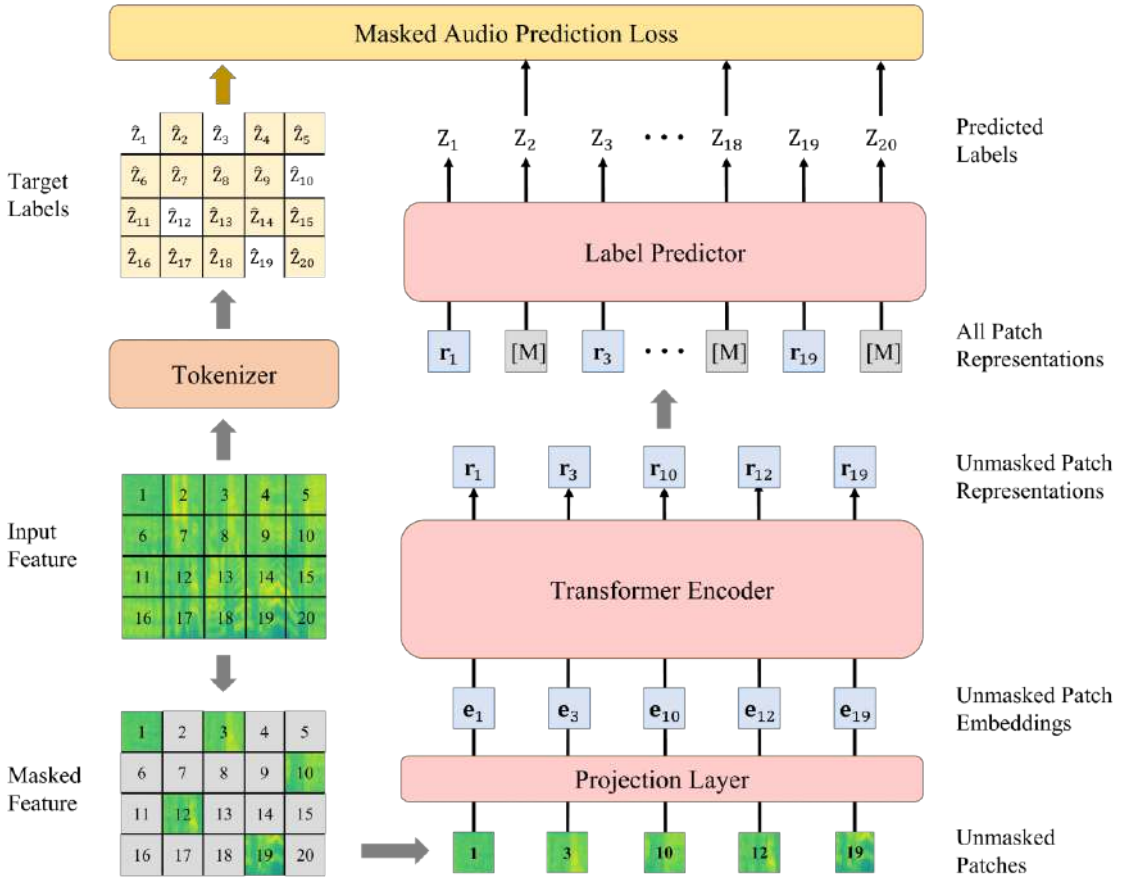

Fig. 2.2: Proceso de entrenamiento del modelo de audio SSL. Fuente: Chen et al. [2022]

1. **Proceso iterativo**:

   - **Tokenizador Acústico**: La primera iteración utiliza un tokenizador de proyección aleatoria que genera etiquetas discretas (parches) a partir de características acústicas continuas[5]. Este proyecta cada parche a través de una capa lineal y

[5] Dada la señal de audio crudo, se calculan las características del banco de filtros Mel mediante varios pasos. Primero, la señal se divide en ventanas de 25 ms con un solapamiento de 10 ms. Para cada ventana, se aplica una Transformada de Fourier de Tiempo Corto (STFT) para calcular el espectro de frecuencia. Luego, se obtiene el espectro de potencia a partir de la magnitud de la STFT. Posteriormente, se aplica un banco de filtros Mel de 128 dimensiones al espectro de potencia. Estos filtros triangulares están diseñados en una escala Mel, que imita la percepción no lineal del oído humano, siendo más discriminativa en frecuencias bajas [Fayek, 2016]. La representación resultante tiene tantas filas como ventanas (proporcional a la duración del audio) y 128 columnas correspondientes a las bandas de frecuencia Mel utilizadas. Estas características acústicas son luego divididas en parches regulares de 16×16. Es decir, el primer parche corresponde a las primeras 16 frecuencias del banco de filtros Mel de las primeras 16 ventanas temporales. Tanto el tokenizador como el modelo SSL reciben estos parches como entrada.

asigna etiquetas basadas en el vecino más cercano de un conjunto de vectores inicializados aleatoriamente.

Las siguientes iteraciones utilizan un Tokenizador Auto-Destilado que busca predecir las salidas del modelo SSL de la iteración anterior. El proceso de destilación de conocimiento permite que el modelo SSL de audio de la iteración anterior actúe como maestro, guiando al nuevo tokenizador para refinar su generación de etiquetas.

- **Modelo SSL de Audio**: Como modelo SSL de audio utilizan la arquitectura de Vision Transformer (ViT) [Dosovitskiy et al., 2021] que consiste en una capa lineal para aplanar los parches y luego capas de Transformers. Este modelo, es entrenado mediante una tarea de Modelado de Audio Enmascarado. En esta tarea, el 75 % los parches de la secuencia de entrada de características acústicas son enmascarados aleatoriamente, y el modelo debe predecir las etiquetas discretas correspondientes en las regiones enmascaradas. La función de pérdida utilizada durante el preentrenamiento es la entropía cruzada, que mide la discrepancia entre las etiquetas acústicas discretas generadas por el tokenizador anterior y las predichas por el modelo SSL de audio. La Figura 2.2 ilustra el proceso de entrenamiento del modelo SSL.

Este modelo fue evaluado y mostró buen desempeño en tareas que corresponden a audio, como la identificación de sonidos, y en tareas que corresponden a habla, como la clasificación de sentimientos y el reconocimiento de comandos. En esta última, se busca determinar a cuál comando de un conjunto finito pertenece el audio.

2.2.5. Modelos de Audio: EnCodecMAE

EnCodecMAE [Pepino et al., 2024] es un modelo diseñado para el aprendizaje de representaciones universales de audio. A diferencia de BEATs, que se centra en el uso de clips de sonidos para su entrenamiento, y de *Wav2Vec 2.0*, que utiliza audiolibros, *EnCodecMAE* combina ambos enfoques y adicionalmente incorpora música de distintos géneros, capturando una gama más amplia de sonidos. Además, otra diferencia crucial con *BEATs* es que *EnCodecMAE* no utiliza parches; en su lugar, divide la señal en ventanas, permitiendo un procesamiento más continuo y directo de la señal de audio.

Este modelo utiliza EnCodec [Défossez et al., 2022], un códec de audio neuronal, para generar objetivos discretos y emplea una arquitectura de Autoencoder Enmascarado (MAE) [He et al., 2021] como mecanismo de aprendizaje autosupervisado. El proceso de preentrenamiento consiste en codificar señales de audio, enmascarar partes de las representaciones codificadas y entrenar el modelo para reconstruir las partes enmascaradas.

La Figura 2.3 muestra la arquitectura del modelo. Esta detalla el procedimiento completo que se sigue para aprender las representaciones. La parte izquierda de la figura representa el proceso de obtención de targets discretos:

1. **Encodec Encoder:** La señal de audio es introducida en el codificador de EnCodec. El codificador, compuesto por capas convolucionales y LSTM [Hochreiter and Schmidhuber, 1997], procesa la señal de audio para reducir su tasa de muestreo de 24 kHz a 75 Hz. La salida es una secuencia de vectores de 128 dimensiones, cada uno correspondiente a una ventana de la señal de audio.

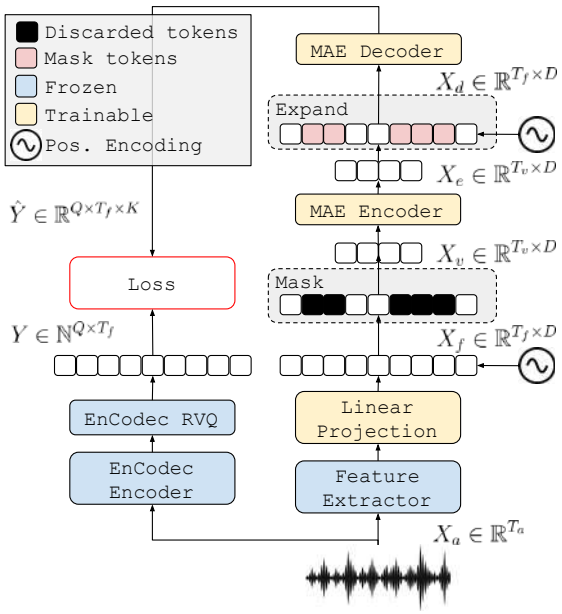

Fig. 2.3: Proceso de entrenamiento de EncodecMAE. Fuente: Pepino et al. [2024]

2. **Encodec RVQ:** Los vectores de 128 dimensiones son procesados por el bloque de Residual Vector Quantization (RVQ) en donde se mapea la entrada a valores discretos usando múltiples libros de códigos (32 en total, cada uno con 1024 valores).

La parte derecha de la figura ilustra el proceso de reconstrucción:

1. **Entrada de la Señal de Audio:** La señal de audio, representada como $X_a \in \mathbb{R}^{T_a}$, es introducida en el modelo.

2. **Feature Extractor:** La señal de audio es procesada obteniendo las características para cada ventana. Existen diferentes posibilidades de extraer estas características, por ejemplo se puede usar EspectrogramaMel o, también, las representaciones generadas por el Encodec Encoder.

3. **Linear Projection:** Esta parte proyecta linealmente las características a una secuencia de vectores de dimensiones más altas, resultando en $X_f \in \mathbb{R}^{T_f \times D}$. Luego, a la secuencia X_f se le añaden embeddings posicionales sinusoidales para mantener la información del orden temporal.

4. **Enmascaramiento:** Una proporción de los vectores en X_f son enmascarados, produciendo una secuencia enmascarada $X_v \in \mathbb{R}^{T_v \times D}$.

5. **MAE Encoder:** La secuencia X_v sin los segmentos enmascarados, sirve de entrada para el codificador MAE, que genera una representación codificada $X_e \in \mathbb{R}^{T_v \times D}$. Esta es expandida para incluir tokens de máscara en las posiciones originales enmascaradas, resultando en $X_d \in \mathbb{R}^{T_f \times D}$. Este codificador está compuesto por capas de Transformers.

6. **MAE Decoder:** La secuencia expandida X_d es introducida en el decodificador MAE. Este genera una secuencia de posteriors $\hat{Y}$, que son las probabilidades de los valores discretos en las posiciones enmascaradas.

Una vez obtenidas las representaciones de ambas partes (derecha e izquierda), se computa la pérdida. En este caso, utiliza como función de pérdida la entropía cruzada ponderada que es calculada entre las posteriors $\hat{Y}$ generadas por el MAE Decoder y los objetivos discretos Y producidos por EnCodec.

2.3. Técnicas de análisis

Gran parte del trabajo consiste en determinar si espacios generados por diferentes modelos, o por el mismo modelo en diferentes capas, comparten similitudes bajo alguna métrica específica. Para ello, contamos con n observaciones para las cuales disponemos de representaciones generadas en un espacio vectorial que codifican n palabras escritas (o sus respectivas señales acústicas).

Definimos entonces $X \in \mathbb{R}^{n \times d_1}$ como la matriz de representaciones para n ejemplos compuesta por n vectores $x_i \in \mathbb{R}^{d_1}$. De manera análoga, definimos $Y \in \mathbb{R}^{n \times d_2}$ como la matriz de representaciones para los mismos n ejemplos, ahora los cuales pertenecen a $\mathbb{R}^{d_2}$. Nos interesa estudiar entonces **medidas de similaridad** $m(X, Y)$ que permitan comparar estos espacios.

2.3.1. Análisis de Correlación Canónica

El Análisis de Correlación Canónica (CCA) [Hotelling, 1936] es una técnica estadística que explora la relación entre dos conjuntos de datos multidimensionales, representados por las matrices X e Y. El CCA busca identificar combinaciones lineales óptimas de las variables de cada conjunto para maximizar la correlación entre ellos. Encontrar una alta correlación luego de aplicar transformaciones lineales a los espacios de representaciones indica que existen patrones o estructuras comunes en las direcciones de máxima correlación.

El método funciona de la siguiente manera. Dadas X e Y, se calcula en primer lugar direcciones de proyección u_1 para X y v_1 para Y, que maximizan la correlación de las matrices proyectadas. Es decir:

$$\rho(u_1, v_1) = \max_{u_1, v_1} \operatorname{corr}(u_1^T X, v_1^T Y).$$

Posteriormente, el proceso es repetido de manera iterativa para obtener direcciones u_j, v_j para $j \in [2, k]$ siendo $k = min(d_1, d_2)$ bajo la condición de ortogonalidad entre las proyecciones para garantizar que cada par de variables canónicas aporte información independiente y no redundante. La similitud entre las representaciones de X e Y, es entonces, el promedio de las correlaciones de todos los pares de variables canónicas:

$$\bar{\rho} = \frac{1}{k} \sum_{i=j}^{k} \rho(u_j, v_j).$$

Un valor de $\bar{\rho}$ cercano a 1 indica una fuerte relación lineal entre las representaciones, mientras que un valor cercano a 0 sugiere que no tienen una relación lineal fuerte.

2.3.2. Centered Kernel Alignment

El **Centered Kernel Alignment** [Kornblith et al., 2019] (CKA) es una técnica estadística utilizada para medir qué tan similares son dos conjuntos de datos, denotados

como X e Y. Antes de aplicar CKA, es importante que estas matrices sean centradas; esto implica restar a cada elemento de una columna la media de esa columna.

La técnica utiliza el **Criterio de Independencia de Hilbert-Schmidt** [Gretton et al., 2005] (HSIC) para evaluar la dependencia o independencia entre los conjuntos de datos. Este criterio se beneficia de la capacidad de los *métodos de kernel* para transformar los datos a otros espacios, permitiendo capturar relaciones no lineales complejas. La selección de diferentes funciones de kernel permite personalizar la medida de similitud según las características específicas de los datos o las necesidades del problema a resolver, permitiendo capturar dependencias que pueden ser lineales, no lineales o incluso polinomiales.

Un ejemplo de función de kernel es el *kernel lineal*, que se basa en el producto punto entre vectores de características. Al seleccionar este kernel para CKA, estamos midiendo cuánta de la información contenida en un conjunto de datos puede ser explicada linealmente por el otro. Este proceso busca identificar correlaciones lineales entre las representaciones de los datos. Cuando utilizamos el kernel lineal, la fórmula de CKA se expresa como:

$$LinearCKA(X, Y) = \frac{\|Y^T X\|_F^2}{(\|X^T X\|_F \|Y^T Y\|_F)}$$

Esta fórmula nos indica hasta qué punto las estructuras de las representaciones internas de los dos conjuntos de datos son similares. Un valor más alto de CKA implica que la estructura de las correlaciones de un conjunto se refleja de manera similar en el otro, independientemente de transformaciones lineales como el escalamiento isotrópico (escalado uniforme en todas las direcciones) o las transformaciones ortogonales.

2.3.3. ASIF

Hasta este momento, presentamos dos métricas, CCA y CKA, que indican similitud entre espacios de representación. Sin embargo, estas métricas no proporcionan una interpretación detallada sobre qué palabras específicas están siendo representadas de manera similar o diferente por los dos modelos. Además, si obtenemos valores como 0.5 o 0.7, nos gustaría entender cuál es la diferencia entre los espacios de representación. Para superar esta limitación, incorporamos ASIF, una técnica novedosa propuesta en Norelli et al. [2023], que ofrece una interpretación más detallada.

ASIF se basa en la premisa de que las representaciones relativas inducen un espacio común significativo entre dos modalidades distintas. Suponiendo que tenemos dos modelos buenos, donde cada uno codifica adecuadamente la información, esperamos que un nuevo punto cercano a ciertas observaciones en un espacio mantenga su proximidad en el segundo espacio.

Partiendo de las mismas matrices X e Y, donde X corresponde a las representaciones generadas por un modelo de audio e Y a las generadas por un modelo de texto, podemos considerar esta técnica como una medida de similitud. Específicamente, evaluamos si, para un audio x, se puede obtener su transcripción y. La idea es que, si los espacios de representación fueran similares, un audio x que está cercano a ciertos audios, x_1, x_2, y x_3, entonces, la transcripción del audio, y, debería estar cercana a las correspondientes transcripciones, y_1, y_2, e y_3. Si esto sucediera para un alto porcentaje de pares (x, y) nos diría que los espacios son similares. En este sentido, ASIF propone el procedimiento para asignar la mejor transcripción dentro del conjunto de transcripciones originales para un audio x^*:

1. Partimos de las matrices X e Y.

2. Calculamos la representación relativa n-dimensional para cada transcripción en Y, $\hat{y}_i = [\text{sim}(\hat{y}_i, y_1), \ldots, \text{sim}(\hat{y}_i, y_n)]$, donde sim es una función de similitud, por ejemplo, similitud coseno. Luego para cada $\hat{y}_i$ asignamos cero a todos los valores excepto los k más altos, y elevamos a $p \geq 1$. Finalmente, normalizamos y almacenamos los n vectores procesados $\hat{y}_i$. Los valores de k y p son hiperparámetros con valores 800 y 8 respectivamente en el trabajo original.

3. Calculamos la representación relativa de x^* en X y repetimos el mismo procesamiento con el k y p elegidos.

4. Consideramos la representación relativa de la nueva grabación x^* como si fuera la representación relativa de su transcripción ideal y^*, es decir, definimos $y^* := x^*$. De esta manera, elegimos la transcripción candidata $\hat{y}_i$ más similar al ideal, con $i = \arg\max_i(\text{sim}(y^*, \hat{y}_i))$.

Para generar una transcripción para otra grabación repetimos desde el paso 3.

2.3.4. Comparación entre las métricas

En este trabajo, abordaremos el análisis utilizando tres métricas distintas, cada una con propiedades diferentes que pueden influir de manera significativa en los resultados y la interpretación de nuestro estudio. La Tabla 2.2 proporciona un resumen comparativo inicial de estas métricas. A continuación, profundizaremos en las diferencias específicas entre ellas para comprender las ventajas y limitaciones de cada métrica en el contexto de la convergencia representacional entre diferentes modalidades de datos.

Tab. 2.2: Resumen de Métricas

Métrica	Tipo	Invariante	Complejidad Computacional
CCA [Hotelling, 1936]	Global	Transformaciones lineales inversibles, escalamiento isotrópico, transformaciones ortogonales	$O(n \cdot d_1 \cdot d_2 + d_1^3 + d_2^3 + \min(d_1^2 \cdot d_2, d_1 \cdot d_2^2))$
Linear CKA [Kornblith et al., 2019]	Global	Escalamiento isotrópico, transformaciones ortogonales	$O(n(d_1 + d_2 + d_1 d_2 + d_1^2 + d_2^2))$
ASIF [Norelli et al., 2023]	Local	Escalamiento isotrópico, transformaciones ortogonales	$O(n^2 \log n)$

CCA mide la relación lineal entre dos conjuntos de variables, encontrando las combinaciones lineales de estas que tienen la mayor correlación posible. CCA es invariante a cualquier transformación lineal invertible, escalamiento isotrópico y transformaciones

ortogonales. Esta primera propiedad no es deseada para un índice de similitud, ya que la invarianza a la transformación lineal invertible implica que la escala de las direcciones en el espacio de activación es irrelevante. Sin embargo, empíricamente, la información de escala es consistente a través de las redes y útil en diversas tareas [Kornblith et al., 2019].

En cuanto a la complejidad computacional de CCA, con X e Y matrices de entrada con dimensiones $n \times d_1$ y $n \times d_2$, se puede descomponer en la suma de varios cálculos significativos. La complejidad general puede expresarse como:

$$O(n \cdot d_1 \cdot d_2 + d_1^3 + d_2^3 + \text{mín}(d_1^2 \cdot d_2, d_1 \cdot d_2^2))$$

donde el primer término $O(n \cdot d_1 \cdot d_2)$ proviene del cálculo de las matrices de covarianza C_{XY} y C_{YX}, y los términos $d_1^3 + d_2^3$ y $\text{mín}(d_1^2 \cdot d_2, d_1 \cdot d_2^2)$ provienen de la inversión de las matrices de covarianza y la descomposición en valores singulares (SVD) del producto de matrices normalizadas, respectivamente.

Como alternativa, surge CKA, que en su versión lineal mide la similitud entre las representaciones de X e Y al evaluar qué tan alineadas están sus dimensiones en términos de correlaciones lineales, independientemente del escalado de los datos. A diferencia de CCA, CKA no es invariante a cualquier transformación lineal invertible, aunque sí lo es a escalamiento isotrópico y transformaciones ortogonales. CKA se convirtió en un enfoque popular, siendo ampliamente utilizado para comparar las representaciones de diferentes capas de una red, de redes arquitectónicamente similares entrenadas de manera diferente, o de modelos con diferentes arquitecturas entrenados con los mismos datos. Sin embargo, también presenta desventajas, como su sensibilidad a valores atípicos [Davari et al., 2023]. Por ejemplo, dos espacios que son iguales excepto por un punto que difiere significativamente harán que el valor de Linear CKA disminuya considerablemente, a pesar de que en nuestra intuición los espacios son casi idénticos.

En cuanto a su costo computacional, es una alternativa más eficiente en comparación con CCA, presentando la siguiente complejidad:

$$O(nd_1 + nd_2 + nd_1d_2 + nd_1^2 + nd_2^2) = O(n \cdot (d_1 + d_2 + d_1d_2 + d_1^2 + d_2^2))$$

donde los primeros dos términos provienen del centrado de cada una de las matrices y los últimos tres términos de las multiplicaciones entre matrices (X^TY, X^TX y Y^TY respectivamente).

Tanto CCA como CKA son métricas globales que evalúan la similitud de los espacios de representación considerando su totalidad, proporcionando un valor único que, a menudo, no es fácilmente interpretable. Por esta razón, incorporamos ASIF. ASIF preserva las mismas invariancias que CKA, ya que si se realiza una rotación en el espacio, las distancias se mantienen, y si se aplica un escalamiento isotrópico, los k valores más altos seguirán siendo los mismos. Sin embargo, a diferencia de CKA, se distingue como una métrica local. En lugar de abarcar el espacio completo de representaciones, ASIF considera exclusivamente las k representaciones de audios que más se asemejan en términos relativos.

Este enfoque permite una interpretación más directa y contextual de las similitudes. Específicamente, al analizar un nuevo audio, calculamos primero su representación en el espacio de representaciones del conjunto de datos, resultando en un vector de n dimensiones, donde n representa el número total de observaciones. De este vector, seleccionamos únicamente las k componentes más altas, las cuales corresponden a las direcciones que muestran la mayor similitud con el audio en cuestión. Esencialmente, estamos considerando los k audios más parecidos al nuevo audio.

Para asignar una transcripción a este audio, calculamos la similitud coseno entre la representación del nuevo audio y las representaciones relativas de las transcripciones existentes. Es importante destacar que solo las k dimensiones seleccionadas son relevantes, ya que todas las otras dimensiones son cero, y por lo tanto, no contribuyen a la similitud coseno. Este método puede llevar a dos escenarios principales: que la transcripción que maximiza la similitud sea exactamente la de uno de los k audios más parecidos, o que no coincida directamente con estas, pero comparta características significativas en las direcciones que definen las k componentes más relevantes.

Además, en términos de complejidad del algoritmo ASIF se puede aproximar a:

$$O(n^2) + O(n^2 \log n) + O(n \log n) + O(n \cdot k) \approx O(n^2 \log n)$$

El primer término corresponde al cálculo de la representación relativa, el segundo término corresponde a ordenar cada una de las representaciones relativas consideradas para las n observaciones, y los últimos dos términos corresponden al cálculo de representación relativa del nuevo audio y a obtener la transcripción que maximice la similitud.

3. METODOLOGÍA

3.1. Dataset

3.1.1. LibriSpeech

LibriSpeech [Panayotov et al., 2015] es un corpus de grabaciones de audiolibros de dominio público en inglés. Fue diseñado para el entrenamiento y la evaluación de modelos de reconocimiento automático de voz (ASR, por sus siglas en inglés). Se encuentra disponible para descargarlo en el sitio OpenSLR (`http://www.openslr.org/12/`).

Contiene aproximadamente 1000 horas de grabaciones de audiolibros en inglés, leídos por una variedad de hablantes. Las grabaciones están en formato FLAC con una frecuencia de muestreo de 16 kHz y cada archivo de audio está acompañado por su correspondiente transcripción alineada a nivel de oración.

Para facilitar su uso, LibriSpeech está dividido en varios subconjuntos que incluyen datos de entrenamiento, desarrollo y prueba, cada uno categorizado por la calidad de las grabaciones:

- **Entrenamiento**: Incluye 100 horas (`train-clean-100`), 360 horas (`train-clean-360`) y 500 horas (`train-other-500`) de audio.
- **Desarrollo**: Conjuntos `dev-clean` (alta calidad) y `dev-other` (calidad diversa).
- **Prueba**: Conjuntos `test-clean` (alta calidad) y `test-other` (calidad diversa).

Para este trabajo, utilizamos exclusivamente el subconjunto `dev-clean` de LibriSpeech. Este subconjunto incluye aproximadamente 5.4 horas de audio de alta calidad, con un total de 2,703 oraciones leídas por 40 hablantes (20 hombres y 20 mujeres). Estas oraciones suman un total de 54,402 palabras no únicas, de las cuales 28,075 fueron pronunciadas por mujeres y 26,327 por hombres.

3.1.2. Alineamientos

El concepto de *alineamiento* refiere a la correspondencia temporal precisa entre el audio y las transcripciones. Este proceso es fundamental ya que permite identificar con exactitud dónde se encuentra cada palabra en la señal de audio.

Los alineamientos de LibriSpeech fueron calculados por Lugosch et al. [2019] usando el Montreal Forced Aligner [McAuliffe et al., 2017], un sistema de alineamiento automático de texto y habla de código abierto. Estos alineamientos están disponibles en el sitio `https://zenodo.org/records/2619474`.

Para cada grabación de audio, proporciona un archivo `.TextGrid`, formato utilizado por el software de análisis de habla *Praat*, que contiene información detallada sobre los intervalos de tiempo para palabras y fonemas. Cada `.TextGrid` incluye:

- **IntervalTier para palabras**: Indica el tiempo de inicio y fin, en segundos, de cada palabra en el audio.
- **IntervalTier para fonemas**: Proporciona detalles sobre el tiempo de inicio y fin, en segundos, de cada fonema.

3.2. Extracción de representaciones

Considerando los audios del conjunto `dev-clean`, nuestro objetivo es extraer las representaciones generadas para cada palabra dentro de cada archivo de audio. Estas son calculadas para los modelos descritos en el Capítulo 2, abarcando cada una de las capas de sus respectivas arquitecturas.

A continuación, describimos el proceso llevado a cabo para obtener las representaciones. En total, logramos calcular 46,906 representaciones correspondientes a palabras no únicas, de las cuales 24,310 fueron pronunciadas por mujeres y 22,596 por hombres.

Modelos de Texto

Para obtener las representaciones de los modelos que reciben como entrada texto, seleccionamos la transcripción de cada audio.

Para GloVe, descargamos el modelo preentrenado `glove.42B.300d`[1], donde las representaciones son vectores de 300 dimensiones. Dado que su arquitectura genera una única representación para cada palabra, consideramos cada palabra de la transcripción y la procesamos con el modelo preentrenado para obtener su representación correspondiente.

Por otro lado, para BERT[2] utilizamos la configuración BASE, que está compuesta por 12 capas de bloques Transformer, cuyas representaciones tienen un tamaño de 768. Procesamos la transcripción completa de cada audio, a través del modelo, obteniendo la representación de cada *token* en cada una de las capas. Finalmente, calculamos la representación de una palabra, en una capa específica, como el promedio de las representaciones de los *tokens* que la componen en esa capa. Es importante recordar que, en BERT, una palabra suele dividirse en varias subpalabras para su procesamiento, y que la representación de una palabra depende del contexto que genera la trascripción.

Modelos de Audio

Para los modelos que reciben audio como entrada, primero procesamos el audio completo con el modelo para obtener su representación. Posteriormente, identificamos en qué ventanas de esta representación aparece cada palabra, utilizando los alineamientos y la información sobre la frecuencia de las ventanas proporcionada por el modelo. Una vez que identificadas las ventanas correspondientes a cada palabra, extraemos las representaciones intermedias generadas para cada una.

En el caso de Wav2Vec 2.0[3], en su configuración BASE, que cuenta con 12 bloques de Transformer con una dimensión de 768, la representación de una palabra en una capa específica la calculamos como el promedio de las representaciones de las ventanas correspondientes a esa palabra.

Para EncodecMAE[4], consideramos la configuración BASE donde el *MAE Encoder* cuenta con 10 bloques de Transformers, también con una dimensión de 768 y realizamos el mismo procedimiento, obteniendo las representaciones promediadas de las ventanas en

[1] Descargado de https://nlp.stanford.edu/projects/glove/.

[2] Descargado de https://huggingface.co/google-bert/bert-base-uncased.

[3] Representaciones obtenidas a través de https://github.com/s3prl/s3prl que utiliza el código oficial del modelo.

[4] Representaciones obtenidas a través de https://github.com/mrpep/easyaudio que utiliza el código oficial del modelo.

cada una de las capas del *Transformer* para cada palabra. En esta configuración BASE, consideramos dos métodos de extracción para el *Feature Encoder*: el primero utiliza EspectrogramaMel y el segundo directamente el Encodec.

En el caso de BEATs[5]. Procesamos parches de EspectrogramaMel de 16x16, seguidos por 12 bloques de Transformers con una dimensión de 768. Dado que cada parche contiene información de 16 ventanas contiguas, la representación de una palabra la obtenemos promediando los parches que contienen las ventanas correspondientes a esa palabra. Esta metodología implica que la representación de una palabra es imprecisa, ya que incluye información de ventanas adicionales a las que realmente pertenecen a la palabra. Por ejemplo, si una palabra comienza en la ventana 14 y termina en la 25, su representación tendrá en cuenta los parches generados desde la ventana 1 hasta la 32.

3.3. Implementación e hiperparámetros

En esta sección describimos las decisiones de implementación para las tres métricas consideradas: CCA, CKA y ASIF. Además, detallamos el proceso de selección de hiperparámetros para cada una de estas, en los casos que fue necesario. Todo el código se encuentra disponible en https://github.com/ceciliabolanos/layer-analysis.

3.3.1. CCA (Canonical Correlation Analysis)

Implementamos el Análisis de Correlación Canónica (CCA) utilizando la biblioteca `sklearn`, específicamente a través de la clase `CCA` del módulo `sklearn.cross_decomposition`. La función CCA maneja de manera automática la normalización de los datos. Para determinar la medida de similitud entre dos representaciones, calculamos la correlación promedio entre las componentes canónicas de ambos conjuntos de datos transformados.

3.3.2. CKA (Centered Kernel Alignment)

En el caso de CKA, optamos por utilizar Linear CKA, realizando una implementación propia a partir de la fórmula matemática correspondiente. La fórmula asume que las matrices están centradas, lo cual aseguramos en nuestra función. Cada matriz estará compuesta por las 46,906 representaciones de las palabras obtenidas para algún modelo y en alguna capa específica. A continuación, se presenta el código de nuestra implementación en Python:

```
def linear_CKA(X, Y):
    X_centered = X - X.mean(axis=0)
    Y_centered = Y - Y.mean(axis=0)
    hsic = np.linalg.norm(X_centered.T @ Y_centered, 'fro') ** 2
    var1 = np.linalg.norm(X_centered.T @ X_centered, 'fro')
    var2 = np.linalg.norm(Y_centered.T @ Y_centered, 'fore')

    return hsic / (var1 * var2)
```

[5] Representaciones obtenidas a través de https://github.com/mrpep/easyaudio considerando el checkpoint correspondiente a la tercera iteración del modelo https://github.com/microsoft/unilm/tree/master/beats.

Tab. 3.1: Resultados del Accuracy zero-shot para diferentes valores de val_exp (p) y non_zeros (k) y diferentes combinaciones de modelos.

val_exp (p)	**non_zeros (k)**	**Modelo de Audio**	**Modelo de Texto**
6	250	Wav2Vec 2.0	GloVe
4	100	Wav2Vec 2.0	BERT
1	100	BEATs	GloVe
6	250	BEATs	BERT
6	250	EncodecMAE (Encodec)	GloVe
4	100	EncodecMAE (Encodec)	BERT
6	250	EncodecMAE (Mel)	GloVe
4	100	EncodecMAE (Mel)	BERT

3.3.3. ASIF

Para evaluar la similitud utilizando ASIF entre dos conjuntos de representaciones de datos, empleamos el *accuracy zero-shot*. Específicamente, reservamos el 15 % de las observaciones de los conjuntos $X \in \mathbb{R}^{n \times p_1}$ e $Y \in \mathbb{R}^{n \times p_2}$ para la evaluación. Es importante mencionar que esta selección aleatoria puede resultar en que algunas palabras no tengan pares correspondientes en el conjunto para realizar el *retrieval*, además de reducir la cantidad de observaciones disponibles para la comparación. Es decir, la métrica se calcula sobre 7,035 representaciones, utilizando las 39,871 restantes para definir los espacios que utiliza ASIF.

El *accuracy zero-shot* lo medimos sobre el conjunto reservado, lo que implica verificar si, dada una palabra y su representación en X, ASIF es capaz de asociarla correctamente con su representación correspondiente en Y. Este proceso lo realizamos utilizando el código oficial de la implementación de ASIF[6], centrando la evaluación exclusivamente en el cálculo del *accuracy*.

Para el uso de ASIF, a diferencia de CCA y CKA, es necesario seleccionar los hiperparámetros p y k. Para esto, consideramos las representaciones de las capas cuyo valor de Linear CKA haya sido máximo. Basándonos en la metodología de búsqueda descrita en el artículo de referencia, inicialmente configuramos una grilla de búsqueda con valores de p variando de 1 a 4 y k de 250 a 750 en pasos de 250.

Para todos los modelos, esta búsqueda arrojó al punto $(4, 250)$ como valor óptimo. Además, por la forma en la que estaban distribuidos los valores, sugerían la posibilidad de obtener un *accuracy* aún mayor con valores más altos de p y, a su vez, valores más chicos de k. Por está razón, realizamos una búsqueda más detallada incluyendo los pares $(1, 100)$, $(2, 100)$, $(3, 100)$, $(4, 100)$, $(4, 150)$, $(4, 200)$, $(4, 250)$, $(5, 250)$, $(6, 250)$, $(7, 250)$, $(8, 250)$, $(9, 250)$. La Tabla 3.1 muestra los valores óptimos para este algoritmo para cada una de las combinaciones entre modelos.

[6] https://github.com/noranta4/ASIF

4. EXPERIMENTACIÓN

En el trabajo de Pasad et al. [2021], se evalúa la similitud entre las representaciones generadas por las distintas capas de modelos Wav2Vec 2.0 y GloVe usando CCA como métrica principal. En este trabajo proponemos en primer lugar cambios metodológicos que implican mejoras en cuanto a requerimientos computacionales (Sección 4.1) y la inclusión de ASIF como técnica complementaria (Sección 4.2).

En segundo lugar, exploramos cómo se comportan nuevos modelos que incluyen BERT, BEATs y EncodecMAE. Para ello en la Sección 4.3 exploramos cómo varían las representaciones de nuestras instancias dentro de un mismo modelo – al variar las capas seleccionadas de los mismos. Y luego, en la Sección 4.4, cómo se comparan las representaciones generadas al variar la modalidad sobre la cual se generaron estas representaciones.

4.1. CKA como reemplazo a CCA

Al buscar replicar el trabajo de Pasad et al. [2021] encontramos limitaciones de cómputo debido a la complejidad algorítmica de CCA (ver Sección 2.3.4). Por esta razón, proponemos entonces utilizar la técnica Linear CKA como reemplazo. Para verificar que ambas métricas den resultados cualitativamente similares realizamos tres experimentos: (*E1*) realizamos una comparación visual entre los resultados encontrados en el paper y los nuestros, en los que utilizamos Linear CKA; (*E2*) Analizamos la estabilidad del método Linear CKA ante variaciones en la cantidad de datos; (*E3*) replicamos el experimento original utilizando tanto CCA como Linear CKA, pero sobre un subconjunto reducido de los datos (3,000 de las 46,906 instancias).

Vemos primero en la Figura 4.1 el resultado original encontrado por los autores del paper que buscamos replicar. Este gráfico muestra cómo hay una mayor similaridad entre las representaciones generadas en las capa 7 y las obtenidas mediante Glove. Adicionalmente, este gráfico también muestra como finetunear Wav2Vec 2.0 para una tarea específica acerca las representaciones en capas subsiguientes.

En la Figura 4.2, observamos el gráfico generado para el experimento *E1* al correr la comparación entre los mismos modelos (Wav2Vec 2.0 y Glove), pero esta vez utilizando Linear CKA. A pesar de algunas diferencias, podemos observar similitudes en los patrones generales obtenidos: un pico en la capa 7, un decrecimiento sostenido luego de esa capa y un salto en la capa final del modelo.

En la Figura 4.3 vemos el resultado del experimento *E2*. Cada panel representa el resultado obtenido al variar la cantidad de datos utilizados entre 50 y 39,000 instancias. Por su parte, la Figura 4.4 muestra la correlación de Pearson entre cada una de las curvas y la generada con el conjunto completo de datos (46,906 instancias). Como puede verse en la figura, la métrica se muestra estable incluso desde los 3,000 ejemplos ($\rho = 0{,}9988$).

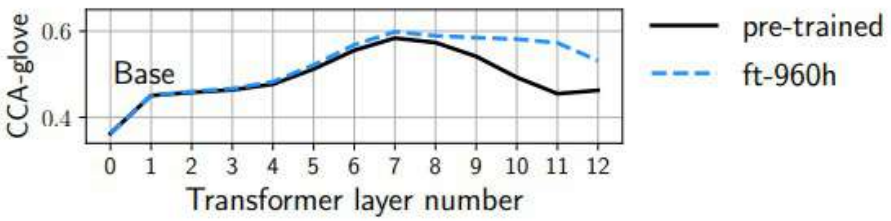

Fig. 4.1: Gráfico tomado del estudio Pasad et al. [2021], que muestra la comparación de representaciones entre el modelo Wav2Vec 2.0 y GloVe. Este análisis incluye tanto el modelo preentrenado, también utilizado en nuestro análisis, como el modelo ajustado posteriormente (finetuneado). Además, extienden la evaluación más allá de las capas de Transformer para incluir la salida de la red convolucional.

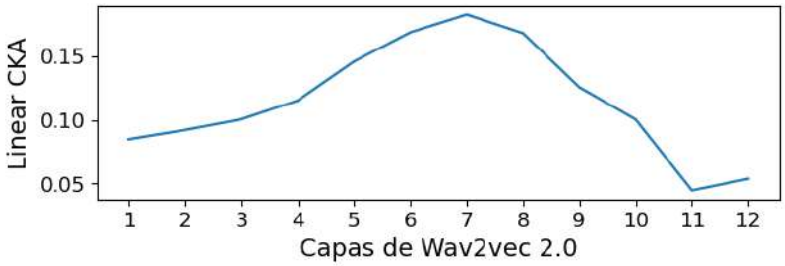

Fig. 4.2: Comparación, usando Linear CKA, de las representaciones de palabras generadas por GloVe y por cada capa de Wav2vec 2.0.

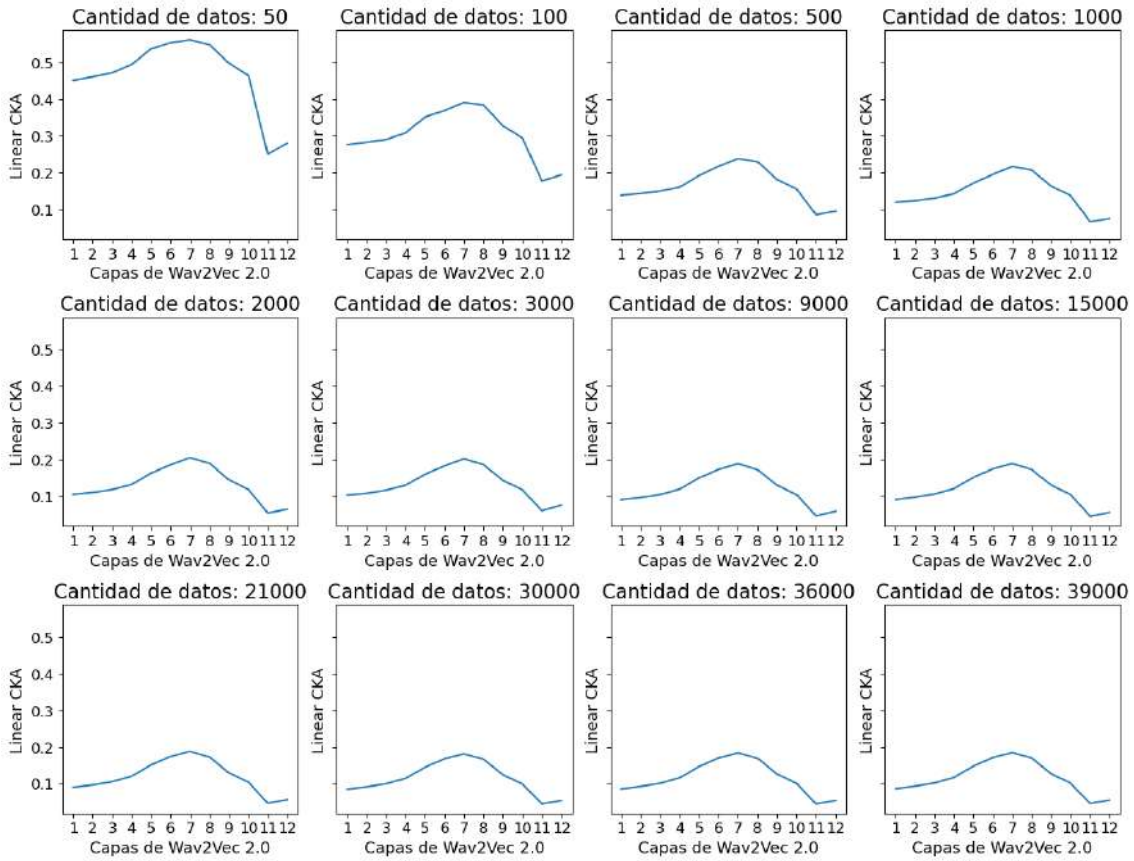

Fig. 4.3: Evolución de Linear CKA en función de la cantidad de datos. Cada subgráfico muestra cómo cambia la métrica Linear CKA para cada capa de Wav2Vec 2.0 cuando se utiliza una cantidad específica de datos, variando desde 50 hasta 39,000 observaciones.

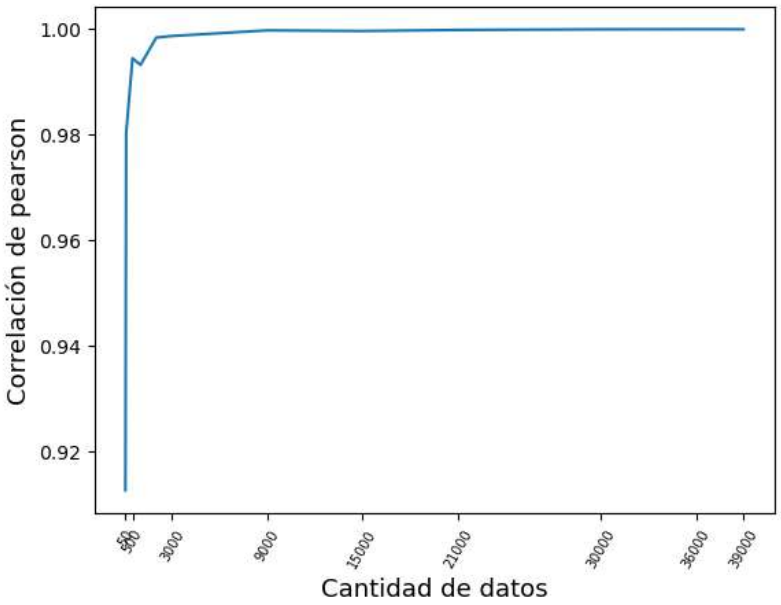

Fig. 4.4: Correlación de Pearson entre cada curva generada con diferentes subconjuntos de datos y la curva de referencia, que se calcula utilizando el conjunto completo de datos.

A continuación, analizamos el resultado de *E3*, en donde comparamos las curvas generadas por Linear CKA y CCA utilizando 3000 ejemplos. En la Figura 4.5, observamos que ambas métricas mantienen tendencias similares, destacando un valor máximo en la capa 7 de Wav2Vec 2.0 y un incremento en la similitud entre las capas 11 y 12, siendo más notable para Linear CKA.

Vemos entonces que Linear CKA presenta valores más bajos de similitud, lo cual podría explicarse por la mayor flexibilidad que ofrece CCA para modificar las representaciones y hacerlas más similares. No obstante, ambas métricas generan tendencias similares, por lo tanto, decidimos utilizar Linear CKA en futuros experimentos debido a su viabilidad computacional.

4.2. ASIF como métrica complementaria

Para validar ASIF como métrica de similaridad, utilizamos el 15 % de los datos que habíamos reservado, evaluando si esta métrica conserva las tendencias observadas en las curvas generadas por métricas anteriores a partir de las representaciones de Wav2Vec 2.0 y GloVe.

La Figura 4.6 presenta un análisis comparativo que muestra cómo la tendencia de la curva de ASIF se alinea con las tendencias observadas en las métricas Linear CKA y CCA. También notamos que, igual que en la sección anterior, la tendencia de la curva permanece, aunque la escala del valor de similitud varía.

Este último resultado nos permite utilizar ASIF como técnica complementaria para calcular la similitud entre espacios de representación, lo que introduce importantes novedades dado que es una métrica más interpretable que las anteriores. Esta interpretabilidad proviene del método de cálculo de la similitud, donde se evalúa la precisión en clasificación 'zero shot'. Es decir, medimos cuántas palabras pueden ser correctamente asociadas con su

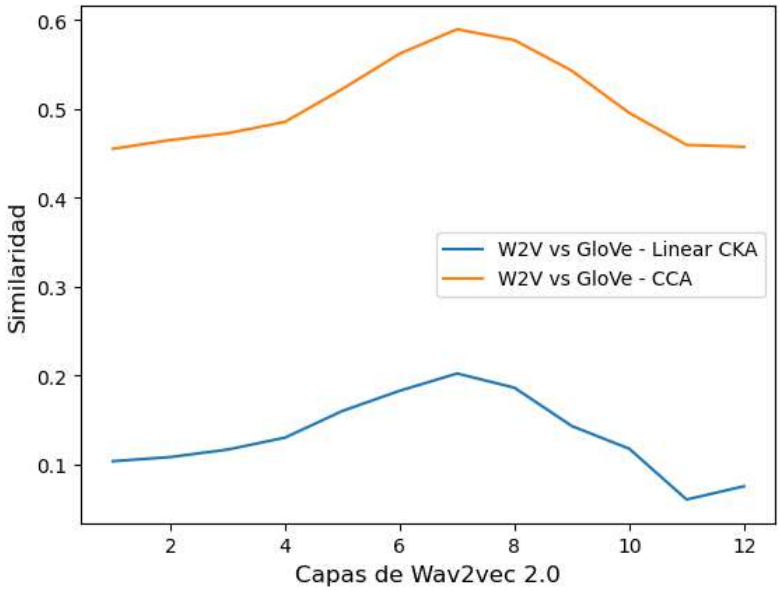

Fig. 4.5: Comparación entre las curvas de CCA y de Linear CKA que se generan considerando los primeras 3000 observaciones del conjunto completo.

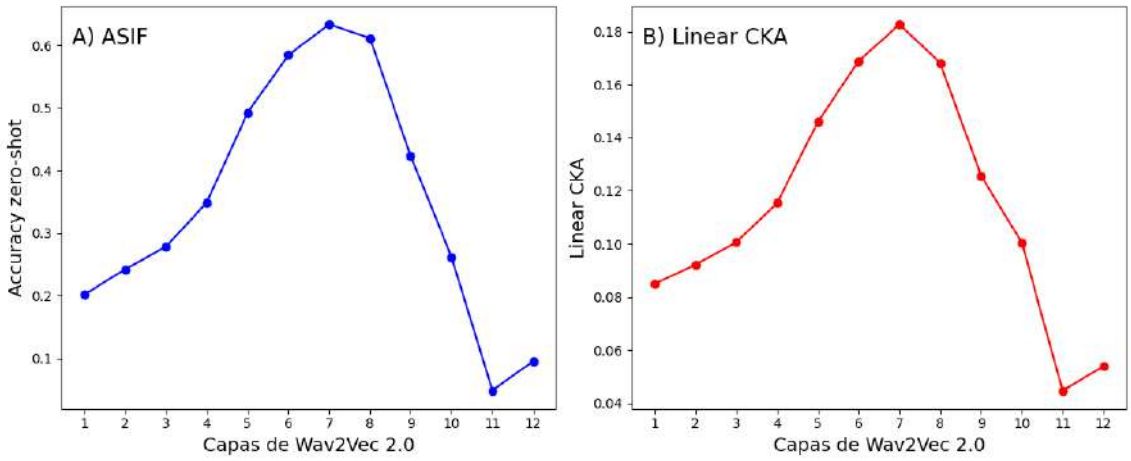

Fig. 4.6: Validación de la preservación de tendencias en el uso de diferentes métricas. (A) ASIF: Accuracy zero-shot para cada capa del modelo Wav2Vec 2.0, utilizando la métrica ASIF con GloVe. (B) Linear CKA: Análisis de la similitud, usando Linear CKA, entre las representaciones de GloVe y de Wav2Vec 2.0 para cada capa del modelo.

representación en texto a partir únicamente de su representación en audio. Un valor alto en esta métrica indica que muchas de las palabras evaluadas conservan un lugar similar en ambos espacios. Además, permite analizar las palabras donde no se logra esta correspondencia, explorando las razones detrás de estos fallos para proporcionar más información sobre los procesos en los espacios de representación.

4.3. Experimentos en un mismo modelo

En esta sección, nos proponemos entender de manera más profunda y detallada los modelos que incorporamos al análisis, explorando cómo representan y procesan la información a través de sus capas. Tal como introducimos en la Sección 2.2, las representaciones varían en función de múltiples factores de diseño, como el conjunto de datos de entrenamiento, los objetivos de entrenamiento y la función de pérdida.

Con este objetivo en mente, examinamos las variaciones en las representaciones de los modelos seleccionados para este estudio. Nos enfocamos en identificar las diferencias entre estos modelos para comprender qué indican estos resultados sobre lo que aprenden.

Posteriormente, realizamos una exploración más específica con el modelo EncodecMAE, para el cual contabamos con diferentes configuraciones dependiendo del conjunto de datos de entrenamiento utilizado. Nos centramos en evaluar cómo la dinámica interna del modelo responde a diferentes datasets, examinando si las diferencias en la composición y características de estos conjuntos de datos pueden provocar cambios notables en la manera en que el modelo procesa y representa la información de habla.

4.3.1. ¿Cómo cambian las representaciones capa a capa?

Para comprender en profundidad la dinámica interna de las representaciones generadas por distintos modelos preentrenados, seleccionamos cuatro modelos representativos: BERT, Wav2Vec 2.0, BEATs y EncodecMAE. La Figura 4.7 presenta los mapas de calor resultantes de los cálculos de Linear CKA, mostrando visualmente la similitud entre las representaciones de las diferentes capas de cada modelo.

En el modelo BERT, observamos que la similitud entre capas alcanza su máximo cuando son adyacentes y disminuye a medida que la distancia entre las capas aumenta. De este modo, cada capa refina las representaciones de las anteriores, ajustando características específicas para las tareas de procesamiento del lenguaje, tal como proponen en Ethayarajh [2019] donde afirman que BERT captura características superficiales en las capas inferiores, características sintácticas en las capas medias y características semánticas en las capas superiores. En contraste, Wav2Vec 2.0 muestra un patrón interesante en el que la similitud dentro de una capa específica disminuye progresivamente hasta la séptima capa, para luego incrementarse. Este comportamiento, que se puede ver en más detalle en la Figura 4.8, indica un proceso similar a un autoencoder, donde el modelo va codificando la información hasta una representación central y luego, desde esa representación, intenta reconstruir la entrada. Además, las últimas dos capas muestran diferencias notables respecto a las anteriores, lo que sugiere una optimización enfocada en la tarea específica de entrenamiento del modelo: identificar la representación correcta entre un conjunto de distractores.

Finalmente, los modelos BEATs y EncodecMAE exhiben un patrón similar al de BERT en cuanto a la similitud entre capas adyacentes, pero con menos variación entre capas distantes. Cabe destacar que, al ser modelos de audio, fueron entrenados con diversos sonidos. Esto podría explicar por qué la representación de palabras, que es lo que se

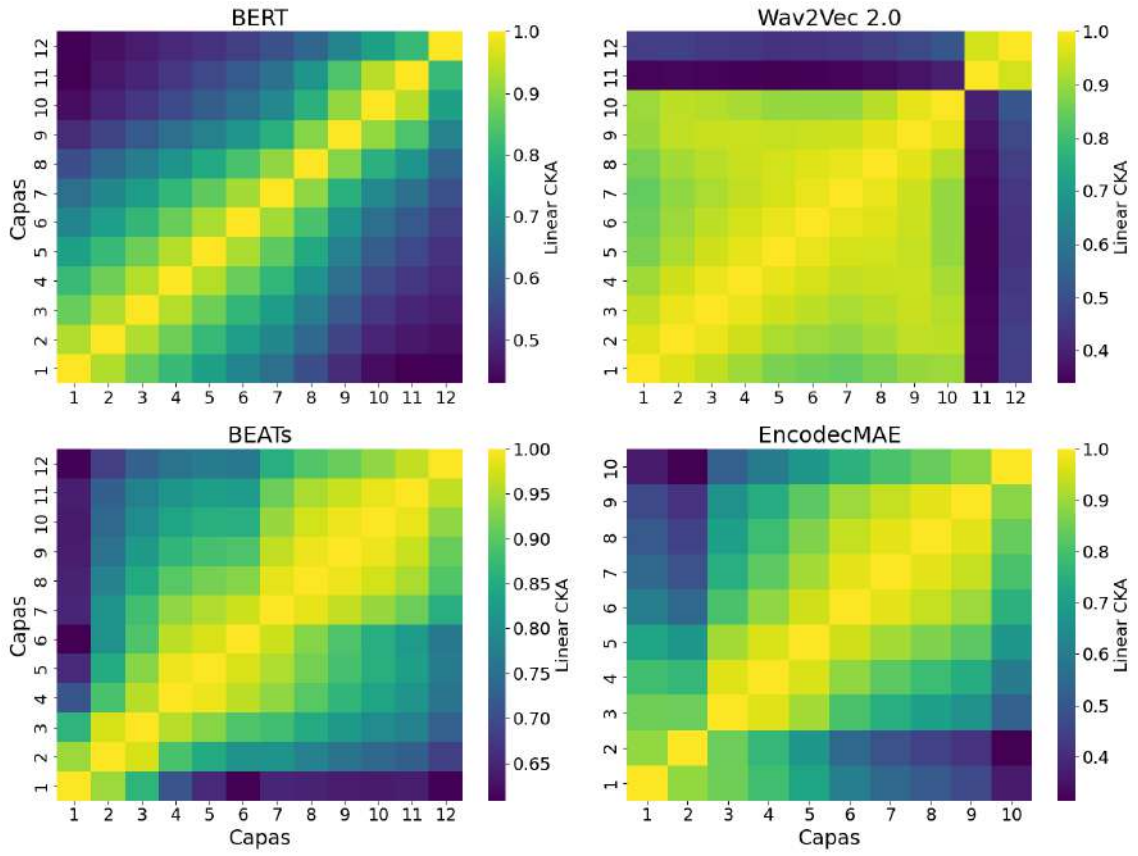

Fig. 4.7: Mapas de calor de Linear CKA para las representaciones internas de los modelos BERT, Wav2Vec 2.0, BEATs y EncodecMAE, ilustrando cómo las representaciones varían a lo largo de las capas.

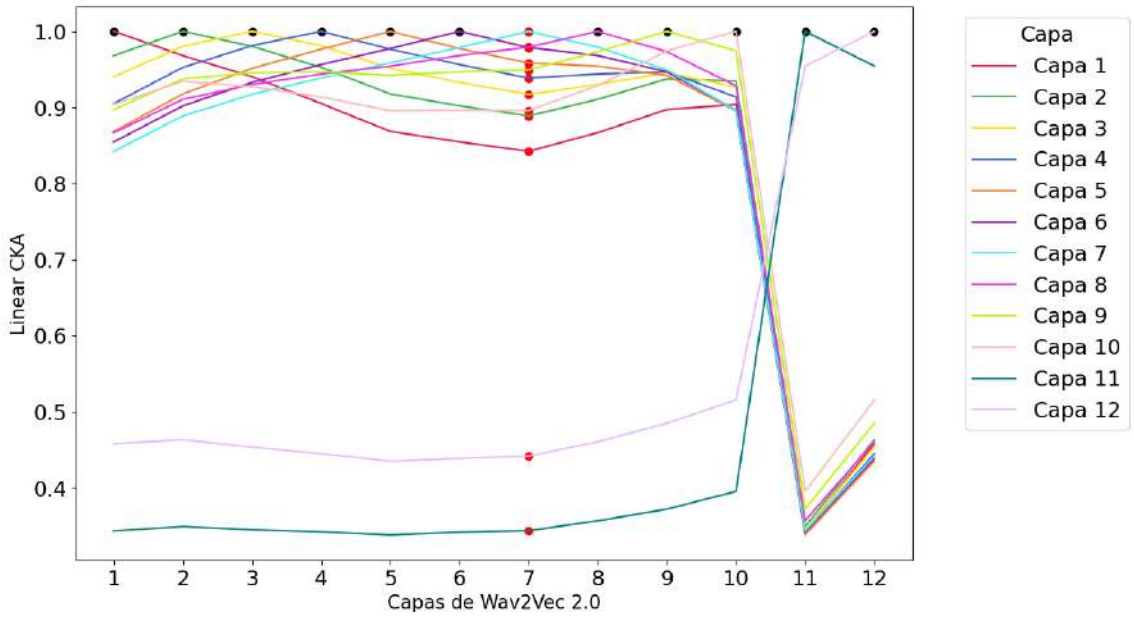

Fig. 4.8: Análisis de similitud intercapa utilizando Linear CKA para cada capa del modelo Wav2Vec 2.0, comparado con las demás capas. Las marcas negras indican la similitud de cada capa consigo misma. Se observa un patrón distintivo donde la similitud, de una capa específica, disminuye hasta la capa 7 (resaltada con marcas rojas) y luego aumenta, sugiriendo un proceso similar a un autoencoder. Este comportamiento indica que el modelo codifica la información hasta una representación central en la capa 7, y posteriormente intenta reconstruir la entrada a partir de esta representación.

observa en estos gráficos, no cambia entre las capas de manera tan notoria como en modelos puramente textuales. En estas capas, se podría estar aprendiendo sobre otros aspectos de la señal de audio, más allá del contenido lingüístico.

4.3.2. ¿Cómo impacta el dataset en las representaciones generadas?

Anteriormente, discutimos cómo varían las representaciones a lo largo de las capas en diferentes modelos. Notamos que en modelos generales de audio, la representación de palabras muestra poca variabilidad, posiblemente debido al tipo de datos de entrenamiento. Una pregunta pertinente que surge de este análisis es si el dataset de entrenamiento impacta en cómo se modifican estas representaciones. Para investigar esta cuestión, consideramos el modelo EncodecMAE entrenado con tres datasets diferentes y, además, con la combinación de todos ellos. Esta configuración del EncodecMAE utiliza como *Feature Extractor* al EspectrogramaMel.

Los datasets empleados para el preentrenamiento incluyen Audioset [Gemmeke et al., 2017], Free Music Archive [Defferrard et al., 2017] y LibriLight [Kahn et al., 2020]. Audioset es una extensa colección de más de 2 millones de clips de audio de 10 segundos, extraídos de videos de YouTube, que cubren 527 eventos de audio distintos y suman alrededor de 4500 horas de audio. Por otro lado, Free Music Archive ofrece una diversidad de 106,574 pistas de 30 segundos, que cubren 161 géneros y suman más de 800 horas de música y LibriLight consta de 6000 horas de discursos provenientes de audiolibros.

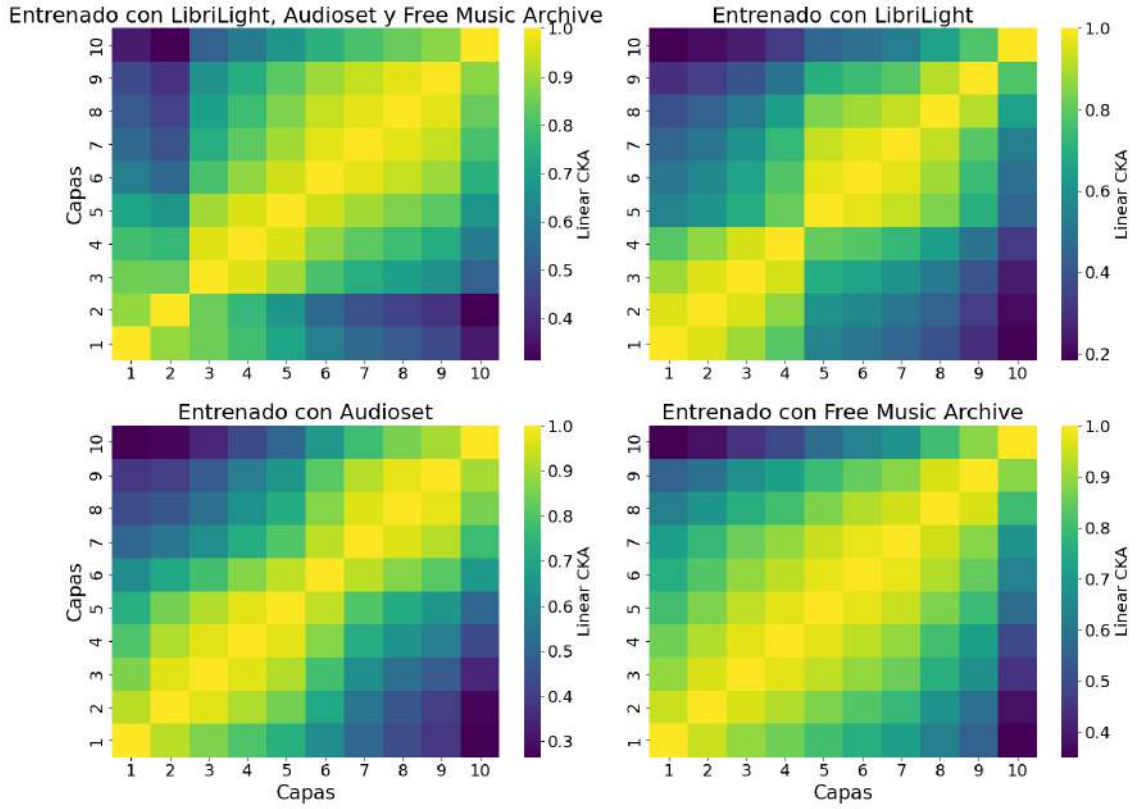

Fig. 4.9: Mapas de calor de Linear CKA para las representaciones internas del modelo Encodec-MAE, entrenado con diferentes datasets.

En la Figura 4.9, observamos los mapas de calor de Linear CKA para las representaciones internas del modelo EncodecMAE entrenado con diferentes datasets. Si bien se preserva en cierta medida la estructura jerárquica observada en BERT, donde las capas adyacentes muestran mayor similitud, en este caso vemos una formación de clusters que varía según el dataset de entrenamiento. Para los datasets de Audioset y Free Music Archive, observamos clusters más amplios y difusos, indicando una mayor uniformidad en la transformación de las representaciones a través de las capas. Esto podría atribuirse a la naturaleza diversa de estos datasets, que incluyen una amplia variedad de sonidos y posiblemente más variabilidad acústica. En contraste, el mapa correspondiente al dataset de LibriLight muestra dos clusters más definidos y contrastados. Dado que LibriLight es un dataset centrado en habla de audiolibros, esta estructura más marcada podría reflejar una adaptación más específica a las características del lenguaje hablado.

Estos hallazgos indican que para desarrollar representaciones de palabras más ricas, resulta beneficioso entrenar con un conjunto de datos que contenga principalmente contenido hablado. En Pepino et al. [2024], analizan el desempeño de estos modelos en tareas downstream, observando que los modelos con mejores resultados en las tareas de reconocimiento de habla fueron aquellos entrenados con todas las bases de datos y específicamente con LibriLight. Estos resultados confirman nuestras observaciones sobre la importancia de elegir un dataset de entrenamiento que esté alineado con la tarea específica que se pretende realizar.

4.4. Modelos de Distintas Modalidades

En etapas anteriores de este estudio, el foco principal estuvo en analizar cómo la información se propaga a través de distintas redes neuronales. En esta etapa, ampliaremos nuestro enfoque hacia la comparación de las representaciones generadas por modelos que procesan diferentes modalidades de entrada y que fueron entrenados con diferentes objetivos. Cabe recordar que *GloVe* está diseñado para producir representaciones útiles de palabras, mientras que *BERT* se centra en aprender representaciones generales del lenguaje; *Wav2Vec 2.0* busca generar representaciones para el reconocimiento del habla, y tanto *BEATs* como *EncodecMAE* se orientan hacia representaciones generales de audio.

El objetivo de esta sección es evaluar la similitud en las representaciones de palabras generadas por estos modelos. Para ello, se llevarán a cabo dos experimentos: el primero se centrará en analizar los espacios de representación generados, y el segundo determinará la cantidad de palabras que se representan de manera similar en ambos espacios.

En la primera parte del análisis, abordamos la comparación directa entre BERT y Wav2Vec 2.0, centrándonos exclusivamente en representaciones específicas del habla. En la segunda etapa, extendemos nuestro estudio para incluir una comparación entre los modelos de audio BEATs y EncodecMAE, y los modelos de texto GloVe y BERT. Finalmente, concluimos nuestro análisis examinando las diferencias entre las representaciones generadas por modelos de audio en general y aquellos especializados en habla, utilizando como ejemplos Wav2Vec 2.0 y EncodecMAE. Este análisis final nos ayuda a entender cómo las modalidades de audio general y habla específica influyen en lo que los modelos captan y omiten en las representaciones de palabras.

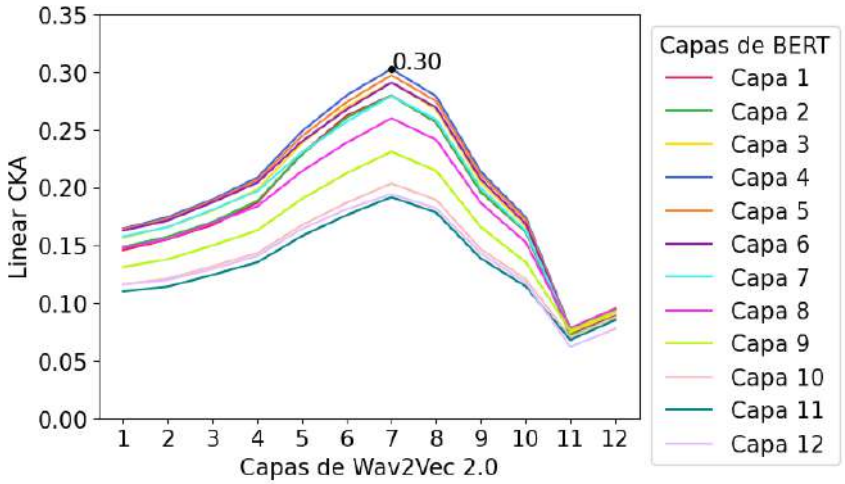

Fig. 4.10: Mapa de calor comparando Linear CKA entre las capas de BERT y Wav2Vec 2.0.

4.4.1. Habla vs Texto

BERT vs Wav2Vec 2.0

En el estudio de la comparativa entre los modelos BERT y Wav2Vec 2.0, aplicamos dos métricas de evaluación: Linear CKA y Accuracy Zero Shot, utilizando ASIF.

La Figura 4.10, muestra el Linear CKA obtenido de la comparación entre las capas de ambos modelos. Es notable que la máxima similitud sucede entre la capa 7 de Wav2Vec 2.0 y la capa 4 de BERT. Este resultado es coherente con nuestras observaciones previas, en las que identificamos que la capa 7 de Wav2Vec 2.0 era aquella que más información codificaba sobre la entrada, en este caso, la palabra. Además, este resultado es coherente con nuestras observaciones sobre cómo las capas en BERT progresan en su aprendizaje: las primeras capas tienden a codificar características superficiales del lenguaje, mientras que las capas intermedias se especializan en rasgos sintácticos, y las últimas capas capturan principalmente información semántica [Jawahar et al., 2019]. Un modelo de audio como Wav2Vec 2.0, que está diseñado para manejar señales acústicas complejas, en sus primeras capas podría estar capturando información mas local (como fonemas) mientras que, a medida que crecen las capas, podría estar integrando los fonemas a silabas y las silabas a palabras. Esto podría explicar por qué la capa 7 de Wav2Vec 2.0 alcanza una similitud máxima con una capa temprana de BERT.

En la Figura 4.11, observamos una tendencia similar en la capa 7 de Wav2Vec 2.0, destacándose por su alta eficacia al emplear sus representaciones para ASIF. No obstante, en contraste con el análisis mediante Linear CKA, el máximo valor de similitud es alcanzado en la capa 2 de BERT. Pese a ello, la discrepancia entre las primeras cuatro capas de BERT es mínima al aplicar ASIF, lo cual indica que el rendimiento podría estar condicionado por el tipo de palabras empleadas en el proceso de recuperación de información. Específicamente, mientras que ASIF mide la similitud de manera local, Linear CKA evalúa las representaciones de forma global. Por lo tanto, es posible que, aunque globalmente las representaciones sean óptimas en la capa 7 de Wav2Vec 2.0 y en la capa 4 de BERT, el análisis local de ciertas palabras pueda alterar este orden, dado que ciertas observaciones específicas están "mejor representadas" en capas cercanas. Lo importante es

que la capa 7 de Wav2Vec 2.0 presenta la mayor efectividad con cualquier capa de BERT, reforzando nuestra hipótesis de que contiene la mayor cantidad de información relevante sobre la palabra.

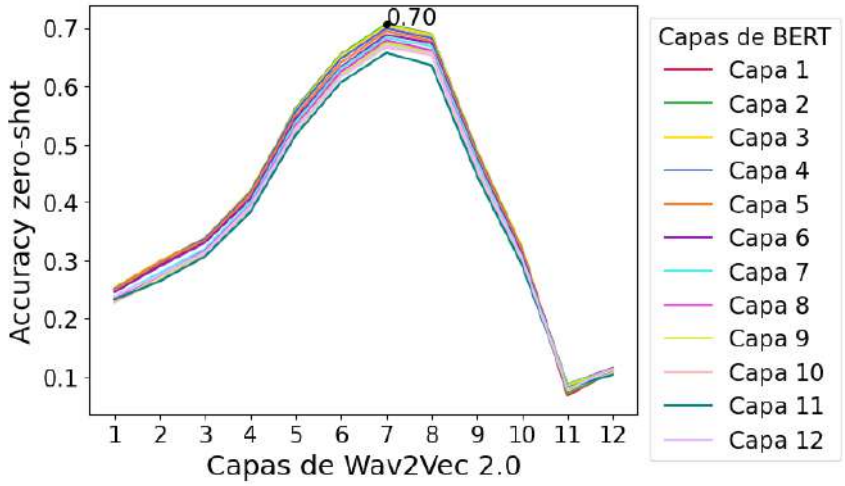

Fig. 4.11: Rendimiento de ASIF en Accuracy zero-shot utilizando las representaciones de Wav2Vec 2.0 y de BERT.

Otro aspecto relevante de ambas métricas es que la similitud alcanzada con BERT es superior a la obtenida con las representaciones de GloVe, lo cual indica que Wav2Vec 2.0 podría estar aprendiendo mejor el contexto en el que se enuncian las palabras, una capacidad que GloVe no posee. Sin embargo, este resultado también podría estar influenciado por las diferencias en las dimensiones de las representaciones, siendo que BERT cuenta con 768 dimensiones, mientras que GloVe tiene solo 300. A su vez, esta diferencia podría deberse a los sesgos de la arquitectura ya que BERT y Wav2Vec 2.0 tienen la misma arquitectura de Transformer y GloVe es mucho más sencillo. Para confirmar esta hipótesis sería necesario realizar un análisis más detallado.

4.4.2. Audio vs Texto

Ampliamos nuestro análisis comparativo a modelos de procesamiento de audio que, a diferencia de los modelos típicos de habla como Wav2Vec 2.0, no están específicamente diseñados y preentrenados para procesar y entender señales de habla, sino que sonidos en general, incluyendo también música y sonidos ambiente. Los modelos de audio considerados son BEATs y EncodecMAE, mientras que los modelos de texto evaluados son GloVe y BERT. Esta comparación nos permite explorar si los modelos generales de audio aprenden a representar a las palabras y su semántica, y si llegan a representaciones similares a modelos de texto.

BEATs

En el análisis de las representaciones generadas por BEATs, calculamos la similaridad entre estas y las de GloVe y BERT. Los resultados de este estudio son presentados en la Figura 4.12, donde observamos los valores de similaridad obtenidos.

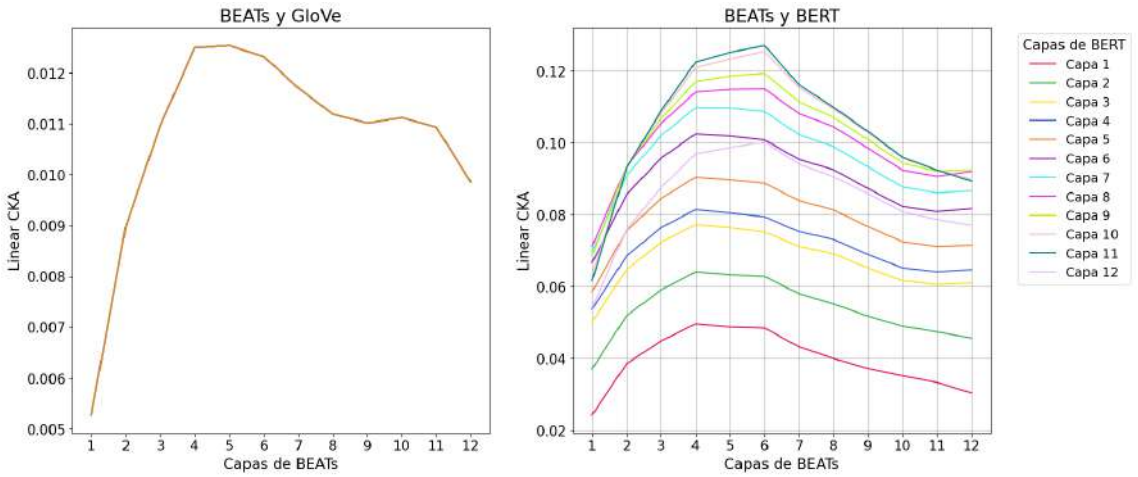

Fig. 4.12: Comparación de la similaridad entre las representaciones de BEATs y las de GloVe y de BEATs y BERT. (A) Representa la similaridad (Linear CKA) entre cada capa de BEATs y GloVe, destacando cómo evoluciona esta métrica a lo largo de las capas del modelo BEATs. (B) Muestra un mapa de calor de la similaridad (Linear CKA) entre las capas de BEATs y de BERT.

Aunque el modelo BEATs mostró un mejor rendimiento, en términos de similitud, con BERT que con GloVe, los valores obtenidos son relativamente bajos. Sin embargo, los autores del modelo BEATs presentaron que tiene un buen desempeño en en algunas tareas de habla como reconocimiento de comandos y emociones, pero no hicieron un analisis del desempeño en transcripcion de habla. Consideramos que los bajos valores en similitud pueden deberse a cómo se obtiene la representación de palabras en BEATs: al utilizar parches de EspectrogramaMel que cubren múltiples ventanas de tiempo, el modelo no sólo captura información de la palabra de interés, sino también de otras palabras presentes en la señal de entrada. Esta "contaminación" en la representación podría afectar la capacidad del modelo para reflejar con precisión las características lingüísticas específicas, resultando en una menor similitud en comparación con modelos puramente textuales como GloVe y BERT.

Dada la baja similaridad observada, realizamos un análisis adicional utilizando ASIF para continuar evaluando estas representaciones. Sin embargo, los resultados obtenidos con ASIF fueron malos también, con valores por debajo de 0.01, lo que hizo inviable cualquier conclusión significativa basada en estos datos. Por este motivo, decidimos no incorporar estos resultados de ASIF en el trabajo final.

EncodecMAE

Otro de los modelos de audio considerados para el análisis es EncodecMAE, en sus dos configuraciones de Feature Extractor: EspectrogramaMel y Encodec. Comparamos sus representaciones con las de GloVe y BERT mediante Linear CKA. En la Figura 4.13 visualizamos los valores de similaridad encontrados.

En la Sección 4.3, observamos cómo la arquitectura de EncodecMAE se vuelve progresivamente más específica. Esta especialización se manifiesta en una mayor concentración de información en su última capa, donde ambos modelos de texto alcanzan valores máximos

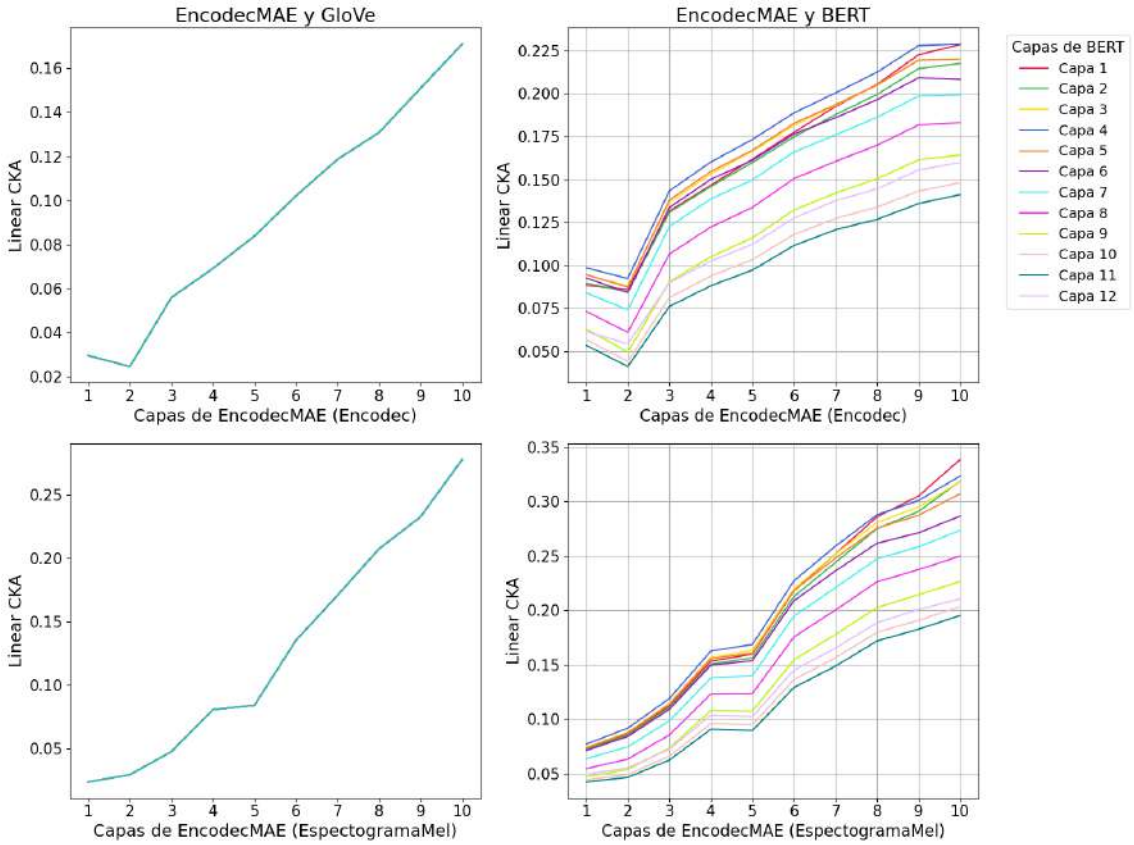

Fig. 4.13: Comparación de la similitud entre las representaciones de EncodecMAE con GloVe y BERT. A la izquierda, se ilustra la similitud (mediante Linear CKA) entre cada capa de EncodecMAE y GloVe, considerando dos configuraciones para el Feature Extractor: Encodec y EspectrogramaMel. A la derecha, se presentan los mapas de calor correspondientes que comparan estas configuraciones con las capas respectivas de BERT.

de similaridad. Adicionalmente, al analizar las capas que muestran las mayores similaridades con BERT, notamos que coinciden con las combinaciones de las últimas capas de EncodecMAE y las primeras de BERT. Este patrón corrobora nuestras observaciones anteriores: las primeras capas de BERT tienden a capturar aspectos más generales del lenguaje, de manera similar a como EncodecMAE, presumiblemente, procesa la información lingüística, centrándose en aspectos más generales.

Otro aspecto destacable de estos resultados es la diferencia con respecto a otros modelos de audio, como BEATs y Wav2Vec 2.0. En primer lugar, la similitud entre las representaciones de EncodecMAE y los modelos de texto resulta significativamente más alta en comparación con BEATs, lo cual sugiere que evitar el uso de parches en el manejo de señales continuas es una decisión acertada. En segundo lugar, como se observa en la segunda fila, EncodecMAE utilizando un EspectogramaMel supera los valores máximos obtenidos por Wav2Vec 2.0, tanto con GloVe como con BERT. Este hallazgo es particularmente relevante dado que EncodecMAE no está especializado en el reconocimiento del habla, aunque muestra un desempeño robusto en tareas relacionadas. Este resultado, sugerido por el Linear CKA, indica que EncodecMAE con EspectogramaMel podría estar capturando información lingüística de manera más efectiva que Wav2Vec 2.0, aunque tal interpretación requiere un análisis más detallado que se explorará en secciones posteriores.

Un análisis adicional que surge de estos resultados investiga si EncodecMAE, utilizando EspectrogramaMel y entrenado exclusivamente con datos de habla, podría alcanzar una similitud aún mayor con las representaciones de BERT, superando significativamente a Wav2Vec 2.0. Contando con variaciones de este modelo entrenado en diferentes conjuntos de datos, decidimos realizar este experimento. Los resultados, que se muestran en la Figura 4.14, reafirman la tendencia observada: los valores máximos de CKA se localizan en las últimas capas de EncodecMAE y en las primeras capas de BERT. Es importante destacar que el valor máximo de Linear CKA cambia en función del conjunto de datos empleado en el entrenamiento, siendo el más elevado aquel obtenido con datos de habla.

Los resultados obtenidos indican que el espacio de representación de palabras generado por EncodecMAE es más cercano al de BERT que el generado por Wav2Vec 2.0. Sin embargo, estos hallazgos no nos permiten concluir definitivamente qué están aprendiendo exactamente estos modelos sobre las palabras. Para profundizar en nuestra comprensión de las representaciones generadas por EncodecMAE y compararlas con las de los modelos de texto, procederemos a utilizar ASIF.

En la Figura 4.15 presentamos los resultados de este experimento. A partir de estos datos, podemos explorar dos aspectos clave: las diferencias encontradas con la métrica anterior y cómo estos resultados afectan la hipótesis planteada inicialmente, donde sugeríamos que EncodecMAE podría estar aprendiendo de manera más efectiva sobre las palabras que Wav2Vec 2.0.

En primer lugar, notamos que las capas con máxima similitud según ASIF difieren de aquellas identificadas por Linear CKA; sin embargo, esto no supone un problema. Las capas superiores de EncodecMAE mantienen altos valores de similitud con las primeras capas de BERT, que tienden a captar características lingüísticas generales. Esta diferencia sugiere que, aunque Linear CKA pueda identificar una capa como óptima a nivel global por su similitud, el análisis de representaciones de palabras específicas puede beneficiarse de observar capas adyacentes que ofrezcan mejores representaciones para esos casos particulares.

En segundo lugar, estos resultados nos permiten evaluar la hipótesis sobre Encodec-

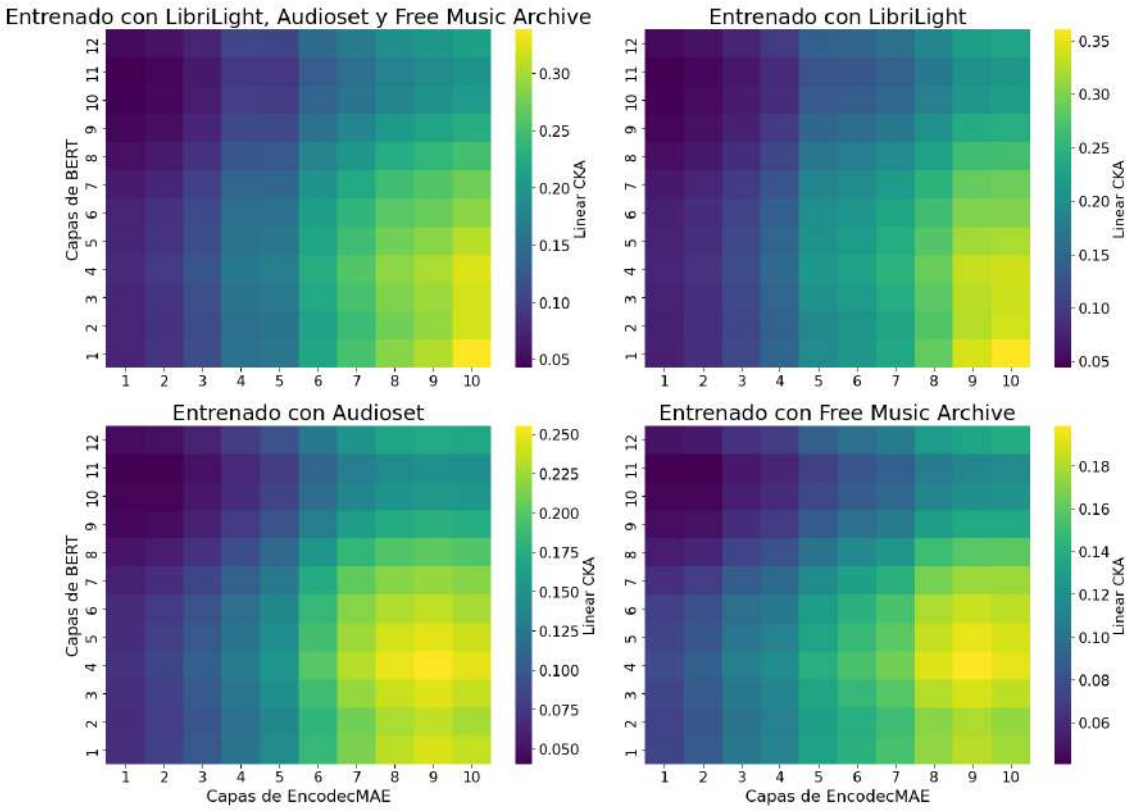

Fig. 4.14: Resultados de la similitud calculada mediante Linear CKA entre representaciones de las capas de EncodecMAE y las de BERT. Para EncodecMAE se consideran cuatro configuraciones diferentes variando el dataset de entrenamiento.

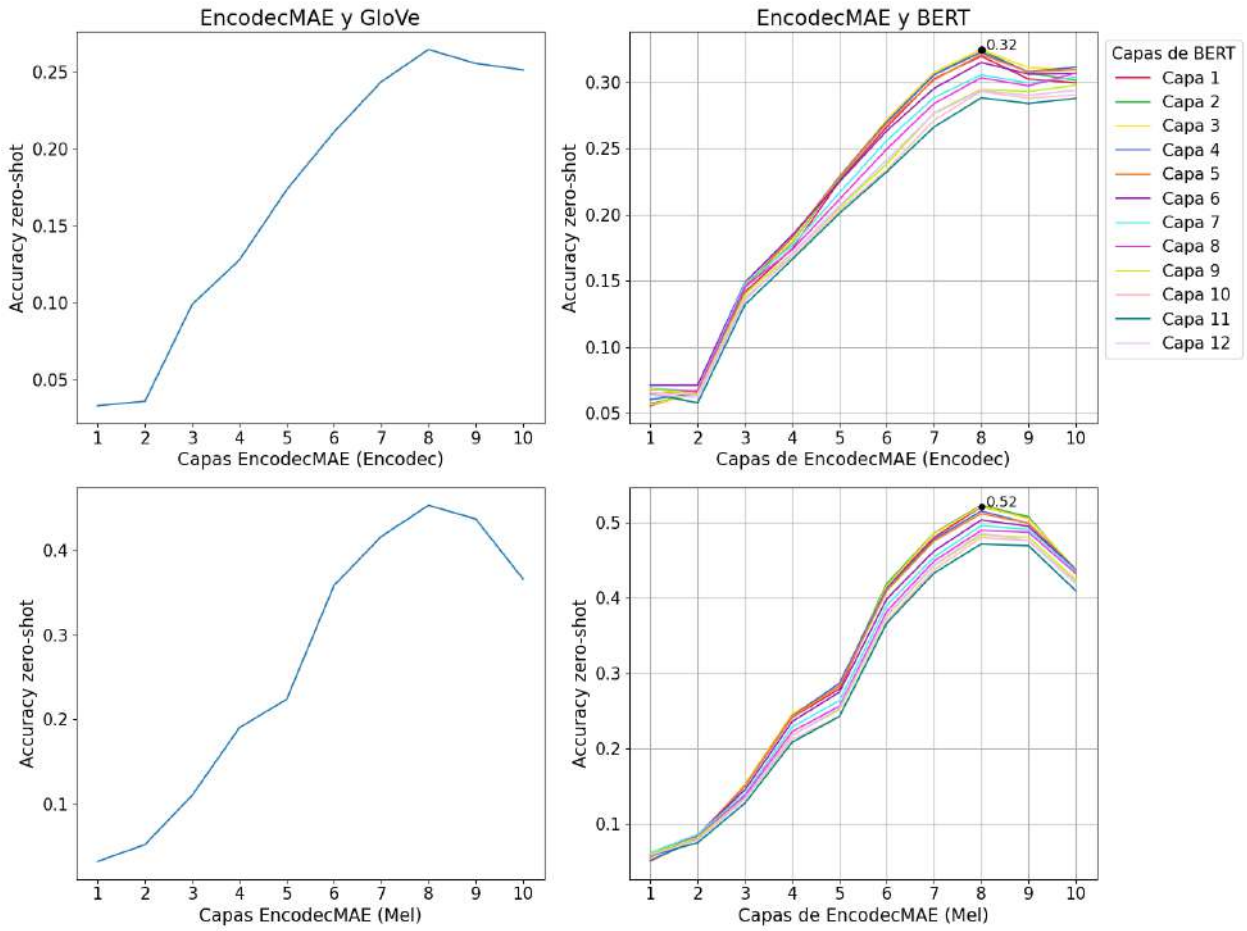

Fig. 4.15: Resultados de la similitud calculada mediante ASIF entre representaciones de EncodecMAE y GloVe y EncodecMAE y BERT.

MAE, que sugería que este modelo podría estar aprendiendo representaciones más eficaces de palabras. Esto se basa en que EncodecMAE logró una mayor similitud en Linear CKA con BERT, superando los valores alcanzados por Wav2Vec 2.0 con BERT. Sin embargo, al incorporar ASIF, observamos que los valores para EncodecMAE son inferiores a los de Wav2Vec 2.0. Dado que ASIF evalúa la capacidad del modelo para identificar palabras concretas a través de su representación en audio, sugiere que podría ser una métrica más relevante para medir la eficacia en el reconocimiento de habla. Esto resalta cómo una dependencia exclusiva en Linear CKA podría llevarnos a conclusiones erróneas sobre el desempeño real de un modelo. Por tanto, la inclusión de ASIF permite una evaluación complementaria a la realizada con Linear CKA y, a su vez, más completa de las representaciones.

4.4.3. Habla vs Audio

Para finalizar nuestros experimentos nos gustaría profundizar en las capacidades de representación de los modelos de habla y audio. Para esto, nos centraremos en Wav2vec 2.0 y EncodecMAE, en sus dos configuraciones, excluyendo a BEATs debido a resultados insatisfactorios en experimentos anteriores. En lugar de medir el accuracy zero-shot de recuperación, nuestro enfoque será el análisis de las palabras que no aparecieron dentro del top 5 de los resultados devueltos por estos modelos en el marco del ASIF. Evaluaremos estos resultados utilizando la combinación de capas que maximiza la métrica.

El objetivo de este estudio es explorar los casos en los cuales los modelos no lograron incluir la palabra adecuada entre los primeros cinco resultados. Esta indagación nos ayudará a identificar patrones de errores o deficiencias en las representaciones. Detallaremos el análisis de las palabras no recuperadas para determinar si las representaciones estuvieron cerca de ser correctas y si existe alguna tendencia o información no capturada por los modelos.

Para Wav2Vec 2.0, el 26 % de las palabras (1803 palabras) no aparecieron en los primeros cinco resultados, mientras que para EncodecMAE, este porcentaje fue del 63 % (4482 palabras) y del 47 % (3138 palabras) para las configuraciones con Encodec y EspectrogramaMel respectivamente. Considerando palabras únicas, estas cifras se reducen a 1523, 2300 y 2053, respectivamente.

Al analizar las palabras no recuperadas, encontramos varios casos: palabras comunes entre modelos y aquellas específicas de algunos. Entre las palabras erradas por todos los modelos, 430 no aparecían en el dataset de entrenamiento, lo que es razonable dado que los modelos no tenían referencia textual para ellas. Además, encontramos 1042 palabras erradas por todos los modelos que podrían haber sido correctas; sin embargo, en vez de devolver sinónimos, los modelos tendieron a ofrecer palabras con pronunciaciones similares o palabras que no tenían ningún tipo de sentido. Algunos ejemplos de esto se pueden ver en la Tabla 4.1.

Podemos observar que los modelos de audio, específicamente Wav2Vec 2.0 y EncodecMAE utilizando EspectrogramaMel, tienden a resaltar la similaridad fonética sobre la sintáctica o semántica en las palabras que generan. Este fenómeno, también observado por Choi et al. [2024] en su estudio sobre modelos de habla, sugiere una inclinación hacia la fonética más que hacia la estructura sintáctica. Los hallazgos indican que estos modelos pueden “escuchar” y procesar la pronunciación de manera más efectiva.

Esta observación fue corroborada mediante un análisis cuantitativo, como se ilustra

Tab. 4.1: Comparación de resultados de los modelos EncodecMAE y Wav2Vec 2.0 para algunas palabras.

Palabra	**EncodecMAE (Encodec)**	**EncodecMAE (EspectrogramaMel)**	**Wav2Vec 2.0**
catching	half (x5)	cash (x2), half (x3)	catch (x2), attaching, cast, touching
rhythm	mirrors (x2), number, unborn, bay	em (x5)	room (x5)
pages	his (x5)	age (x5)	ages (x4) age
springs	us (x5)	things (x5)	sighs, spinning, offences, bliss, jars
bed	day (x5)	dead (x5)	beg, had, did (x3)
haughty	whole (x5)	whole (x5)	whole (x5)
kind	handed, on (x3), line	time (x5)	hand (x5)
holds	whole (x4), all	old (x5)	hold (x5)
found	on (x5)	wound (x3), fall (x2)	balls, mom, alley, dough, glance
wert	were (x5)	were (x5)	were (x5)
asleep	lily (x2), sleeping, sleeps, jetty	sleep (x5)	sleep (x5)
throughout	profound, prop, raw, more, womb	through (x5)	out (x5)
october	are (x5)	motive, later, our, later, are	rug, hour (x4)
fort	for (x5)	for (x5)	four (x5)
corridor	her (x5)	corner (x3), george (x2)	quarter (x5)
words	air, roar, were, whirl (x2)	word (x5)	word (x5)
campaign	man (x4), ann	man (x5)	pane (x3), man (x2)
am	him (x5)	a (x5)	him (x2), them (x3)
plan	end (x2), indian (x3)	plain (x5)	plain (x5)
instinct	indian (x4), mating	singing, age (x2), plains (x2)	in (x4), ending

en la Figura 4.16. En este análisis, calculamos la distancia fonética[1] y semántica[2] para cada par de palabras. Luego, variamos el umbral de distancia fonética para seleccionar diferentes conjuntos de pares de palabras. Para cada umbral, seleccionamos los pares de palabras cuya distancia fonética era menor que dicho umbral, y calculamos la distancia coseno promedio entre las representaciones de estos pares seleccionados. Para este cálculo, utilizamos la capa del modelo de audio que había mostrado el mejor rendimiento según la métrica ASIF. Realizamos un proceso análogo para la distancia semántica. Los resultados visualizados en la figura revelan que, para diversos umbrales, los pares de palabras seleccionados por similitud fonética consistentemente presentan una menor distancia coseno promedio en comparación con los pares seleccionados por similitud semántica. Esta evidencia empírica respalda nuestra hipótesis inicial, demostrando que los modelos tienden a capturar y representar con mayor precisión las similitudes fonéticas que las semánticas en su espacio de embeddings.

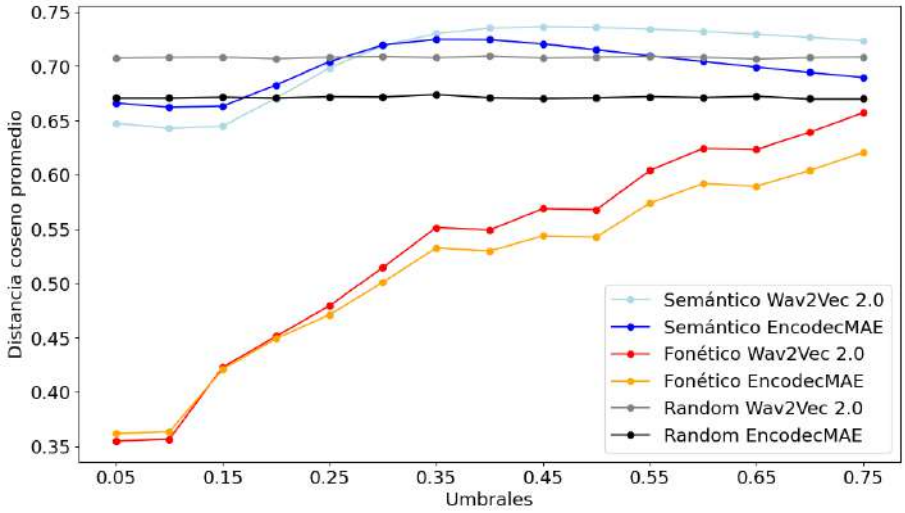

Fig. 4.16: Análisis comparativo de la distancia fonética y semántica en representaciones de palabras. Variando el umbral de distancia, vemos cómo las palabras más cercanas entre sí fonéticamente son consistentemente representadas con menor distancia coseno en los modelos, destacando la precisión en la captura de similitudes fonéticas sobre las semánticas.

Otro aspecto relevante que vemos en la Tabla 4.1 son las diferencias en la similaridad fonética entre los modelos. Por ejemplo, ante la palabra "holds", EncodecMAE tiende a captar "old", que es la parte más prolongada de la señal, mientras que Wav2Vec 2.0 identifica con precisión la "h" inicial, que es más breve. Un patrón similar ocurre con la palabra "kind", donde EncodecMAE se enfoca en la pronunciación de la vocal, devolviendo "time", mientras que Wav2Vec 2.0 reconoce "hand", que fonéticamente se asemeja más a "kind" en su terminación. Estos ejemplos sugieren que EncodecMAE podría ser más sensible a los componentes de mayor duración en la señal de audio, mientras que Wav2Vec 2.0 parece captar con mayor precisión detalles fonéticos más sutiles. Sin embargo, para

[1] La distancia fonética entre dos palabras se calcula como la distancia de Levenshtein entre las secuencias fonéticas de ambas palabras, utilizando un léxico de fonemas. En este caso, el diccionario de fonemas de LibriSpeech.

[2] La distancia semántica entre dos palabras se calcula como la distancia coseno de los embeddings generados por GloVe para esas palabras.

confirmar esta hipótesis y comprender plenamente las diferencias en el procesamiento de señales entre ambos modelos, se requiere un análisis más exhaustivo.

Esta diferencia en la detección fonética entre EncodecMAE y Wav2Vec 2.0 puede relacionarse con sus respectivos desempeños en diferentes tareas de reconocimiento de voz. Según Pepino et al. [2024], mientras EncodecMAE alcanza un rendimiento similar al de Wav2Vec 2.0 en el reconocimiento de comandos de voz, Wav2Vec 2.0 destaca en el reconocimiento de habla continua. Esta variación se debe a las exigencias particulares de cada tarea: el reconocimiento de comandos de voz se centra en identificar opciones distintas, mientras que el reconocimiento del habla requiere captar los detalles finos entre sonidos parecidos. Este hallazgo subraya la importancia de elegir tareas de evaluación que reflejen con precisión las capacidades fonéticas específicas que se esperan de un modelo de audio.

5. CONCLUSIONES

En este trabajo, nos propusimos replicar y expandir investigaciones previas, en particular el estudio de Pasad et al. [2021], que examina cómo distintas capas del modelo de habla Wav2Vec 2.0 se alinean con las representaciones de palabras generadas por el modelo de texto GloVe mediante el Análisis de Correlación Canónica (CCA). Introducimos mejoras tanto computacionales como interpretativas, añadiendo métricas adicionales: Linear Centered Kernel Alignment (Linear CKA) [Kornblith et al., 2019], que mide la similitud global de los espacios de representación, y ASIF [Norelli et al., 2023], una métrica local que proporciona una visión más detallada de las representaciones. Además, ampliamos el alcance del estudio para incluir modelos modernos de procesamiento de audio como BEATs y EncodecMAE, así como modelos de texto contextualizados como BERT, enfocándonos en cómo estos modelos aprenden y representan palabras a partir de los datos. Concluimos nuestro estudio con una comparación exhaustiva entre los modelos de audio y de habla, identificando en qué aspectos sus aprendizajes sobre las palabras son similares y en cuáles difieren.

Nuestros resultados sugieren que tanto Linear CKA como CCA ofrecen evaluaciones equivalentes en términos de similitud entre representaciones. Sin embargo, Linear CKA se destaca como una alternativa más eficiente computacionalmente en comparación con CCA, proporcionando valores de similitud que, aunque ligeramente más bajos, son igualmente válidos para nuestro análisis. Adicionalmente, proponemos la métrica ASIF por su mayor interpretabilidad y su capacidad para analizar de manera localizada las diferentes representaciones generadas por los modelos. ASIF demostró ser eficaz en captar la evolución de las tendencias a lo largo de las capas de los modelos, ofreciendo así una herramienta valiosa para evaluar la similitud en representaciones complejas pero además dando interpretabilidad.

Nuestra investigación revela que, aunque los modelos evaluados son Transformers, las variaciones en sus entradas, objetivos y funciones a minimizar generan dinámicas internas distintas. Este análisis permitió identificar procesos específicos dentro de cada modelo y determinar las capas que aprenden información redundante. Además, confirmamos que alinear los datos de entrenamiento con la tarea específica aumenta la similaridad entre las representaciones, lo que subraya la importancia de seleccionar adecuadamente el dataset.

Respecto a los modelos de distintas modalidades, en la comparación entre habla y texto, encontramos que Wav2Vec 2.0 muestra una mayor similitud con el espacio representativo de BERT en comparación con GloVe. Esto podría sugerir que Wav2Vec 2.0, al aprender sobre palabras, también considera el contexto en el que se enuncian. Sin embargo, esta hipótesis podría no ser completamente precisa debido a las diferencias en las dimensiones de las representaciones entre BERT y GloVe. Serán necesarios análisis adicionales para validar esta suposición.

En nuestra evaluación de las representaciones entre modelos generales de audio y texto, los resultados iniciales con BEATs no mostraron resultados positivos, destacando la necesidad de profundizar en análisis futuros. Por otro lado, en los experimentos con EncodecMAE, reafirmamos la importancia del dataset de entrenamiento. Además, la adopción de ASIF como métrica fue crucial; inicialmente, EncodecMAE mostró mayor similaridad con los modelos de texto en comparación con Wav2Vec 2.0, pero esta tendencia cambió

al aplicar ASIF, sugiriendo que una sola técnica de análisis, o método de evaluación, no es suficiente para sacar conclusiones respecto al desempeño de una representación en una tarea downstream, y que distintas técnicas de análisis como ASIF o Linear CKA, pueden no coincidir en sus resultados.

Finalmente, nuestra comparativa entre modelos de audio y habla reveló diferencias significativas en la naturaleza de los errores cometidos por EncodecMAE y Wav2Vec 2.0. Resaltamos dos hallazgos clave: primero, los modelos de audio tienden a enfocarse más en aspectos fonéticos que semánticos. Esto fue corroborado por nuestro análisis cuantitativo, donde las distancias coseno entre representaciones de palabras fueron consistentemente menores para pares con similitudes fonéticas que para aquellos con similitudes semánticas. Segundo, en términos de captura de pronunciaciones, el modelo de audio general EncodecMAE parece centrarse en segmentos más prolongados de la señal, mientras que Wav2Vec 2.0, enfocado en habla, capta detalles más sutiles esenciales para una identificación precisa de palabras. En trabajos futuros, profundizaremos estos análisis para corroborar nuestra hipótesis.

Bibliografía

Ankita Pasad, Ju-Chieh Chou, and Karen Livescu. Layer-wise analysis of a self-supervised speech representation model. *CoRR*, abs/2107.04734, 2021. URL https://arxiv.org/abs/2107.04734.

Minyoung Huh, Brian Cheung, Tongzhou Wang, and Phillip Isola. The platonic representation hypothesis. *arXiv preprint arXiv:2405.07987*, 2024.

Yu-An Chung, Yonatan Belinkov, and James Glass. Similarity analysis of self-supervised speech representations. In *ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 3040–3044, 2021. doi: 10.1109/ICASSP39728.2021.9414321.

Simon Kornblith, Mohammad Norouzi, Honglak Lee, and Geoffrey Hinton. Similarity of neural network representations revisited, 2019.

Christos Matsoukas, Johan Fredin Haslum, Moein Sorkhei, Magnus Söderberg, and Kevin Smith. What makes transfer learning work for medical images: Feature reuse & other factors. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 9225–9234, June 2022.

Zhiwei Hao, Jianyuan Guo, Kai Han, Yehui Tang, Han Hu, Yunhe Wang, and Chang Xu. One-for-all: Bridge the gap between heterogeneous architectures in knowledge distillation. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, editors, *Advances in Neural Information Processing Systems*, volume 36, pages 79570–79582. Curran Associates, Inc., 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/file/fb8e5f198c7a5dcd48860354e38c0edc-Paper-Conference.pdf.

Haoyu Chen, Jinjin Gu, Yihao Liu, Salma Abdel Magid, Chao Dong, Qiong Wang, Hanspeter Pfister, and Lei Zhu. Masked image training for generalizable deep image denoising. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1692–1703, June 2023.

Zhenda Xie, Zigang Geng, Jingcheng Hu, Zheng Zhang, Han Hu, and Yue Cao. Revealing the dark secrets of masked image modeling, 2022. URL https://arxiv.org/abs/2205.13543.

Ivan Vulić, Edoardo Maria Ponti, Robert Litschko, Goran Glavaš, and Anna Korhonen. Probing pretrained language models for lexical semantics. In Bonnie Webber, Trevor Cohn, Yulan He, and Yang Liu, editors, *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7222–7240, Online, November 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-main.586. URL https://aclanthology.org/2020.emnlp-main.586.

Kawin Ethayarajh. How contextual are contextualized word representations? Comparing the geometry of BERT, ELMo, and GPT-2 embeddings. In Kentaro Inui, Jing Jiang, Vincent Ng, and Xiaojun Wan, editors, *Proceedings of the 2019 Conference on Empirical*

Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), pages 55–65, Hong Kong, China, November 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-1006. URL https://aclanthology.org/D19-1006.

Kwanghee Choi, Ankita Pasad, Tomohiko Nakamura, Satoru Fukayama, Karen Livescu, and Shinji Watanabe. Self-supervised speech representations are more phonetic than semantic, 2024.

Mayug Maniparambil, Raiymbek Akshulakov, Yasser Abdelaziz Dahou Djilali, Sanath Narayan, Mohamed El Amine Seddik, Karttikeya Mangalam, and Noel E. O'Connor. Do vision and language encoders represent the world similarly?, 2024.

Antonio Norelli, Marco Fumero, Valentino Maiorca, Luca Moschella, Emanuele Rodolà, and Francesco Locatello. Asif: Coupled data turns unimodal models to multimodal without training, 2023.

Elnaz J. Heravi, Hamed H. Aghdam, and Domenec Puig. Classification of foods by transferring knowledge from ImageNet dataset. In Antanas Verikas, Petia Radeva, Dmitry P. Nikolaev, Wei Zhang, and Jianhong Zhou, editors, *Ninth International Conference on Machine Vision (ICMV 2016)*, volume 10341, page 1034128. International Society for Optics and Photonics, SPIE, 2017. doi: 10.1117/12.2268737. URL https://doi.org/10.1117/12.2268737.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding, 2019.

Jeffrey Pennington, Richard Socher, and Christopher Manning. GloVe: Global vectors for word representation. In Alessandro Moschitti, Bo Pang, and Walter Daelemans, editors, *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar, October 2014. Association for Computational Linguistics. doi: 10.3115/v1/D14-1162. URL https://aclanthology.org/D14-1162.

Alexei Baevski, Henry Zhou, Abdelrahman Mohamed, and Michael Auli. wav2vec 2.0: A framework for self-supervised learning of speech representations, 2020.

Sanyuan Chen, Yu Wu, Chengyi Wang, Shujie Liu, Daniel Tompkins, Zhuo Chen, and Furu Wei. Beats: Audio pre-training with acoustic tokenizers, 2022.

Leonardo Pepino, Pablo Riera, and Luciana Ferrer. Encodecmae: Leveraging neural codecs for universal audio representation learning, 2024.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need, 2023.

Monica Romero, Sandra Gomez, and Iván G. Torre. Asr advancements for indigenous languages: Quechua, guarani, bribri, kotiria, and wa'ikhana, 2024. URL https://arxiv.org/abs/2404.08368.

Zhiyun Fan, Meng Li, Shiyu Zhou, and Bo Xu. Exploring wav2vec 2.0 on speaker verification and language identification. pages 1509–1513, 08 2021. doi: 10.21437/Interspeech.2021-1280.

Changhan Wang, Anne Wu, Juan Miguel Pino, Alexei Baevski, Michael Auli, and Alexis Conneau. Large-scale self- and semi-supervised learning for speech translation. *CoRR*, abs/2104.06678, 2021. URL https://arxiv.org/abs/2104.06678.

Leonardo Pepino, Pablo Riera, and Luciana Ferrer. Emotion recognition from speech using wav2vec 2.0 embeddings. *CoRR*, abs/2104.03502, 2021. URL https://arxiv.org/abs/2104.03502.

Haytham Fayek. Speech processing for machine learning, 2016. URL https://haythamfayek.com/2016/04/21/speech-processing-for-machine-learning.html. Accessed: 2024-06-20.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale, 2021.

Alexandre Défossez, Jade Copet, Gabriel Synnaeve, and Yossi Adi. High fidelity neural audio compression, 2022.

Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners, 2021.

Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

Harold Hotelling. Relations between two sets of variates. *Biometrika*, 28(3/4):321–377, 1936.

Arthur Gretton, Olivier Bousquet, Alex Smola, and Bernhard Schölkopf. Measuring statistical dependence with hilbert-schmidt norms. In Sanjay Jain, Hans Ulrich Simon, and Etsuji Tomita, editors, *Algorithmic Learning Theory*, pages 63–77, Berlin, Heidelberg, 2005. Springer Berlin Heidelberg. ISBN 978-3-540-31696-1.

MohammadReza Davari, Stefan Horoi, Amine Natik, Guillaume Lajoie, Guy Wolf, and Eugene Belilovsky. Reliability of CKA as a similarity measure in deep learning. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=8HRvyxc606.

Vassil Panayotov, Guoguo Chen, Daniel Povey, and Sanjeev Khudanpur. Librispeech: An asr corpus based on public domain audio books. In *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5206–5210, 2015. doi: 10.1109/ICASSP.2015.7178964.

Loren Lugosch, Mirco Ravanelli, Patrick Ignoto, Vikrant Singh Tomar, and Yoshua Bengio. Speech model pre-training for end-to-end spoken language understanding, 2019.

Michael McAuliffe, Michaela Socolof, Sarah Mihuc, Michael Wagner, and Morgan Sonderegger. Montreal Forced Aligner: Trainable Text-Speech Alignment Using Kaldi. In *Proc. Interspeech 2017*, pages 498–502, 2017. doi: 10.21437/Interspeech.2017-1386.

Jort F. Gemmeke, Daniel P. W. Ellis, Dylan Freedman, Aren Jansen, Wade Lawrence, R. Channing Moore, Manoj Plakal, and Marvin Ritter. Audio set: An ontology and human-labeled dataset for audio events. In *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 776–780, 2017. doi: 10.1109/ICASSP.2017.7952261.

Michaël Defferrard, Kirell Benzi, Pierre Vandergheynst, and Xavier Bresson. Fma: A dataset for music analysis, 2017.

J. Kahn, M. Riviere, W. Zheng, E. Kharitonov, Q. Xu, P.E. Mazare, J. Karadayi, V. Liptchinsky, R. Collobert, C. Fuegen, T. Likhomanenko, G. Synnaeve, A. Joulin, A. Mohamed, and E. Dupoux. Libri-light: A benchmark for asr with limited or no supervision. In *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, May 2020. doi: 10.1109/icassp40776.2020.9052942. URL http://dx.doi.org/10.1109/ICASSP40776.2020.9052942.

Ganesh Jawahar, Benoît Sagot, and Djamé Seddah. What does bert learn about the structure of language? In *Annual Meeting of the Association for Computational Linguistics*, 2019. URL https://api.semanticscholar.org/CorpusID:195477534.