La investigación reportada en esta tesis es parte de los programas de investigación del CICESE (Centro de Investigación Científica y de Educación Superior de Ensenada, Baja California).
La investigación fue financiada por el SECIHTI (Secretaría de Ciencia, Humanidades, Tecnología e Innovación).

Centro de Investigación Científica y de Educación Superior de Ensenada, Baja California

Maestría en Ciencias en Ciencias de la Computación

Reconocimiento automático de señas dinámicas de la Lengua de Señas Mexicana

Tesis

para cubrir parcialmente los requisitos necesarios para obtener el grado de

Maestro en Ciencias

Presenta:

Jesús Antonio Navarrete López

Ensenada, Baja California, México

2025

Tesis defendida por

Jesús Antonio Navarrete López

y aprobada por el siguiente Comité

Dr. Irvin Hussein López Nava
Director de tesis

Dr. Jesús Favela Vara

Dra. María del Carmen Maya Sánchez

Dra. Isabel López Hurtado

Dr. Pedro Gilberto López Mariscal
Coordinador del Posgrado en Ciencias de la Computación

Dra. Ana Denise Re Araujo
Directora de Estudios de Posgrado

Resumen de la tesis que presenta Jesús Antonio Navarrete López como requisito parcial para la obtención del grado de Maestro en Ciencias en Ciencias de la Computación.

Reconocimiento automático de señas dinámicas de la Lengua de Señas Mexicana

Resumen aprobado por:

Dr. Irvin Hussein López Nava
Director de tesis

En México, se estima que 4.2 millones de personas tienen discapacidad auditiva, y cerca del 14 % presenta pérdida total de la audición. Una parte utiliza la Lengua de Señas Mexicana (LSM) como principal medio de comunicación. No obstante, el bajo interés general por aprender esta lengua se refleja en la existencia de apenas 40 intérpretes certificados a nivel nacional. Ante este panorama, las herramientas tecnológicas emergen como un recurso valioso para fortalecer la accesibilidad y ampliar las oportunidades de inclusión. El objetivo de este trabajo fue implementar y evaluar un sistema de traducción automática de la LSM al español, basado en visión por computadora, aprendizaje de máquina y modelos grandes de lenguaje (LLMs). Se emplearon dos conjuntos de datos en video: uno compuesto por señas dinámicas aisladas (glosas) y otro por frases con entre una y cinco glosas. A partir de estos videos se extrajeron puntos clave del cuerpo, manos y rostro mediante técnicas de captura de movimiento. La metodología incluyó dos enfoques: reconocimiento en modalidad aislada y modalidad continua. En la modalidad aislada se evaluaron diversas arquitecturas de aprendizaje profundo, destacando la red ResNet, que alcanzó un F1-score del 92 %. En la modalidad continua se utilizó una estrategia basada en ventanas deslizantes de 32 fotogramas con un traslape del 50 %, cuyas salidas fueron postprocesadas mediante el promediado de inferencias repetidas. Para evaluar el rendimiento se propuso la métrica Presence and Order Penalty Error (POPE), que penaliza errores tanto en la presencia como en el orden de las glosas predichas, obteniendo un error promedio del 37 %. Finalmente, las secuencias de glosas reconocidas en modalidad continua fueron traducidas al español mediante un LLM, analizando casos de éxito y fallo en la fidelidad semántica de las traducciones. El sistema logró realizar traducciones funcionales de LSM al español. Persisten áreas de mejora, como la ampliación del corpus utilizado y el perfeccionamiento de las técnicas de segmentación en la modalidad continua. A pesar de estas limitaciones, los resultados obtenidos constituyen un avance significativo hacia el desarrollo de aplicaciones en tiempo real que fomenten la inclusión social de la comunidad Sorda.

Palabras clave: Lengua de Señas Mexicana (LSM), aprendizaje profundo, mediapipe, openpose, visión por computadora, señas dinámicas, reconocimiento automático de señas dinámicas, reconocimiento continuo de señas, generación de frases, modelos grandes de lenguaje

Abstract of the thesis presented by Jesús Antonio Navarrete López as a partial requirement to obtain the Master of Science degree in Computer Science.

Automatic recognition of dynamic signs of Mexican Sign Language

Abstract approved by:

Dr. Irvin Hussein López Nava
Thesis Director

In Mexico, it is estimated that 4.2 million people have some form of hearing impairment, with approximately 14 % experiencing total hearing loss. A portion of this population uses Mexican Sign Language (LSM) as their primary means of communication. However, the general lack of interest in learning this language is reflected in the fact that there are only around 40 certified interpreters nationwide. In this context, technological tools emerge as a valuable resource to strengthen accessibility and expand opportunities for inclusion. The objective of this work was to implement and evaluate an automatic translation system from LSM to Spanish, based on computer vision, machine learning, and large language models (LLMs). Two video datasets were used: one consisting of isolated dynamic signs (glosses) and another composed of phrases containing between one and five glosses. Keypoints from the body, hands, and face were extracted from these videos using motion capture techniques. The methodology comprised two approaches: recognition in isolated and continuous modalities. In the isolated modality, several deep learning architectures were evaluated, with the ResNet model standing out by achieving an F1-score of 92 %. For the continuous modality, a sliding window strategy was implemented using 32-frame windows with a 50 % overlap, and the outputs were post-processed by averaging repeated inferences. To evaluate performance, the Presence and Order Penalty Error (POPE) metric was proposed, penalizing errors in both the presence and sequence of predicted glosses, yielding an average error of 37 %. Finally, the recognized gloss sequences in the continuous modality were translated into Spanish using an LLM, analyzing both successful and failed cases in terms of semantic fidelity. The system successfully performed functional translations from LSM to Spanish. There are still areas for improvement, such as expanding the dataset and refining segmentation techniques in the continuous modality. Despite these limitations, the results represent a significant step toward the development of real-time applications that promote the social inclusion of the Deaf community.

Keywords: Mexican Sign Language (LSM), deep learning, mediapipe, openpose, computer vision, dynamic signs, automatic recognition of dynamic signs, continuous sign recognition, sentence generation, large language models

Dedicatoria

A la memoria de mi tío Gustavo, quien confió en mí sin dudarlo, y a mi madre María, quien me ha acompañado durante todo el proceso.

Agradecimientos

A la Secretaría de Ciencia, Humanidades, Tecnología e Innovación (SECIHTI, antes CONAHCYT) por el apoyo económico otorgado durante mi estancia en la maestría.

Al Centro de Investigación Científica y de Educación Superior de Ensenada, Baja California (CICESE), por la oportunidad de realizar la maestría y el respaldo brindado durante la estancia.

A la Asociación Regional de Sordos Ensenadenses, en especial al profesor Fernando, quien, gracias a sus clases, me permitió introducirme en la Lengua de Señas Mexicana y adquirir una mayor comprensión de la cultura sorda.

A mi director de tesis, Irvin Hussein López Nava, quien sin duda ha sido como un padre para mí en el ámbito académico y me ha guiado durante todo este trayecto y en la elaboración de mi tesis.

A mi comité de tesis: Jesús Favela Vara, María del Carmen Maya Sánchez e Isabel López Hurtado, por sus observaciones y correcciones.

A Rosa Escobar, por ser esa persona que me acompañó, escuchó y cuidó durante todo este proceso, y por estar ahí para mí en los momentos más difíciles.

A mis padres, por todo el apoyo incondicional que siempre me han otorgado y por darme la certeza de que siempre existirá un lugar seguro a su lado.

A la Dra. Cynthia Pérez, a quien considero como una madre académica, por haberme introducido al mundo de la investigación desde la licenciatura y por seguir acompañando de cerca mi desarrollo.

A mis compañeros de laboratorio: Joan, Eliaf, Johnny y Jesús, a quienes considero como mis hermanos y que siempre estuvieron ahí para intercambiar ideas durante la investigación.

Y a mis amigos: Juan, Jonathan, Ariana, Scarllet y Rachel, por todas las risas, pláticas, noches de estrés y recuerdos inolvidables obtenidos en estos dos años de maestría.

Tabla de contenido

Página

Lista de figuras

Figura Página

Figura Página

Lista de tablas

Capítulo 1. Introducción

1.1. Motivación

A nivel mundial, más de 1.5 mil millones de personas viven con algún grado de pérdida auditiva, de las cuales la incidencia aumenta con la edad.[1]. En México, según datos del INEGI, en el año 2020, se estimó que aproximadamente 4.2 millones de personas tienen discapacidad auditiva, representando alrededor del 3.3 % de la población total del país[2]. Este dato incluye desde personas con una limitación auditiva leve, hasta personas con pérdida total de la audición (sordas), representando estas últimas un estimado de más de 600 mil personas, que es equivalente a un 14 %. De esta población, una parte, utiliza la Lengua de Señas Mexicana (LSM) como su principal medio de comunicación, quienes integran la comunidad Sorda[3]. Es importante recalcar que las personas con discapacidad auditiva, presentan múltiples dificultades en su desarrollo educativo, profesional y humano, por lo cual se ven limitadas sus oportunidades de inclusión.

Conocer con certeza la cantidad de usuarios que utilizan la LSM es algo complicado, se ha estimado que hay una población de entre 87,000 y 100,000 usuarios (Cruz Aldrete & Serrano, 2018). Esto es así debido a que en el censo del INEGI, no se pregunta puntualmente sobre el uso de la lengua de señas. Por otro lado, este número es sin tomar en cuenta otras lenguas de señas que puedan producirse dentro de las comunidades indígenas. Dentro de la propia LSM existen variantes de acuerdo con la lengua a la que tuvieron contacto en un principio. Un ejemplo de esto es la variante de Tijuana, Baja California, donde la comunidad Sorda de esta ciudad tuvo influencias directas de la Lengua de Señas Americana. Fue si no hasta un tiempo después que se tuvo contacto con la comunidad Sorda de Ciudad de México, que la comunidad logró diferenciar estas variantes.

Existe una gran falta de interés en aprender Lengua de Señas Mexicana (LSM) por parte de la comunidad en general. En diciembre de 2021, el Congreso del Estado exhortó al INEGI a realizar un muestreo sobre las personas que dominan la LSM, estimando que existen únicamente 40 intérpretes certificados en todo el país. Esta situación refleja que el uso de la LSM se encuentra mayormente limitado a las personas Sordas y a sus interlocutores más cercanos, como se menciona en Solis et al. (2015); es decir, la población en general no muestra interés por aprenderla hasta que entra en contacto directo con una persona Sorda.

[1] Organización Mundial de la Salud (WHO): Deafness and hearing loss.

[2] Instituto Nacional de Estadística y Geografía (INEGI): Estadísticas sobre Discapacidad.

[3] Por convención, se utiliza el termino *Sordo, Sorda* para hacer referencia al grupo de personas sordas que reconoce la lengua de señas como una lengua natural. Por otro lado, *sordo* hace referencia a una persona con alguna discapacidad auditiva (Cruz Aldrete, 2008).

Dada la escasa disponibilidad de intérpretes certificados y el limitado interés social por aprender LSM, las herramientas tecnológicas emergen como un recurso valioso para fortalecer la accesibilidad y ampliar las oportunidades de inclusión para las personas Sordas, ampliando así sus oportunidades en distintos ámbitos de la vida cotidiana.

En la actualidad, persisten importantes limitaciones para modelar de manera efectiva el problema de traducción entre la LSM y el español, lo que dificulta una inclusión plena de las personas sordas en la sociedad. A pesar del avance tecnológico, aún no existe una aplicación capaz de realizar esta traducción de forma precisa y funcional (Ramírez Sánchez et al., 2021). En el ámbito de las ciencias de la computación, este desafío ha captado la atención de diversos investigadores, ya que la lengua de señas presenta características únicas al tratarse de un lenguaje viso-gestual, lo que lo distingue radicalmente de los lenguajes orales y escritos convencionales.

Para lograr un hito tan grande como lo es un sistema de traducción que sea bidireccional, es necesario abarcar dos tipos de tareas: el reconocimiento y la producción de lengua de señas. El reconocimiento se refiere al proceso de interpretar las señas realizadas por una persona para traducirlas a un lenguaje oral o escrito. En cambio, la producción implica generar secuencias en lengua de señas, a partir de una entrada en lenguaje oral o escrito. Esta tesis se centra en el reconocimiento de la LSM.

1.2. Antecedentes

Es fundamental comprender el contexto en torno al uso de las lenguas de señas a nivel mundial, especialmente considerando los supuestos que, como personas oyentes, tendemos a asumir. Estos son solo algunos de ellos:

- **La lengua de señas no es un lenguaje universal.** Esta es una creencia común entre personas oyentes, debido al desconocimiento que existe sobre el uso de la lengua de señas en distintas regiones. Es importante señalar que cada país —e incluso algunas comunidades dentro del mismo país— tiene su propia lengua de señas.
- **No existe una correspondencia directa entre la LSM y el español.** Aunque se han realizado esfuerzos por establecer equivalencias entre ambas lenguas, la Lengua de Señas Mexicana difiere considerablemente del español. La cantidad de señas en la LSM es mucho menor en comparación

con el número de palabras del español, lo que demuestra que no existe una relación uno a uno entre ambos sistemas lingüísticos.

Con el fin de proponer soluciones específicas para el reconocimiento automático de señas, es esencial comprender los distintos componentes que conforman la problemática a abordar. Por ello, a continuación se describen en detalle los tipos de señas existentes, así como los distintos dominios en los que puede desarrollarse su reconocimiento.

Tipos de señas. Dentro de la tarea de reconocimiento automático de la lengua de señas, se han identificado dos tipos de señas cuyo componente esta centrado en el movimiento; señas estáticas y señas dinámicas. Las señas estáticas cuyo nombre lo dice, son aquellas que no implican algún movimiento, e. g. una letra del alfabeto como la M, por otro lado las señas dinámicas son aquellas que para realizarse necesitan seguir una trayectoria con un principio y un fin determinado, e.g. la misma letra M llevada hacia la boca para señar la palabra MAMÁ.

Dominios de reconocimiento. Existen dos tipos de dominio en los que se puede realizar el reconocimiento automático de la lengua de señas: aislado y continuo. En el dominio aislado se tiene que identificar una única seña dentro de una muestra (generalmente una imagen), presentándose como una tarea de clasificación, i.e., dentro de un conjunto de opciones, se debe seleccionar la más parecida; la cual ha sido abordada desde muchos enfoques que se presentarán más adelante. Por otro lado el dominio continuo, supone un reto mayor, dado que se tienen que reconocer una serie de señas de forma consecutiva dentro de una muestra (generalmente un secuencia de imágenes).

En CICESE se han desarrollado diversos trabajos enfocados al reconocimiento automático de la LSM, abordando distintos tipos de señas y dominios de reconocimiento. Sainos-Vizuett (2022) trabajó con señas dinámicas, en modalidad aislada, mediante un corpus enfocado a palabras en contextos de emergencia médica. Gortarez-Pelayo et al. (2023) se enfocó en el alfabeto de la LSM, incluyendo tanto señas estáticas como dinámicas, pero únicamente en modalidad aislada, como parte del desarrollo de una plataforma educativa. También, Morfín (2023) abordó el reconocimiento de señas estáticas pertenecientes al alfabeto, explorando tanto el dominio aislado como el continuo. Estos trabajos reflejan una evolución progresiva en el estudio del reconocimiento de señas, cubriendo distintos niveles de complejidad y modalidades, y sirven como base para el presente trabajo.

El problema que se abordará en esta tesis es el reconocimiento aislado y continuo de señas dinámicas de

la LSM. Para ser específicos, se reconocerán palabras (también llamadas glosas[4]) para posteriormente generar oraciones a partir de estas señas reconocidas, con el objetivo de lograr una traducción entre la LSM y el español. La meta principal de este trabajo es contribuir al desarrollo de tecnologías inclusivas que reduzcan las barreras de comunicación entre personas Sordas y oyentes, facilitando la interacción en contextos donde actualmente existen obstáculos importantes. Debido a la complejidad del problema, se decidió acotarlo a un contexto médico, de emergencias y frases cotidianas. Este desafío será abordado mediante el uso de técnicas de visión por computadora, aprendizaje automático y modelos grandes de lenguaje (LLMs).

1.3. Propuesta de investigación

1.3.1. Preguntas de investigación

La presente investigación se estructura a partir de dos preguntas fundamentales que orientan el desarrollo del trabajo y permiten delimitar con claridad el alcance del problema. A continuación, se presentan dichas preguntas:

- ¿Qué tan preciso es el uso de modelos de aprendizaje automático entrenados con glosas aisladas de la LSM para realizar reconocimiento continuo de señas dinámicas en videos 2D, dentro de un contexto médico, de emergencias y frases cotidianas?
- ¿Qué tan efectiva es la generación de frases en español utilizando estrategias de posprocesamiento y LLMs, a partir de las glosas reconocidas en el dominio continuo?

1.3.2. Objetivos

Con el propósito de responder a las preguntas de investigación planteadas, esta tesis establece un conjunto de objetivos que guían el desarrollo metodológico del estudio. A continuación, se describen dichos objetivos.

[4]Una glosa es una representación de una seña mediante una palabra en español escrita en mayúsculas.

1.3.2.1. Objetivo general

Implementar y evaluar un sistema basado en visión por computadora, aprendizaje automático y LLMs, capaz de reconocer glosas de la LSM en un contexto médico, de emergencias y frases cotidianas y traducirlas al español.

1.3.2.2. Objetivos específicos

- Implementar y evaluar modelos de reconocimiento de señas dinámicas entrenados con glosas aisladas, utilizando videos 2D en un contexto médico, de emergencias y frases cotidianas.
- Evaluar la precisión del reconocimiento continuo de señas dinámicas a partir de modelos entrenados en glosas aisladas, utilizando videos 2D en un contexto médico, de emergencias y frases cotidianas.
- Diseñar, implementar y evaluar una estrategia de generación de frases en español a partir de glosas reconocidas, incorporando técnicas de posprocesamiento y LLMs.

1.4. Estructura de la tesis

La presente tesis se estructura en seis capítulos. El Capítulo 2 expone los fundamentos teóricos sobre la Lengua de Señas Mexicana (LSM), las herramientas de extracción de características y las arquitecturas de redes neuronales utilizadas, así como las métricas de evaluación aplicadas. El Capítulo 3 presenta una revisión del estado del arte en el reconocimiento automático de señas, proponiendo una taxonomía para clasificar los enfoques existentes. El Capítulo 4 describe la metodología empleada, incluyendo el diseño experimental, la selección del corpus, el preprocesamiento de datos y la implementación de modelos de aprendizaje profundo. El Capítulo 5 reporta los resultados obtenidos en las modalidades aislada y continua, y analiza el uso de modelos grandes de lenguaje (LLMs) para traducir glosas a frases. Finalmente, el Capítulo 6 ofrece una discusión de los hallazgos, expone las limitaciones del estudio y sugiere líneas de trabajo futuro.

Capítulo 2. Marco teórico

En este capítulo se establece el marco teórico que sustenta la investigación, organizado en cuatro secciones principales. La primera aborda los fundamentos lingüísticos de la Lengua de Señas Mexicana (LSM). La segunda presenta los principios del aprendizaje profundo, las arquitecturas especializadas como las redes convolucionales y recurrentes, que constituyen la base de los sistemas modernos para el reconocimiento de lenguas de señas. La tercera sección revisa las técnicas de extracción de características basadas en visión por computadora, con énfasis en la detección de puntos clave (*keypoints*). Finalmente, el capítulo concluye con una revisión de las métricas de evaluación utilizadas en el reconocimiento de señas, diferenciando entre el dominio aislado y el continuo.

2.1. Lengua de Señas Mexicana

Las lenguas de señas son sistemas lingüísticos complejos que se caracterizan por la secuencialidad, espacialidad y simultaneidad, aspectos que no se presentan conjuntamente en las lenguas orales. Para comunicarse, el usuario de la lengua de señas no solo emplea las manos como articuladores activos, sino que también utiliza su cuerpo, expresiones faciales y aprovecha de manera óptima el espacio disponible para señar (Cruz Aldrete, 2018).

Es fundamental destacar que no existe una relación biunívoca entre las lenguas de señas y las lenguas orales, es decir, no hay una correspondencia directa entre una lengua de señas y la lengua oral hablada en la misma región geográfica donde aquella se utiliza. En los primeros estudios sobre lenguas de señas (Signolingüística), se llegó a considerar que estas presentaban una supuesta agramaticalidad, resultado de comparaciones inadecuadas entre lenguas orales y lenguas visuales (Cruz Aldrete, 2008). Sin embargo, esta percepción ha sido superada, reconociéndose actualmente que las lenguas de señas poseen estructuras gramaticales propias y complejas. En particular, la Lengua de Señas Mexicana se compone de varios elementos fundamentales que permiten su estructura y funcionamiento como un lenguaje completo y efectivo. Estos componentes incluyen la modalidad, la gramática, la dactilología, los ideogramas y los diferentes tipos de señas.

2.1.1. Modalidad

La modalidad lingüística hace referencia al canal a través del cual una lengua es percibida y expresada. En el caso de las lenguas orales, esta se basa en una secuencia de sonidos articulados, lo que su modalidad es **auditivo-verbal** y se desarrolla en el tiempo. Por otro lado, la LSM se caracteriza por tener una modalidad **visual-gestual-manual**. Su canal de producción involucra las manos, el rostro y el cuerpo, mientras que su canal de percepción es la vista. Esta lengua se articula tanto en el espacio como en el tiempo. Al estar sustentadas en medios visuales, las lenguas de señas transmiten el significado mediante configuraciones manuales y expresiones faciales y corporales. La articulación de la LSM se desarrolla dentro de un espacio delimitado por tres ejes principales: vertical, horizontal y de proximidad respecto al cuerpo de la persona señante (Escobedo, 2017).

El **límite vertical** comprende el área que va desde la cintura hasta la parte superior de la cabeza, mientras que el **límite horizontal** se extiende lateralmente hasta la altura de los codos con los brazos flexionados. Movimientos que excedan estas dimensiones suelen interpretarse como exageraciones intencionales o recursos expresivos de énfasis (Ver Figura 1).

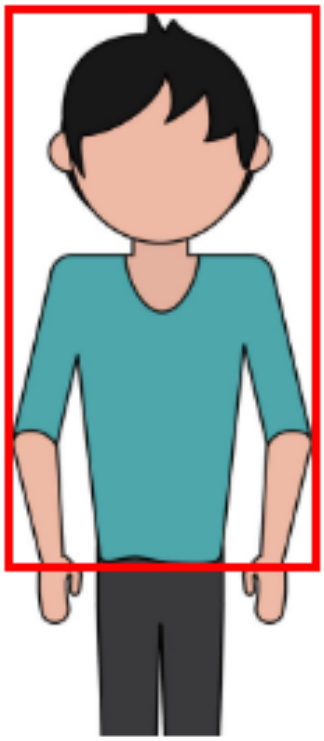


Figura 1. Delimitaciones del espacio señante.

Por otro lado, el **límite de proximidad** hace referencia a la distancia entre las manos y el cuerpo durante la producción de las señas. Esta zona debe mantenerse en un rango cómodo y visualmente accesible. Por ejemplo, señas como **YO** (2(a)) se realizan cerca del pecho, mientras que otras, como **El** (2(b)), pueden extenderse hacia adelante sin perder legibilidad (ver Figura 2).

(a) Seña: **YO** (b) Seña: **EL**

Figura 2. Comparación de límites de proximidad, generada a partir de datos recolectados en Sainos-Vizuett (2022).

2.1.2. Gramática

Una característica distintiva de las lenguas de señas es que son ágrafas, i.e., presentan la ausencia de un sistema de escritura, a diferencia de las lenguas orales. En el ámbito de la investigación lingüística, se recurre al concepto de **glosa** como una herramienta para representar señas utilizando palabras de una lengua diferente, generalmente la lengua dominante del contexto. En el caso de la LSM, se emplean palabras en español (escritas en mayúsculas) para etiquetar cada seña. Este proceso, conocido como glosado, no constituye una traducción literal, sino una estrategia de notación adaptada desde la tradición de las lenguas orales, que permite analizar y documentar las señas de forma sistemática (Herrera & Cantu, 2020).

La fonética, de acuerdo con la Real Academia Española, es la rama de la lingüística encargada del estudio de los sonidos del habla. Su equivalente en el estudio de las lenguas de señas es la querología, disciplina que se enfoca en analizar las señas buscando establecer una analogía con el sistema fonológico de las lenguas orales (González, 1991). Este sistema contempla diversos parámetros quinésicos de formación, de los cuales se abordan tres en este trabajo por considerarse los más representativos:

- **Queirema:** hace referencia a la forma que adopta la mano al producir una seña. Esta configuración incluye aspectos como si la mano está abierta o cerrada, los dedos extendidos o flexionados, así como la posición específica del índice o el pulgar.
- **Toponema:** se refiere al lugar del cuerpo en el que se articula la seña. Este espacio puede situarse frente al cuerpo, la frente, las cejas, los labios, entre otros puntos de referencia.
- **Kinema:** describe el tipo de movimiento que realiza la mano durante la ejecución de la seña. Entre sus variantes se encuentran desplazamientos rectos, circulares o en arco. También abarca componentes quinestésicos como movimientos simples o repetitivos, así como rotaciones del puño o del antebrazo.

En la Figura 3 se observa una representación de los componentes fonéticos de la seña **MI**. Nótese que las flechas de color amarillo indican dichos componentes; la flecha de color rojo representa el recorrido (movimiento) que realiza la seña, y la región sombreada en rojo señala la parte del cuerpo donde se ejecuta. Es importante mencionar que el queirema mostrado en la figura puede ser utilizado en otras señas además de MI, siempre que se combine con un toponema o kinema diferente dentro de la estructura de la seña.

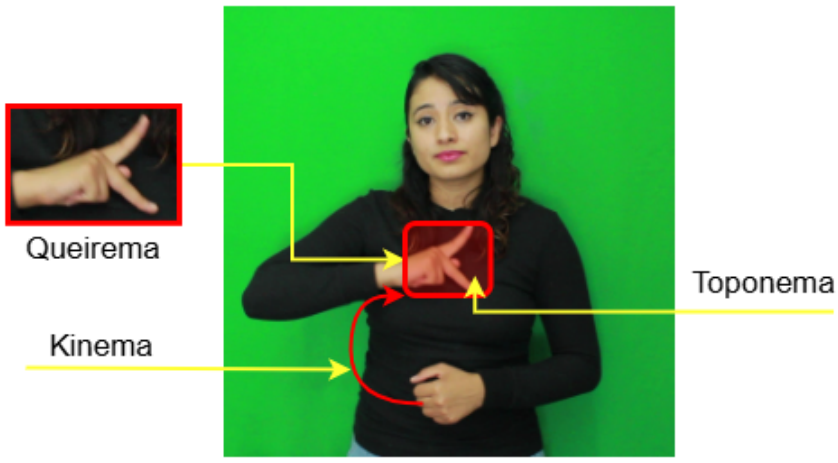


Figura 3. Señalización de la fonética dentro de la seña **MI**.

Se observa que la LSM posee un sistema para la marcación gramatical del tiempo, el cual integra diversos recursos morfológicos y léxicos. Este sistema incluye recursos como la flexión verbal—Cambios en movimiento, orientación, repetición o rasgos no manuales para marcar aspecto, número, etc.—, la combinación de estructuras manuales especificas y el uso de una linea de tiempo como eje organizador del discurso. Asimismo, incorpora formas léxicas adverbiales como: HOY, AYER, MAÑANA, AHORA,

PASADO/ANTES, SIEMPRE, que permiten ubicar un evento dentro de un marco temporal determinado. Estos elementos, utilizados de manera conjunta o independiente, sitúan la acción verbal en una dimensión temporal especifica: presente, pasado (ya sea reciente o remoto) o futuro (próximo o lejano). Un aspecto relevante de este sistema es que, una vez establecida la referencia temporal al inicio de la enunciación, esta se mantiene constante a lo largo del discurso hasta que se indique explícitamente un nuevo cambio de tiempo.

2.1.3. Dactilología

La dactilología consiste en la representación manual de las letras del abecedario en español mediante configuraciones específicas de la mano. Se emplea principalmente para expresar siglas, nombres propios, neologismos o en situaciones en las que no se dispone de una seña convencional establecida. Esta práctica resulta especialmente útil en la comunicación entre personas oyentes que no dominan la Lengua de Señas Mexicana (LSM) y personas sordas, ya que permite establecer un canal de interacción más accesible (Serafín de Fleischmann & González, 2011).

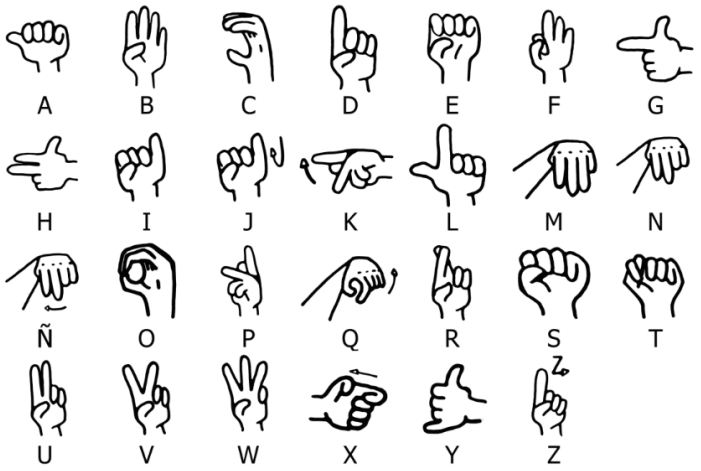
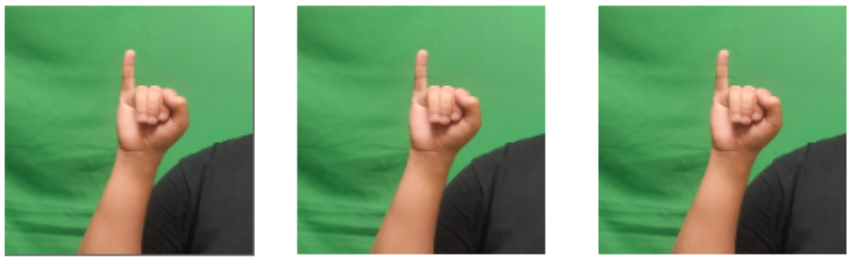


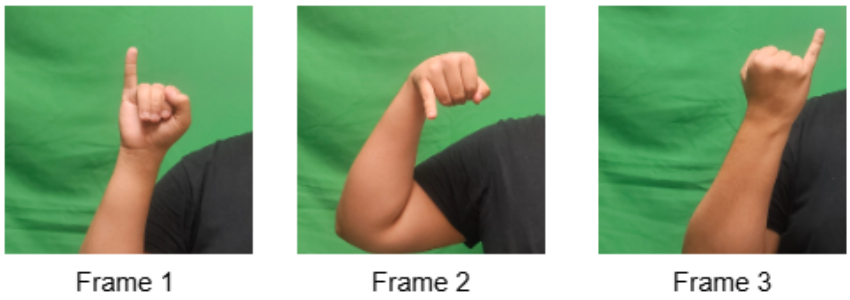
Figura 4. Dactilología completa de la LSM extraída de Gortarez-Pelayo et al. (2023). Note que, algunas configuraciones manuales presentan movimiento, este se representan con flechas que apuntan la dirección del movimiento.

A partir del uso del alfabeto manual, la dactilología permite a las personas sordas representar gráficamente palabras letra por letra utilizando las manos (Ver Figura 4). Esta forma de expresión es particularmente útil en la codificación de señas que inician con la letra correspondiente del término en español, así como en aquellas que derivan directamente de la dactilología. Además, cumple un papel fundamental en la

incorporación de vocabulario nuevo, ya que facilita la transición de términos de la lengua oral a la lengua de señas, reforzando el vínculo entre ambas modalidades lingüísticas y favoreciendo la comprensión en contextos bilingües y educativos. En términos lingüísticos, la dactilología está compuesta por diferentes queiremas que comparten un mismo toponema; no obstante, algunas letras presentan el mismo queirema —como las letras I y J— y se distinguen únicamente por su kinema. En este caso, la letra I no incorpora un movimiento más pronunciado que el que realizaría la letra J, lo que permite diferenciar las señas estáticas de las dinámicas (ver Figura 5).



(a) Secuencia de la letra: **I**



(b) Secuencia de la letra: **J**

Figura 5. Comparación entre la seña estática **I** y la seña dinámica **J**.

2.1.4. Ideogramas

Los ideogramas en la Lengua de Señas Mexicana (LSM) son señas que representan palabras o conceptos completos mediante una o varias configuraciones manuales, acompañadas de expresiones faciales y corporales. Estas señas funcionan como unidades lingüísticas completas, capaces de transmitir significados complejos de forma rápida y directa, sin necesidad de recurrir al uso de la dactilología. En este sentido, cada ideograma actúa como un símbolo visual condensado que comunica una idea específica.

La eficiencia comunicativa que ofrecen los ideogramas los convierte en un elemento clave para la fluidez y coherencia del discurso en LSM. Su integración con el componente gestual y espacial de la lengua permite una representación más rica y dinámica del contenido, lo que los posiciona como una herramienta central en la construcción gramatical y semántica dentro de la comunicación visual de las personas sordas.

Un ejemplo ilustrativo de un ideograma puede observarse en la Figura 6, donde se muestra la seña correspondiente al concepto **ACCIDENTE**. Esta seña combina configuraciones manuales específicas junto con componentes cinéticos que refuerzan su significado.

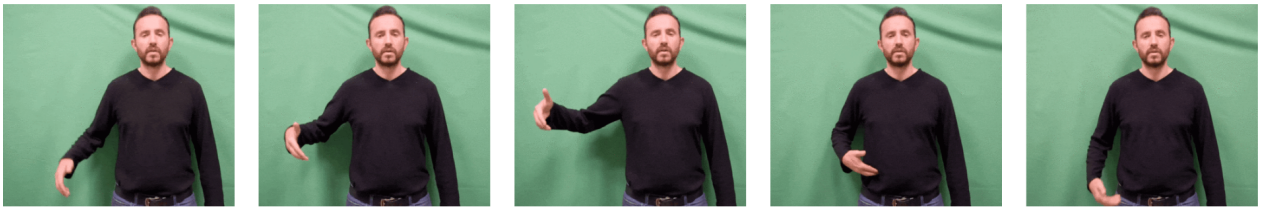


Figura 6. Ejemplo de un ideograma: seña correspondiente al concepto **ACCIDENTE**.

2.2. Extracción de características

A diferencia de otros trabajos centrados en el uso de las imágenes como entrada de los modelos de clasificación, en el presente trabajo se analizará el reconocimiento automático de la LSM desde un enfoque basado en *keypoints*. Un *keypoint* es una característica de alto nivel que representa una unidad elemental de la pose; se trata de un punto específico en la imagen que indica la ubicación de una parte del cuerpo humano. La agrupación de estos *keypoints* permite reconstruir la pose completa de una persona.

En este contexto, los dos *frameworks* más ampliamente utilizados para la estimación de *keypoints* en tareas de reconocimiento automático de señas son **OpenPose** y **MediaPipe**, ya que ambos permiten extraer información detallada sobre la postura corporal, las manos y, en algunos casos, las expresiones faciales, elementos fundamentales para una interpretación precisa de las señas en LSM.

2.2.1. OpenPose

OpenPose es un *framework* de código abierto desarrollado por el *Perceptual Computing Lab* de la Universidad de Carnegie Mellon que implementa en tiempo real el método *Part Affinity Fields* (PAFs)

para estimar la pose 2D de un número arbitrario de personas (Cao et al., 2021). Este *framework* detecta y asocia automáticamente todos los *keypoints* visibles en cuerpo, pies, manos y rostro, dando un total de 135 puntos por persona en una imagen o video, sin necesidad de un detector de personas previo, esto gracias a su estatregia *bottom-up* donde primero localiza partes para posteriormente agruparlas. La mayor parte del tiempo es invertido en una CNN (concepto que se aborda en la siguiente sección) de dos ramas que predice (i) mapas de confianza de partes y (ii) PAFs. Después un algoritmo *greedy* conecta los pares con mayor afinidad para formar el esqueleto.

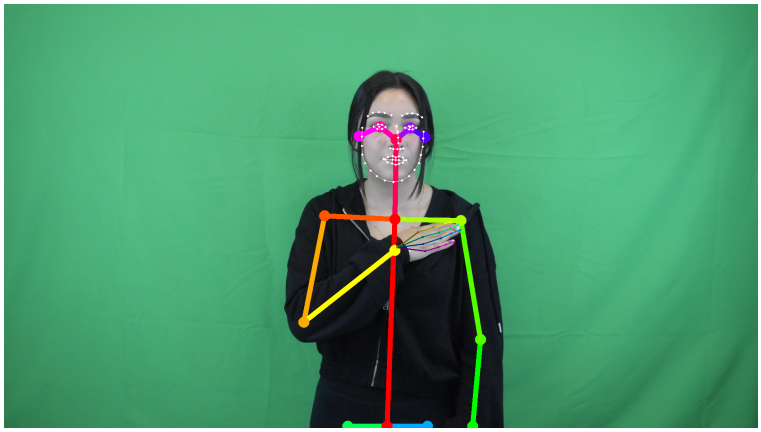


Figura 7. Imagen representativa de la salida obtenida del *framework OpenPose*.

La estructura de los *keypoints* se compone de un triplete (x, y, c), donde (x, y) representa las coordenadas de la proyección 2D del punto en la imagen, y $c \in [0, 1]$ corresponde al nivel de confianza estimado por la red. OpenPose concatena los vectores de todas las regiones detectadas por persona (ver Figura 7); estas se dividen en tres regiones, que son las siguientes:

- **Cuerpo:** Esta región consta de 25 puntos que constan de la nariz, ojos, orejas, hombros, codos, muñecas, caderas, rodillas, tobillos y en los pies son 3 puntos que representan el dedo gordo, quinto dedo y el talón. Cada punto se conecta siguiendo un árbol donde cada conexión (e.g. hombro-codo) dispone de un PAF que codifica la orientación y facilita la asociación por persona.

- **Manos:** La región comprende 21 puntos por mano, donde la información comienza desde la muñeca correspondiente al punto 0 se asignan cuatro articulaciones por cada dedo (pulgar, indice, medio, anular, meñique) y estos siguen una secuencia anatómica de la base a la punta, y los PAFs conectan cada par consecutivo considerando los segmentos anatómicos que son el Metacarpofalángica, Interfalángica Proximal y Interfalángica Distal; esto para codificar la orientación de cada falange y de la mano completa.

- **Rostro:** Se anotan 70 puntos faciales distribuidos del siguiente modo: 17 a lo largo de la mandibula, 10 en las cejas, 9 en la nariz, 12 al rededor de los ojos y 22 en los labios (formando una linea externa y otra interna). Esta densidad permite modelar con precisión expresiones faciales y estimar la orientación de la cabeza. Los PAFs enlazan los puntos vecinos (e.g. comisura-labio superior), preservando la coherencia geométrica incluso con oclusiones parciales o variaciones de iluminación.

2.2.2. MediaPipe

Desarrollado por Google, MediaPipe es un *Framework* multiplataforma de código abierto para el procesamiento de aprendizaje automático. Se especializan en el contexto de visión por computadora y la interacción humano-computadora. Esta herramienta permite a los desarrolladores crear canalizaciones de procesamiento de datos (e.g. videos o audio) utilizando modelos de aprendizaje automático para tareas como la detección de pose, seguimiento de manos, reconocimiento facial, etc.

Para el contexto de este trabajo, nos centraremos específicamente en el modulo de detección holístico (*MediaPipe Holistic*), el cual de manera similar a su contraparte OpenPose, nos permite detectar información de la postura del cuerpo (ver Figura 8). *MediaPipe Holistic* engloba 3 modelos de detección: *Pose*, *Hands* y *Face Mesh*, donde cada modelo se encarga de una región correspondiente.

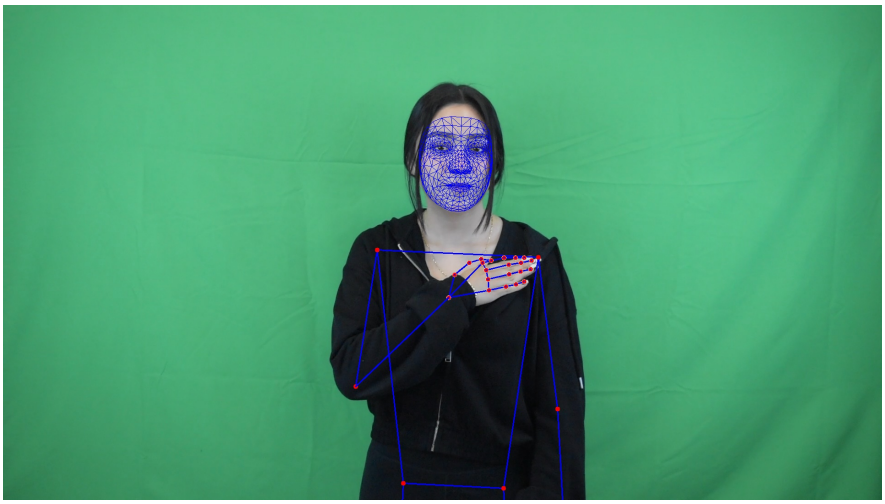


Figura 8. Imagen representativa de la salida obtenida del *framework MediaPipe Holistic*.

Face Mesh: Este modelo se encarga de estimar la malla facial dentro de cada fotograma. El proceso comienza con un detector ligero (*BlazeFace*) que recibe la imagen completa y, mediante una CNN,

localiza el rostro prediciendo un *bounding box* y seis *keypoints* (centros de los ojos, tragus de ambas orejas, punta de la nariz y centro de la boca). Con estos puntos se calcula el ángulo de *roll* y la orientación aproximada del rostro; a partir de ello se genera un recorte rotado y normalizado (i.e., la cara queda centrada y a escala homogénea). Este recorte se introduce en el modelo *Face Mesh*, el cual predice 468 vértices 3D de alta fidelidad. Al operar únicamente sobre el recorte alineado, *Face Mesh* no necesita ser invariante a la escala o a la rotación, lo que le permite dedicar su capacidad a la precisión geométrica. Mientras el rostro permanezca alineado en fotogramas sucesivos, la etapa del detector ligero se omite y se continúa el seguimiento directamente con *Face Mesh*, reduciendo de forma drástica el consumo computacional (Bazarevsky et al., 2019).

Pose: El modelo que se encarga de reconocer la pose se llama BlazePose. En primera instancia, se aplica a cada fotograma completo el modelo *BlazeFace* para obtener ádemas del *bouding box* facial, tres parametros corporales: (i) punto medio entre las caderas, (ii) diametro del circulo que circunscribe al cuerpo y (iii) ángulo de inclinación hombros-caderas. Posteriormente, se crea un recorte cuadrado que está centrado en las caderas, para reducir la variabilidad para despues servir como entrada de un **pose tracker** que predice simultaneamente las coordenadas 2D de 33 *keypoints*, un indicador de visibilidad por cada uno y un ROI (*Region of Interest*) refinado para el siguiente fotograma. Mientras que el tracker indique presencia humana, la parte de detección se omite y se actualiza el ROI mediante la salida del tracker. Estos 33 *keypoints* se distribuyen en el rostro, tronco brazos, piernas y pies (Bazarevsky et al., 2020).

Hands: Para la detección de las manos, se cuenta con un modelo que ligero que solo detecta las palmas llamado *BlacePalm*, este opera sobre el fotograma completo y devuelve una caja orientada que encierra la palma. Esta caja se usa para recortar, rotar y normalizar la mano antes de pasarla al modelo *landmarks* reduciendo la variabilidad con la que trabajar. El modelo *landmarks* regresa de forma simultanea las coordenadas 2.5D (e.i. x, y, z, siendo z una profundidad relativa) de 21 *keypoints*, la probabilidad de presencia de una mano correctamente alineada y su lateralidad (izquierda/derecha). Durante el video, la caja del siguiente fotograma se deriva de los *landmarks* actuales, por lo que el detector solo se ejecuta en el primer fotograma o cuando la confianza cae por debajo de un umbral. Los 21 *keypoints* están distribuidos considerando las articulaciones carpometacarpiana, metacarpofalángica, interfalángicas proximal y yema tomando como origen la muñeca (Zhang et al., 2020).

2.3. Aprendizaje profundo

Para abordar el problema de reconocimiento automático, este trabajo se basa en el uso de arquitecturas de aprendizaje profundo (*deep learning*), una sub-disciplina del **Aprendizaje Automático** que, a su vez, pertenece al campo de la **Inteligencia Artificial**. El aprendizaje profundo emplea redes neuronales artificiales de gran profundidad, compuestas por múltiples capas interconectadas, capaces de extraer representaciones jerárquicas de alto nivel a partir de los datos sin recurrir a la ingeniería manual de atributos. Esta facultad de aprender características relevantes directamente de la información bruta ha demostrado ser especialmente eficaz en tareas de reconocimiento de patrones visuales o secuenciales.

La Figura 9 ilustra de manera representativa las diferencias entre el enfoque tradicional del aprendizaje de máquina y el paradigma del aprendizaje profundo, destacando cómo este último automatiza la extracción de características.

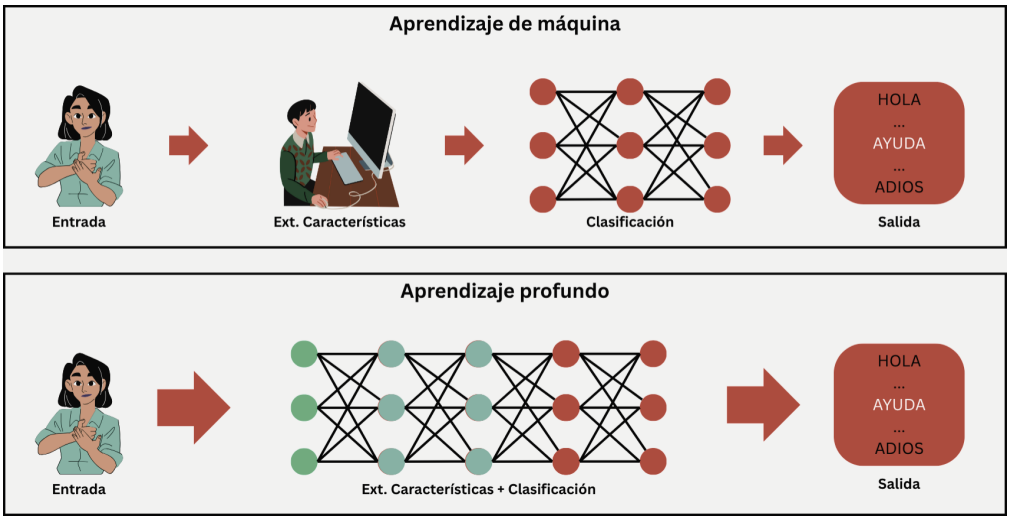


Figura 9. Comparación entre el enfoque tradicional del aprendizaje de máquina y el aprendizaje profundo

Antes de profundizar en las arquitecturas utilizadas, resulta imprescindible situar el tipo de tarea que esta tesis aborda. En este trabajo, las redes neuronales se emplean dentro del marco del **Aprendizaje Supervisado**, una rama del Aprendizaje de Máquina que se caracteriza por el uso de ejemplos etiquetados para guiar el entrenamiento del modelo, indicando de antemano el resultado deseado.

En nuestro caso, la tarea que debe realizar el modelo es de clasificación, ya que el resultado esperado corresponde a la etiqueta asociada a una seña. En una **tarea de clasificación** debe construirse un modelo capaz de asignar cada observación a una de varias clases previamente establecidas. Para ello, se le entrena con un conjunto amplio y representativo de estas observaciones (de ahora en adelante llamados

datos) etiquetadas: el modelo analiza dichos ejemplos, identifica los rasgos que distinguen a cada clase y aprende un conjunto de reglas internas que codifican esos patrones (ver Figura 10). Una vez completado el entrenamiento, aplica esas reglas para etiquetar correctamente nuevas observaciones no vistas durante la fase de entrenamiento, demostrando así su capacidad de generalización. El entrenamiento se rige por un algoritmo de clasificación (en este caso redes neuronales) que actúa como el motor del modelo, determinando como deben separarse los datos, compara cada entrada con los criterios aprendidos y decide la etiqueta que le corresponde para posteriormente evaluar su rendimiento (Belcic, 2024).

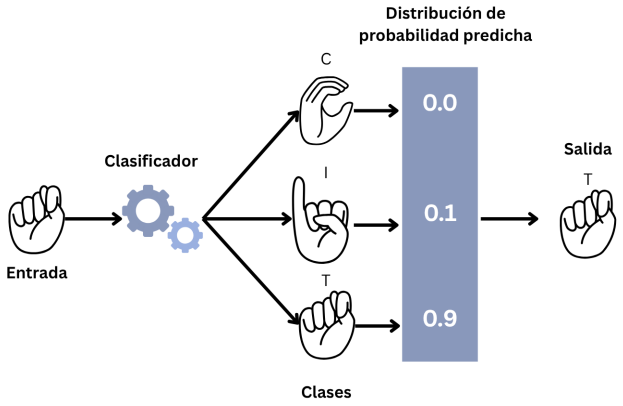


Figura 10. Ejemplo ilustrativo de una tarea de clasificación

2.3.1. Arquitecturas de redes neuronales

Las arquitecturas presentadas en este trabajo corresponden a los modelos utilizados para el reconocimiento de señas. Dichos modelos están diseñados para realizar tareas de clasificación, reconociendo una glosa a partir de una seña dinámica capturada en video. Cabe resaltar que todo el funcionamiento de una red neuronal, desde el perceptron a hasta una red totalmente conectada, se encuentra en el Anexo .1.

2.3.1.1. Redes Neuronales Convolucionales

Las *Convolutional Neural Networks* (CNN) son arquitecturas diseñadas para procesar datos con estructura espacial (e.g. imágenes). En lugar de capas totalmente conectadas, emplean **operaciones de convolución** que explotan la localidad y la invariancia traslacional de los patrones visuales, reduciendo

drásticamente el número de parámetros y mejorando la capacidad de generalización (Zhang et al., 2023).

En una red neuronal convolucional, el **kernel** (o filtro) es un pequeño tensor de pesos $K \in \mathbb{R}^{k_h \times k_w}$ que se desliza sobre la entrada realizando multiplicaciones elemento a elemento, generando un mapa de activaciones. Estos pesos se comparten en todas las posiciones, lo cual introduce un fuerte sesgo inductivo de *equivarianza traslacional*. Las entradas pueden tener múltiples **canales**, como ocurre en imágenes RGB, y las capas intermedias pueden producir decenas o cientos de canales de características. Al aplicar C_{out} filtros distintos sobre una entrada con C_{in} canales, se obtiene un tensor de salida de forma (C_{out}, H', W'), lo que permite a la red aprender diversos detectores de patrones.

Para controlar el tamaño espacial de la salida, se emplea **padding**, que consiste en añadir filas y columnas de ceros alrededor de los bordes, lo cual también ayuda a preservar la información perimetral. Por su parte, el **stride** s indica el paso con el que el kernel se desplaza; valores mayores a uno reducen la resolución espacial y actúan como un muestreo aprendido. Complementariamente, las capas de **pooling** (ya sea máximo o promedio) sustituyen regiones locales por un único valor representativo, reduciendo la dimensionalidad y agregando invariancia ante pequeñas traslaciones, lo que además ayuda a prevenir el sobreajuste. Finalmente, la **operación de convolución** en el dominio discreto se define como una suma ponderada de productos locales entre la entrada $\mathbf{x}$ y el kernel $\mathbf{k}$, siendo la base matemática que permite a la red extraer características jerárquicas de la entrada. Para una señal unidimensional $\mathbf{x} \in \mathbb{R}^N$ y un kernel $\mathbf{k} \in \mathbb{R}^r$:

$$(\mathbf{x} * \mathbf{k})[t] \;=\; \sum_{i=0}^{r-1} x_{\,t+i}\, k_{\,i}, \qquad t = 0, \ldots, N - r. \tag{1}$$

En el caso bidimensional (imágenes), con una imagen $\mathbf{X} \in \mathbb{R}^{H \times W}$ y un kernel $\mathbf{K} \in \mathbb{R}^{k_h \times k_w}$, la convolución en la posición (u, v) se expresa como

$$(\mathbf{X} * \mathbf{K})_{u,v} \;=\; \sum_{i=0}^{k_h-1} \sum_{j=0}^{k_w-1} X_{u+i,\,v+j}\; K_{i,j}, \qquad u = 0, \ldots, H - k_h,\; v = 0, \ldots, W - k_w. \tag{2}$$

Cada valor de la salida se obtiene alineando el kernel con la región local de la entrada y sumando los productos elemento-a-elemento. En redes neuronales, este operador se aplica **con pesos compartidos**: los mismos coeficientes $K_{i,j}$ se utilizan en todas las posiciones, lo que impone equivarianza traslacional y reduce el número de parámetros aprendibles. En la Figura 11 se observa de manera gráfica este operación[1].

[1] Imagen extraída de: El concepto de la convolución en gráficos.

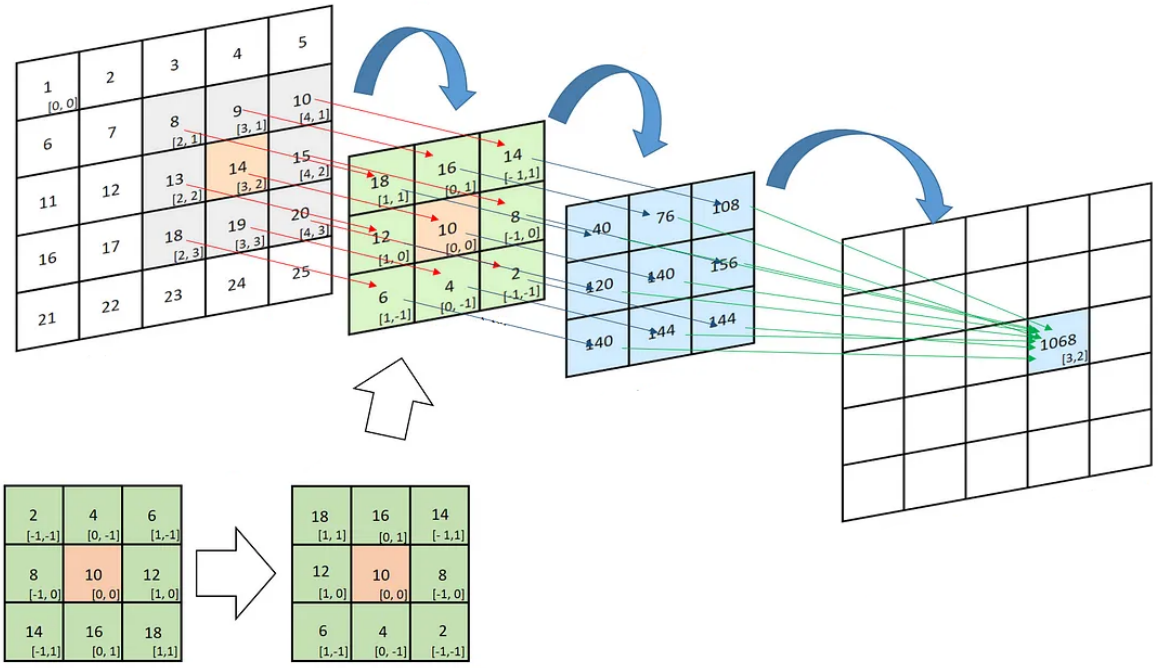


Figura 11. Imagen ilustrativa del proceso de convolución.

Residual Networks (ResNet). Las **ResNet** introducen un cambio paradigmático al incorporar **conexiones de identidad** que permiten entrenar modelos muy profundos sin que el gradiente se desvanezca. A grandes rasgos, una ResNet conserva la filosofía de VGG de usar exclusivamente convoluciones 3×3, pero las organiza en **módulos residuales** y agrupa dichos módulos en cuatro bloques con número creciente de canales ($64\to128\to256\to512$). Después de un par de convoluciones iniciales y un *max-pooling*, cada bloque residual reduce la resolución espacial con un stride de 2, mientras duplica los canales. Finalizada la extracción de características, una *average pooling* global y una capa totalmente conectada generan la predicción. El núcleo de ResNet es el *residual block*, compuesto por dos convoluciones 3×3 seguidas de BatchNorm y ReLU (ver Figura 12. En lugar de aprender directamente la transformación $F(\mathbf{x})$, el bloque aprende una **función residual** $G(\mathbf{x})$ y suma la entrada mediante un atajo de identidad:

$$\mathbf{y} = F(\mathbf{x}) + \mathbf{x}, \qquad F(\mathbf{x}) = \mathrm{ReLU}\big(\mathrm{BN}(\mathrm{Conv}_{3\times3}(\mathrm{ReLU}(\mathrm{BN}(\mathrm{Conv}_{3\times3}(\mathbf{x})))))\big). \tag{3}$$

Cuando el número de canales o la dimensión espacial cambia, el atajo se ajusta con una proyección 1×1. Este simple esquema facilita un flujo de gradiente sin obstáculos, haciendo viable el entrenamiento de redes de 50, 101 o incluso 152 capas, y se ha convertido en un componente básico de los modelos de visión profunda modernos.

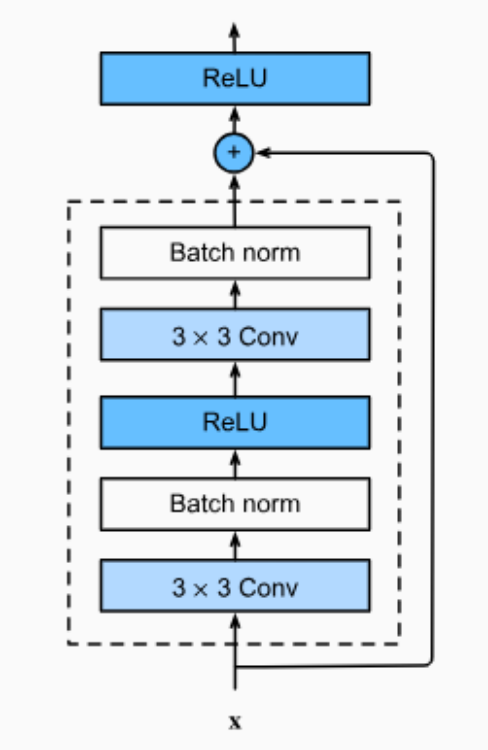


Figura 12. Esquema de un bloque residual de una ResNet. Tomada de Zhang et al. (2023)

2.3.1.2. Redes Neuronales Recurrentes (RNN)

Una *Recurrent Neural Network* incorpora cómputo recurrente sobre un **estado oculto** para modelar datos secuenciales. En cada paso temporal t procesa la entrada $\mathbf{x}_t$ junto con el estado previo $\mathbf{x}_{t-1}$:

$$\mathbf{h}_t = f\big(W_{xh}\mathbf{x}_t + W_{hh}\mathbf{h}_{t-1} + \mathbf{b}_h\big), \qquad \mathbf{y}_t = g\big(W_{hy}\mathbf{h}_t + \mathbf{b}_y\big), \tag{4}$$

donde f es una activación (e.g. $tanh$), g la función de salida y W_{xh}, W_{hh}, W_{hy} son las matrices de pesos compartidas a lo largo del tiempo. El vector $\mathbf{h}_t$ actúa como **memoria** de la secuencia hasta el instante actual, por lo que el número total de parámetros no crece con la longitud de la secuencia (Zhang et al., 2023) (Ver Figura 13).

En síntesis, una RNN introduce un bucle de retroalimentación que le permite modelar la dinámica temporal con un conjunto compacto de parámetros, siendo una herramienta fundamental en lenguaje natural, predicción de series temporales y otras tareas secuenciales.

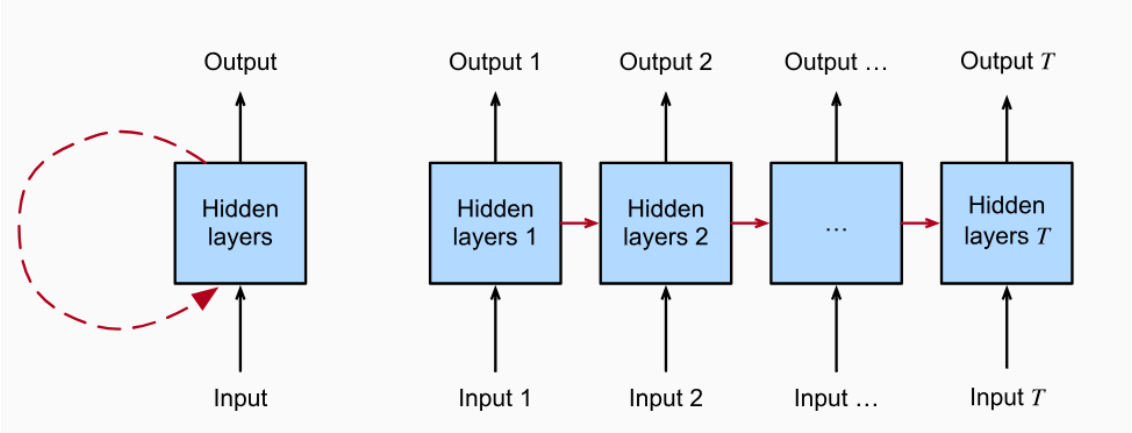


Figura 13. Esquema de una RNN. Tomada de Zhang et al. (2023).

Long Short-Term Memory (LSTM). Es una variante de RNN que añade un **estado de memoria** $\mathbf{C}_t$ y tres compuertas sigmoides para controlar de forma diferenciable el flujo de la información a lo largo del tiempo. Dados la entrada $\mathbf{X}_t \in \mathbb{R}^{n\times d}$ y el estado oculto previo $\mathbf{H}_{t-1} \in \mathbb{R}^{n\times h}$, las compuertas se calculan como

$$\begin{aligned}
\mathbf{I}_t &= \sigma\big(W_{xi}\mathbf{X}_t + W_{hi}\mathbf{H}_{t-1} + \mathbf{b}_i\big),\\
\mathbf{F}_t &= \sigma\big(W_{xf}\mathbf{X}_t + W_{hf}\mathbf{H}_{t-1} + \mathbf{b}_f\big),\\
\mathbf{O}_t &= \sigma\big(W_{xo}\mathbf{X}_t + W_{ho}\mathbf{H}_{t-1} + \mathbf{b}_o\big),\\
\tilde{\mathbf{C}}_t &= \tanh\big(W_{xc}\mathbf{X}_t + W_{hc}\mathbf{H}_{t-1} + \mathbf{b}_c\big).
\end{aligned} \tag{5}$$

El *forget gate* $\mathbf{F}_t$ decide cuánto conservar de la memoria anterior, el *input gate* $\mathbf{I}_t$ cuánto incorporar de la nueva información $\tilde{\mathbf{C}}_t$ y el *output gate* $\mathbf{O}_t$ cuánto revelar al estado oculto de salida. Las actualizaciones son

$$\mathbf{C}_t = \mathbf{F}_t \odot \mathbf{C}_{t-1} \;+\; \mathbf{I}_t \odot \tilde{\mathbf{C}}_t, \qquad \mathbf{H}_t = \mathbf{O}_t \odot \tanh(\mathbf{C}_t), \tag{6}$$

donde $\odot$ denota el producto elemento a elemento. Al separar explícitamente la **memoria de largo plazo** $\mathbf{C}_t$ del estado oculto $\mathbf{H}_t$, LSTM atenúa el problema del desvanecimiento del gradiente, permitiendo capturar dependencias temporales largas en tareas como traducción automática, modelado de texto o series temporales.

Gated Recurrent Unit (GRU). Es otra variante de las RNN que, al igual que las LSTM, emplea compuertas sigmoides para controlar el flujo de información, pero reduce su número a dos: la *compuerta de actualización* y la **compuerta de reinicio**. Dadas la entrada $\mathbf{x}_t$ y el estado oculto previo $\mathbf{h}_{t-1}$, se

definen

$$\begin{aligned}
\mathbf{z}_t &= \sigma\big(W_{xz}\mathbf{x}_t + W_{hz}\mathbf{h}_{t-1} + \mathbf{b}_z\big), && \text{(update gate)}\\
\mathbf{r}_t &= \sigma\big(W_{xr}\mathbf{x}_t + W_{hr}\mathbf{h}_{t-1} + \mathbf{b}_r\big), && \text{(reset gate)}\\
\tilde{\mathbf{h}}_t &= \tanh\big(W_{xh}\mathbf{x}_t + W_{hh}(\mathbf{r}_t \odot \mathbf{h}_{t-1}) + \mathbf{b}_h\big),\\
\mathbf{h}_t &= \mathbf{z}_t \odot \mathbf{h}_{t-1} \;+\; (1-\mathbf{z}_t) \odot \tilde{\mathbf{h}}_t,
\end{aligned} \tag{7}$$

donde σ es la sigmoide y $\odot$ el producto elemento a elemento.

La compuerta de reinicio $\mathbf{r}_t$ decide cuánto del estado pasado debe ignorarse al generar la propuesta $\tilde{\mathbf{h}}_t$; la compuerta de actualización $\mathbf{z}_t$ determina la mezcla convexa entre el estado anterior y la nueva propuesta, permitiendo copiar o sobre-escribir información. Al carecer de una memoria separada como las LSTM, el GRU posee menos parámetros y suele entrenar más rápido, manteniendo una capacidad comparable para capturar dependencias temporales largas en tareas de modelado de secuencias.

2.4. Métricas de evaluación

Para evaluar el rendimiento de los modelos en modalidad aislada, se emplean métricas clásicas de clasificación en aprendizaje de máquina, las cuales se derivan de la matriz de confusión. En contraste, cuando se evalúa el desempeño en modalidad continua —donde el objetivo final es la generación de una frase en español a partir de una secuencia de glosas reconocidas— es necesario comparar cadenas de texto. En este caso, se utilizan métricas específicas de evaluación de secuencias, como la distancia de edición (*edit distance*) o similares, que permiten cuantificar la similitud entre la frase generada y una frase de referencia.

2.4.1. Métricas para la evaluación aislada

Cuando se evalúa un clasificador a partir de su *matriz de confusión*, donde *TP*, *TN*, *FP* y *FN* representan los conteos de verdaderos positivos, verdaderos negativos, falsos positivos y falsos negativos, respectivamente; se emplean las siguientes métricas:

Exactitud (Accuracy). Mide la proporción global de predicciones correctas:

$$\textit{Accuracy} \;=\; \frac{\mathrm{TP}+\mathrm{TN}}{\mathrm{TP}+\mathrm{TN}+\mathrm{FP}+\mathrm{FN}}. \tag{8}$$

Aunque intuitiva, puede resultar engañosa en problemas con clases desbalanceadas.

Sensibilidad (Recall). También llamada *tasa de verdaderos positivos*; cuantifica la capacidad del modelo para detectar la clase positiva:

$$\textit{Recall} \;=\; \frac{\mathrm{TP}}{\mathrm{TP}+\mathrm{FN}}. \tag{9}$$

Precisión (Precision). La precisión mide la proporción de verdaderos positivos entre todas las predicciones positivas realizadas por el modelo:

$$\textit{Precision} = \frac{\mathrm{TP}}{\mathrm{TP}+\mathrm{FP}}. \tag{10}$$

Especificidad. Complemento de la sensibilidad, refleja la proporción de verdaderos negativos identificados correctamente:

$$\text{Especificidad} \;=\; \frac{\mathrm{TN}}{\mathrm{TN}+\mathrm{FP}}. \tag{11}$$

F1-Score. Es la media armónica entre *precision* y *recall*, equilibrando exactitud y cobertura:

$$\mathrm{F1} \;=\; 2\,\frac{\textit{Precision}\times\mathrm{Recall}}{\textit{Precision}+\mathrm{Recall}}. \tag{12}$$

Resulta útil cuando existe desbalance de clases o se desea ponderar uniformemente errores de omisión y comisión.

2.4.2. Métricas para la evaluación continua

En tareas de reconocimiento de continuo se requiere comparar cadenas predichas y de referencia:

Distancia de Levenshtein. Conocida como *edit distance*, es el número mínimo de operaciones de *inserción* (I), *eliminación* (D) y *sustitución* (S) necesarias para transformar la secuencia predicha en la correcta. Se calcula mediante programación dinámica y se denota $\mathrm{ED}(\hat{\mathbf{y}},\mathbf{y}) = S + I + D$ Levenshtein

(1966).

Word Error Rate (WER). Normaliza la distancia de Levenshtein por la longitud de la secuencia de referencia:

$$\text{WER} = \frac{S + I + D}{N}, \tag{13}$$

donde N es el número total de palabras de la referencia. Valores cercanos a 0 indican pocas ediciones necesarias, mientras que 1 denota que cada término debe modificarse (Morris et al., 2004).

2.4.2.1. Ejemplo de evaluación con Distancia de Levenshtein y Word Error Rate

Frase de referencia (correcta): `VOY AL HOSPITAL POR UN ACCIDENTE`
Frase predicha por el sistema: `VOY HOSPITAL POR ACCIDENTE`

- Número total de palabras en la referencia: $N = 6$
- Operaciones necesarias para transformar la predicción en la referencia:
  - Inserción de la palabra `AL`
  - Inserción de la palabra `UN`
- No se requieren eliminaciones ni sustituciones.

Distancia de Levenshtein (ED): $\text{ED}(\hat{\mathbf{y}}, \mathbf{y}) = S + I + D = 0 + 2 + 0 = 2$

Word Error Rate (WER): $\text{WER} = \frac{S+I+D}{N} = \frac{2}{6} \approx 0.33$

Este resultado indica que fue necesario modificar aproximadamente un 33 % de la frase predicha para que coincidiera con la frase correcta.

La revisión teórica presentada ofrece una visión integral de los elementos que convergen en el reconocimiento automático de la LSM. En primer término, se describió la naturaleza lingüística y visual de la lengua de señas, justificando la necesidad de herramientas capaces de procesar información espacial y temporal. Luego se identificó la extracción de características como etapa intermedia indispensable para traducir el movimiento humano en representaciones numéricas robustas. Sobre esta base, el aprendizaje

profundo aporta modelos altamente expresivos (CNN para patrones espaciales y RNN para dependencias temporales) cuyo entrenamiento descansa en funciones de pérdida adecuadas y optimizadores eficientes. Finalmente, se definieron métricas objetivas que permiten comparar de manera rigurosa las distintas aproximaciones experimentales.

Capítulo 3. Trabajo relacionado

En este capítulo se examina el estado del arte del reconocimiento automático de la LSM durante la última década, complementado con una breve revisión de trabajos representativos en otras lenguas de señas. A partir del análisis de la literatura enfocada exclusivamente en la LSM, se propone una taxonomía que clasifica los estudios según cuatro criterios principales: el tipo de seña analizada, ya sea estática o dinámica; la tarea abordada, que puede centrarse en la dactilología o en el reconocimiento de ideogramas; la modalidad de reconocimiento, que se divide en aislada o continua; y las estrategias de extracción de características empleadas, las cuales incluyen procesamiento basado en señales, imágenes y *keypoints*. Esta taxonomía, ilustrada en la Figura 14, ofrece un marco unificado para comparar enfoques, identificar áreas de oportunidad y destacar tendencias emergentes.

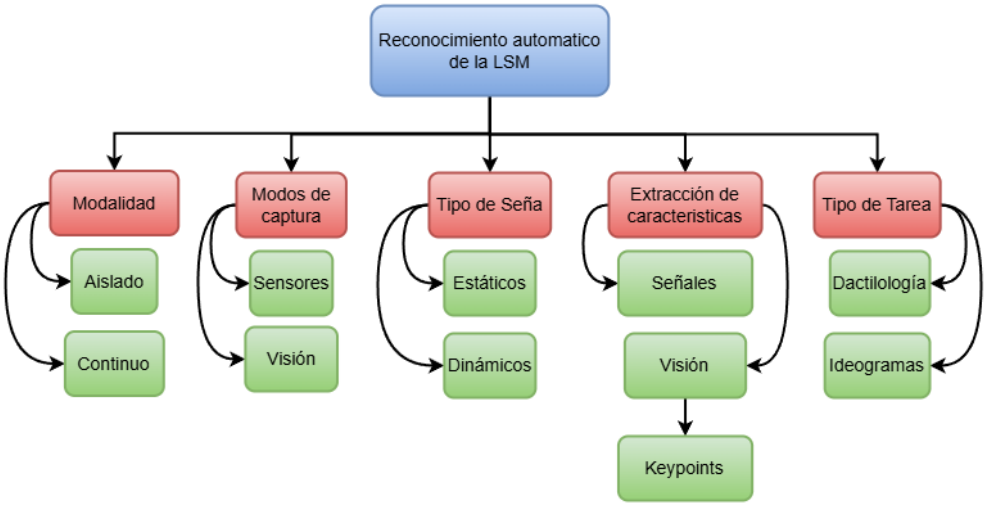


Figura 14. Taxonomía utilizada para clasificar el trabajo relacionado.

3.1. Metodología de revisión

Es importante resaltar los avances realizados específicamente en el contexto de la LSM, con el objetivo de identificar las fortalezas y debilidades que presenta el estado del arte en el reconocimiento automático de esta lengua en particular. La contribución de este trabajo está orientada a la comunidad Sorda en México, por lo que resulta fundamental reconocer y valorar los esfuerzos tecnológicos desarrollados para esta lengua en particular. Por esta razón, se llevó a cabo una búsqueda exhaustiva de investigaciones centradas en el reconocimiento automático de la LSM. No obstante, considerando que la metodología y los tipos de tareas abordadas en otras lenguas de señas pueden ofrecer aprendizajes transferibles, se

incluye también una revisión selectiva de trabajos representativos a nivel internacional, lo que permite establecer puntos de comparación y fortalecer el enfoque metodológico adoptado en esta tesis.

3.1.1. Revisión enfocado a LSM

Para llevar a cabo la búsqueda de la literatura, se realizó una búsqueda con dos bases de datos bibliográficas y un motor de búsqueda, estos fueron: *Scopus, ACM y Semantic Scholar*, este último fue seleccionado para tener una mayor cobertura de trabajos. En todos los casos, se aplicó la siguiente cadena de búsqueda: **(“mexican sign language”) AND (“recognition” OR “translation”)**.

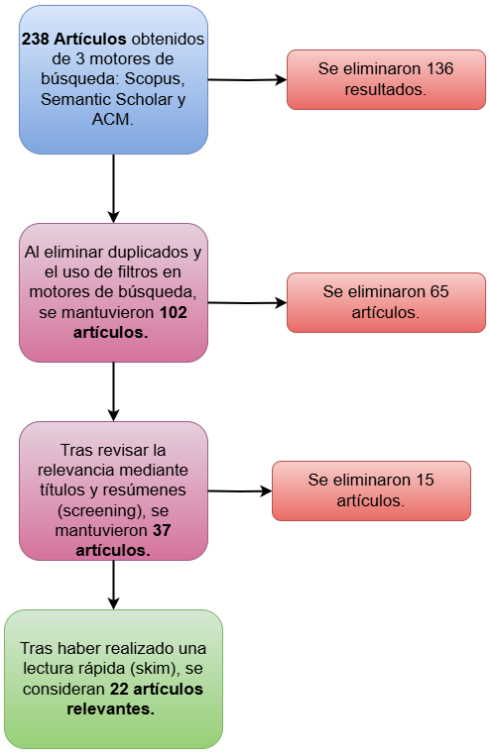


Figura 15. Diagrama de inclusión y exclusión de literatura.

Para la identificación de literatura relevante, se llevó a cabo un proceso de filtrado en todos los casos, el cual consistió en delimitar el rango de años de publicación (2014–2025), seleccionar únicamente artículos de conferencia y revista cuya contribución se enmarque dentro del área de ciencias de la computación, y excluyendo artículos de revisión de literatura. Posteriormente se llevó a cabo una etapa de cribado y lectura rápida para descartar la literatura menos relevante o que no se ajustara a los objetivos del estudio. Este procedimiento se ilustra en la Figura 15.

3.1.2. Revisión en otras lenguas de señas

Actualmente, realizar una revisión exhaustiva sobre el reconocimiento automático en todas las lenguas de señas constituye una tarea de gran escala que excede el alcance y el tiempo disponible para el desarrollo de esta tesis. No obstante, al utilizar la siguiente cadena de búsqueda en *Scopus*: ("sign language") AND (recognition.ºR "translation"), fue posible obtener una visión general de los resultados y detectar las regiones donde se concentra la mayor parte de la investigación en este campo. Posteriormente, se realizaron búsquedas específicas para distintas lenguas de señas, empleando la siguiente cadena: («¡País origen¿¿ sign language") AND (recognition.ºR "translation"). A partir de este análisis, se identificó que la lengua de señas con mayor número de publicaciones es la Lengua de Señas Americana (ASL), seguida por las lenguas de señas utilizadas en India y China (ver Figura 16).

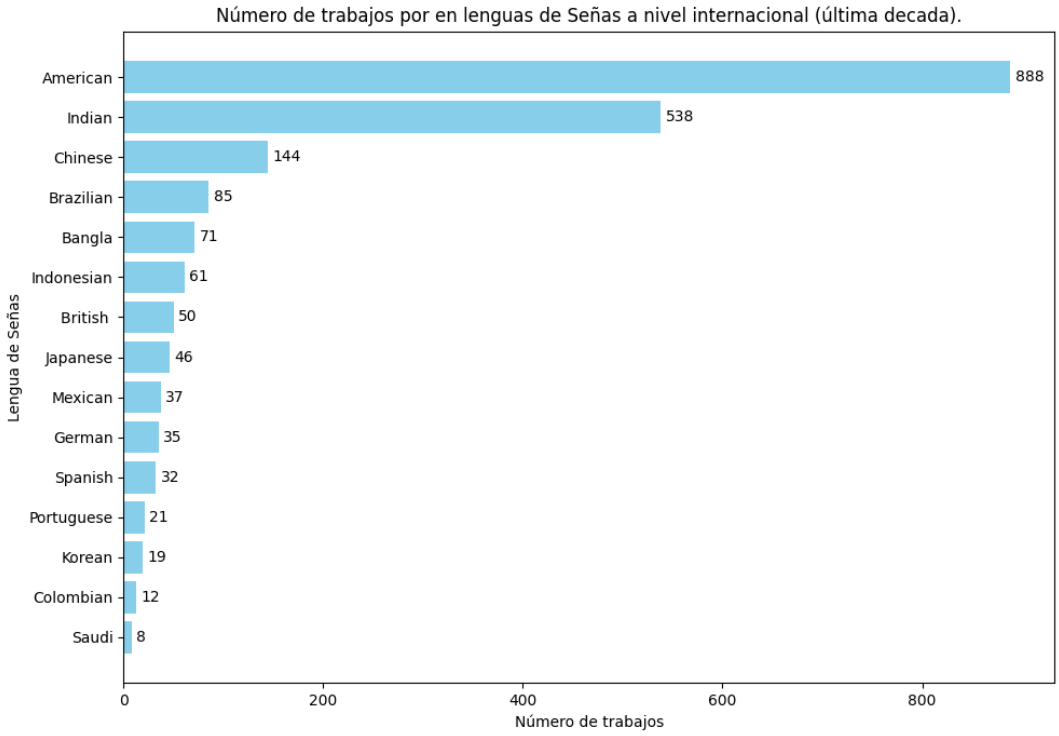


Figura 16. Numero de publicaciones por lengua de señas (resultados de *scopus*).

Por otro lado, el trabajo realizado por Kumar Attar et al. (2023), en donde se realiza una revisión sistemática acerca del estado del arte en la automatización de la lengua de señas, reportan que India es el país con más artículos seleccionados, además de ser el país con mayor número de usuarios estimados en su respectiva lengua de señas (6,815,000 usuarios) lo que da pie a inferir el por qué este país está muy interesado en automatizar la traducción de su lengua de señas.

Con el objetivo de contrastar de manera más clara las contribuciones realizadas en otras lenguas de señas, se seleccionaron artículos con mayor relevancia (de acuerdo a *scopus*) de los últimos dos años (2023–2025) que abordan la Lengua de Señas India (ISL), la Lengua de Señas China (CSL) y la Lengua de Señas Americana (ASL). Estos trabajos se analizaron en función de la taxonomía presentada en la Figura 14, lo que permitió establecer una comparación directa con los enfoques desarrollados para la LSM. La revisión incluyó un total de ocho artículos: tres sobre ISL, dos sobre CSL y tres sobre ASL.

3.2. Estado del arte en Reconocimiento de Lengua de Señas

3.2.1. Tipo de seña

Las señas estáticas se caracterizan por una única configuración manual capturada en una imagen, mientras que las señas dinámicas implican una secuencia de movimientos que requieren análisis espacial y temporal. Esta diferencia determina el tipo de datos utilizado (imagen o video) y las técnicas de procesamiento necesarias en los modelos de reconocimiento.

3.2.1.1. Señas estáticas

En el trabajo de Solis et al. (2015) se utilizó un corpus de 24 gestos estáticos de la dactilología para extraer características a partir de Momentos de *Jacobi-Fourier*, obteniendo una exactitud del 95 %. De manera similar, Pérez et al. (2017) utilizaron un corpus compuesto por 21 gestos estáticos, aplicando Fuzzy C-means y momentos de Hu para la segmentación y extracción de características, con lo cual obtuvieron una exactitud del 91 %.

Se observó que los trabajos donde se utilizan gestos estáticos están relacionados con la dactilología o números, como es el caso de Jimenez et al. (2017); Alejandro & Antonio (2021); Rios-Figueroa et al. (2022); Morfín-Chávez et al. (2024). El trabajo de Estrivero-Chavez et al. (2019) se destaca por el uso de un corpus compuesto por un grupo selecto de letras de la dactilología y palabras que pueden ser representadas con un gesto estático (e.g., AMOR, NEGRO, COMER, BLANCO, FAMILIA, etc.), obteniendo una exactitud de 99.80 %.

En el trabajo de Alsharif et al. (2023) exploran la clasificación señas estáticas utilizando el alfabeto de la ASL utilizando un corpus encontrado en *IEEE Dataport*. A pesar de mencionar que dentro de su alfabeto existen señas dinámicas, este trabajo presenta un enfoque exclusivo a señas estáticas dado a que las configuraciones manuales presentan variaciones distinguibles sin necesidad de movimiento. Del mismo modo para el trabajo de Kothadiya et al. (2023), donde utilizan un conjunto de datos de ISL que incluye 36 señas con un total de 1000 imagenes por clase.

3.2.1.2. Señas dinámicas

Se identificaron trabajos enfocados en la detección de gestos dinámicos que representan palabras en LSM, como los trabajos de Cervantes et al. (2016); Espejel-Cabrera et al. (2021), que utilizan un corpus de 249 gestos dinámicos, obteniendo resultados de 97 % y 96 % de exactitud respectivamente. También existen trabajos con corpus más pequeños, como es el caso de Martínez-Sánchez et al. (2023), donde se explora un corpus de 100 gestos dinámicos introducidos en una RNN, resultando en un 99.62 % de exactitud; y la tesis de Sainos-Vizuett (2022), que utiliza un corpus de 100 frases en español, todas ellas en contexto médico, de emergencias y frases cordiales. De igual manera, Ramírez Sánchez et al. (2021), con un corpus de 75 señas, obtuvo una exactitud de 94 % utilizando modelos ocultos de Markov y CNN. Los trabajos con corpus más pequeños encontrados fueron Miah et al. (2024); García-Bautista et al. (2017), con conjuntos de datos de 20 y 30 gestos respectivamente.

En la literatura se encuentran trabajos que abarcan ambos tipos de gestos, tales como Martínez-Seis et al. (2019); Garcia-Bautista et al. (2016); Mejía-Peréz et al. (2022); Trujillo-Romero & García-Bautista (2023). Se destaca la investigación de Garcia-Bautista et al. (2016), que abarca el reconocimiento de palabras, letras y números. Sin embargo, en la literatura solo se ha encontrado un trabajo donde se mezclen los problemas de reconocimiento de gestos estáticos y dinámicos orientados hacia la LSM. El trabajo de Gortarez-Pelayo et al. (2023) propone un sistema para aprender toda la dactilología (que incluye tanto señas manuales estáticas como dinámicas), utilizando *AI FingerPose classifier* para llevar a cabo la tarea de reconocimiento.

En los trabajos de Abdullahi et al. (2024); Tran et al. (2023) se encargan de reconocer señas a partir de secuencias espaciales y de video respectivamente pertenecientes a la ASL. En CSL, el trabajo de Liu et al. (2023) se encarga de reconocer series de tiempo provenientes de sensores físicos, mientras que Wei & Lan (2023) se encarga de reconocer señas a partir de video RGB. Dentro del trabajo de Rajalakshmi et al.

(2023) se presenta una metodología donde la entrada es una secuencia espacial y una secuencia de video dentro de un solo modelo de reconocimiento para ISL, por otro lado, en el trabajo de Subramanian et al. (2022) se reconocen señas a partir de secuencias de datos espaciales extraídos de videos. En su mayoría de trabajos de señas dinámicas, se trabaja con secuencias de video RGB o características extraídas a partir de estos.

3.2.2. Tipo de tarea

Aunque anteriormente se mencionó que algunos trabajos incluyen el reconocimiento de números dentro de sus estudios (Jimenez et al., 2017; Estrivero-Chavez et al., 2019), las tareas más recurrentes son el reconocimiento de ideogramas y de la dactilología, siendo esta última una de las más exploradas para señas estáticas.

3.2.2.1. Dactilología

La tarea de reconocimiento de la dactilología se enfoca en identificar las configuraciones manuales que representan las letras del abecedario en LSM. Cabe resaltar que el reconocimiento de configuraciones manuales no se limita únicamente a la dactilología, como se mencionó anteriormente, en el trabajo de Estrivero-Chavez et al. (2019) se reconocen otras configuraciones como LOVE y FAMILY además algunas letras de la dactilología. En su mayoría, los trabajos presentados no abarcan toda la dactilología completa. Trabajos como Mejía-Peréz et al. (2022); Miah et al. (2024); Jimenez et al. (2017); Estrivero-Chavez et al. (2019) solamente utilizan palabras selectas, ya sea por motivos de abordar señas estáticas y dinámicas o ideogramas, dactilología y números en una mismo estudio, donde la contribución son las técnicas de detección y reconocimiento de las señas. También existen trabajos que discriminan a la señas dinámicas de la dactilología para enfocarse en las señas dinámicas como en Garcia-Bautista et al. (2016); Solis et al. (2015); Pérez et al. (2017); Alejandro & Antonio (2021); Rios-Figueroa et al. (2022); Morfín-Chávez et al. (2024).

En Alsharif et al. (2023) utilizan la dactilología de la ASL cuyo nombre en inglés es conocido como *Fingerspelling* y es importante resaltar que aunque presenta similitudes con la dactilología de la LSM, este presenta diferencias en letras como la "K" cuya configuración en LSM es similar a la "P" pero

en ASL es similar a la "V", que además, en algunos modelos reportados en este trabajo presentan confusiones con estas dos letras. Del mismo modo, en el trabajo de Kothadiya et al. (2023) utilizan un conjunto de datos compuesto por la dactilología de la ISL y los números del 0 al 9, reconociendo un total de 36 clases.

3.2.2.2. Ideogramas

El reconocimiento de ideogramas implica la identificación de glosas que representan palabras mediante configuraciones específicas. En la literatura revisada, se notó que, esta área ha sido la menos explorada, sobre todo por la dificultad que representa el reconocimiento de un gesto dinámico. Sin embargo, estos trabajos presentan una gran variedad de corpus enfocados hacia distintas áreas, existen trabajos enfocados hacia el reconocimiento de ideogramas para el uso cotidiano de la LSM (Cervantes et al., 2016; García-Bautista et al., 2017; Espejel-Cabrera et al., 2021), existen trabajos enfocados hacia emergencias y ámbitos médicos vistos en Ramírez Sánchez et al. (2021); Sainos-Vizuett (2022), elementos gramaticales (Martínez-Sánchez et al., 2023), entre otros. Los trabajos de Martínez-Seis et al. (2019); Garcia-Bautista et al. (2016); Trujillo-Romero & García-Bautista (2023), se destacan por realizar estudios que involucran bases de datos grandes que incluyen ideogramas de distintos enfoques, la dactilología completa y números. Sin embargo, ninguno de estos reportó algún modelo multitarea que aborde todas estas señas en conjunto.

En ASL, el área de ideogramas esta muy bien explorado, el trabajo más reciente de Abdullahi et al. (2024) utiliza dos conjuntos de datos de ASL capturados desde el sensor *leap motion* reconociendo un total de 38 ideogramas. El trabajo de Tran et al. (2023), del mismo modo, se encarga de reconocer 20 ideogramas los cuales están orientados hacia el área de medicina (e.g. *Anxious, Blood pressure, Cough, Diabetes,* etc.).

Dentro de los trabajos de CSL, Liu et al. (2023); Wei & Lan (2023) se centran en reconocer ideogramas y frases respectivamente. Donde Liu et al. (2023) se centra en reconocer 48 ideogramas comúnmente utilizados en CSL, mientras que Wei & Lan (2023) utiliza el conjunto de datos *CSL-Daily* que consta de un total de 20,654 videos de señas enfocadas a temas de la vida diaria y tópicos como familia, salud y escolares. En ISL los conjuntos de datos se abordan de manera similar, donde trabajos como Rajalakshmi et al. (2023) reconoce 500 ideogramas de uso diario y Subramanian et al. (2022) que reconoce 13 ideogramas como: *Fail, Friend, Good, Hello, I love you, Like,* entre otras.

3.2.3. Modalidad

Existen dos modalidades principales en las que puede abordarse el reconocimiento automático de lengua de señas: la modalidad aislada y la modalidad continua. En la modalidad aislada, cada seña se presenta de forma individual, con inicios y finales bien definidos, lo que facilita su segmentación y clasificación. En contraste, la modalidad continua implica el reconocimiento de una secuencia fluida de señas, sin pausas evidentes entre una y otra, asemejándose a una frase o enunciado completo. La modalidad continua presenta desafíos adicionales, ya que el modelo debe identificar los límites entre señas y manejar transiciones suaves entre ellas.

3.2.3.1. Aislado

El reconocimiento aislado se centra en identificar señas individuales una a una, sin tener en cuenta la secuencia o el contexto en el que se producen. En este enfoque, cada gesto se reconoce como una entidad separada y distinta. Esto es útil para aplicaciones donde las señas se realizan de manera clara y delimitada, como la enseñanza de algún vocabulario especifico de la LSM o en sistemas donde existe una pausa clara entre cada seña. En la literatura, se observa una predominancia de trabajos donde se realiza la tarea de reconocimiento en modalidad aislada. Esto puede observarse en la Tabla 1.

En ASL con el trabajo de Alsharif et al. (2023) al utilizar imágenes como entrada para reconocer letras del alfabeto, se utiliza un enfoque aislado en donde se desea reconocer una a una las letras del alfabeto. Del mismo modo en el trabajo de Abdullahi et al. (2024) que reconoce secuencias de características extraídas por el sensor *leap motion*. Sucede algo interesante en el trabajo de Wei & Lan (2023) que, a pesar de mencionar el reconocimiento continuo, su método de evaluación está basado en la clasificación aislada de frases completas del corpus para CSL. En cuanto a ISL, los trabajos reportados en esta revisión (Subramanian et al., 2022; Rajalakshmi et al., 2023; Kothadiya et al., 2023) se encargan de reconocer señas de manera aislada.

3.2.3.2. Continuo

El reconocimiento continuo de señas implica el identificarlas dentro de una secuencia continua sin pausas claras entre ellas. Este enfoque es más complejo, debido a que se debe lidiar con la variabilidad y la fluidez de las señas tal como se utilizan en una conversación natural. Un sistema de reconocimiento continuo debe ser capaz de segmentar y reconocer cada seña dentro del flujo continuo, tomando en cuenta las transiciones y, en algunos casos, el contexto entre los gestos. Esto es crucial para las aplicaciones que buscan interpretar oraciones completas o interacciones más naturales y fluidas en LSM.

Se encontraron únicamente cuatro trabajos relacionados con el reconocimiento continuo de lengua de señas mexicana. El trabajo de Morfín (2023) es el más cercano a lo que se busca proponer en el presente documento, ya que utiliza modelos entrenados para reconocer señas de la dactilología en modalidad aislada, adaptándolos ahora a un dominio continuo en tiempo real para la generación de palabras. Este trabajo presenta resultados basados en el tiempo (ms) de ejecución del algoritmo en el dominio continuo, así como la distancia de Levenshtein para evaluar la exactitud con la que se reconoció la palabra. Trabajos como Garcia-Bautista et al. (2016); García-Bautista et al. (2017); Trujillo-Romero & García-Bautista (2023); González-Rodríguez et al. (2024), aunque reportan el uso del algoritmo en tiempo real, no definen una estrategia para evaluar el uso del reconocimiento continuo.

En el trabajo de Borges-Galindo et al. (2024) se busca realizar una evaluación del rendimiento de sus modelos enfocado a un escenario en línea donde considera el número de errores que el sistema puede tener cuando el usuario ejecuta la seña. Borges-Galindo et al. (2024) menciona que actualmente no existen métricas que evalúen dicho comportamiento por lo que propone una métrica llamada *Average Error in Live Video during Translation* (AELVT). Esta métrica mide la media de veces que el sistema traduce erróneamente una seña. Esta métrica determina el comportamiento en tiempo real, donde las palabras objetivo son totalmente desconocidas.

Tran et al. (2023) reporta el uso de modelos entrenados con señas aisladas para reconocer señas de la ASL en tiempo real. Emplea un ventaneo con un traslape del 90 %, para analizar las ocurrencias de una seña dentro de la ventana para determinar que seña fue realizada. En CSL, el trabajo de Liu et al. (2023) cuenta con un conjunto de datos con 10 oraciones diferentes, en los cuales utiliza su modelo de reconocimiento aislado, que analiza series temporales, para reconocer dentro de una serie de tiempo múltiples señas. Esto lo realiza mediante un ventaneo dentro de la serie temporal con distintos tamaños (150, 200 y 250 puntos) para adaptarse a distintas duraciones de señas. Sin embargo, aunque los trabajos

realizados en estas lenguas de señas reporte que los resultados son aceptables, no se profundiza en una evaluación más exhaustiva.

3.2.4. Modos de captura

Tras revisar los distintos enfoques según el tipo de seña, tarea y modalidad, el siguiente aspecto clave por considerar es el modo en que se capturan los datos. Este factor resulta determinante, ya que define las posibilidades técnicas y metodológicas del reconocimiento automático. En la literatura se reconocen principalmente dos estrategias de adquisición: una basada en sensores y otra basada en visión, siendo esta última la más utilizada en estudios recientes debido a su naturaleza no invasiva y mayor escalabilidad.

3.2.4.1. Basado en sensores

El enfoque utilizando sensores implica el uso de dispositivos específicos que se colocan en el cuerpo del usuario para capturar movimientos y posición. Entre estos dispositivos se incluyen guantes con sensores flexibles e inerciales que midan la orientación y aceleración como lo visto en el trabajo de Ocampo et al. (2020), donde se desarrolla un guante compuesto por estos sensores para reconocer gestos estáticos. Por otro lado, en el trabajo de Varela-Santos et al. (2021) se combina visión y hardware rudimentario, este consta de una cámara colocada en le cuello del usuario, lo cual le agrega un toque de portabilidad para poder traducir mediante el uso de guantes con placas que tienen patrones aleatorios, y utilizando centroides y distancia euclidiana, se logra reconocer gestos estáticos.

El trabajo de Wei & Lan (2023) utilizan sensores de deformación (*strain sensors*) para cada uno de los dedos, que capturan la flexión y extensión de los dedos registrando el ángulo de flexión de cada articulación. Esto permite inferir la configuración de la mano dado que tan cerrados o extendidos estén los dedos durante la seña, en CSL. También utiliza una Unidad de Medición Inercial (IMU), que esta montada en el dorso de la mano, este mide señales del acelerómetro, giroscopio y magnetómetro, permitiendo capturar la trayectoria del movimiento de la mano en el espacio, logrando registrar el componente dinámico de las señas.

3.2.4.2. Basado en visión

El enfoque basado en visión ha sido uno de las más explorados cuando se trata de tareas de reconocimiento de señas, donde se destaca el uso de cámaras con información de profundidad (RGB-G), como lo es el Kinect v1 y el OAK-D, los trabajos que abarcan esta tecnología son: Garcia-Bautista et al. (2016); Mejía-Peréz et al. (2022); Sosa-Jiménez et al. (2022); Trujillo-Romero & García-Bautista (2023); García-Bautista et al. (2017); Miah et al. (2024); Jimenez et al. (2017); Rios-Figueroa et al. (2022). También existen dispositivos de captura como LeapMotion, cuya aplicación es explorada en el trabajo de Estrivero-Chavez et al. (2019). En dicho estudio, se utiliza LeapMotion para la tarea de reconocimiento, sin embargo, se reporta como una limitación su capacidad reducida para interpretar gestos dinámicos y complejos, lo que restringe su aplicabilidad en escenarios más realistas. Dentro de la ASL, el trabajo Abdullahi et al. (2024) utiliza conjuntos de datos que fueron previamente capturados con el sensor *LeapMotion* el cual incluye información 3D de coordenadas espaciales de la mano, incluyendo posición y orientación para el calculo de angulos y trayectorias.

También se desacata el uso de cámaras RGB convencionales como se observa en los trabajos de Cervantes et al. (2016); Solis et al. (2015); Martínez-Seis et al. (2019); Espejel-Cabrera et al. (2021); Ramírez Sánchez et al. (2021); Martínez-Sánchez et al. (2023); Pérez et al. (2017); Alejandro & Antonio (2021); Morfín-Chávez et al. (2024); González-Rodríguez et al. (2024); Borges-Galindo et al. (2024). El uso de tecnologías basadas en visión, provee muchas ventajas, como el no requerir del uso de dispositivos adicionales, lo cual lo hace cómodo y no invasivo. También que puede capturar movimientos complejos y expresiones faciales junto con los gestos manuales, las cuales se exploran en González-Rodríguez et al. (2024). Del mismo modos los trabajos realizados en otras lenguas de señas como Alsharif et al. (2023); Tran et al. (2023) de ASL, Wei & Lan (2023) de CSL y Rajalakshmi et al. (2023); Subramanian et al. (2022); Kothadiya et al. (2023) para ISL utilizan conjuntos de datos que fueron capturados mediante cámaras RGB. Sin embargo, en esta revisión, no se encontraron trabajos que hayan explorado soluciones que resuelvan limitaciones tales como la oclusión y condiciones de iluminación.

3.2.5. Extracción de características

Cada trabajo revisado se alinea, en mayor o menor medida, con alguno de los tres enfoques predominantes identificados para la extracción de características. Estos enfoques se agrupan, según el tipo de información

utilizada, en: señales, imágenes o coordenadas de puntos clave (keypoints).

3.2.5.1. Señales

El procesamiento de señales en el contexto del reconocimiento de la LSM implica el uso de sensores físicos que capturan datos precisos sobre el movimiento y la posición de las manos y los dedos. Esta técnica es utilizada para capturar y analizar datos que provienen de sensores, como guantes o dispositivos de captura de movimiento, que permiten obtener información detallada sobre la posición y el movimiento de las manos y los dedos. En el trabajo de Ocampo et al. (2020), al utilizar sensores flexibles en los dedos del guante, para medir la flexión y detectar las posición de los dedos se realizó mediante técnicas de procesado de señales. También, como el guante incorpora un giroscopio, se requieren técnicas para calcular la posición y filtrar la señal para que de manera limpia represente la seña que realiza el usuario. El trabajo de Liu et al. (2023) para la CSL, utiliza las series temporales como entrada para ser procesadas por una CNN, donde esta toma las señales como una matriz de 8x200 (8 canales, 200 puntos) en donde las filas representan los canales sensoriales y las columnas el tiempo. Se tienen 8 canales dado al número de características, que son 5 sensores de deformación (uno por cada dedo) y 3 ejes de aceleración lineal.

3.2.5.2. Imágenes

Las técnicas de visión por computadora juegan un papel fundamental en el proceso de extracción de características, proporcionando métodos avanzados para analizar y interpretar los gestos. En el trabajo de Solis et al. (2015) se utilizan los Momentos de Jacobi-Fourier para representar imágenes de señas estáticas, siendo robustos ante transformaciones geométricas, como la traslación, rotación y escalado. Este método permite extraer características que se utilizaron en redes neuronales para la clasificación de señas. El trabajo de Pérez et al. (2017) presenta el uso de Momentos Hu, estos descriptores geométricos, son invariantes a la traslación, rotación y escalado, utilizados para identificar características geométricas de la manos.

Por otro lado, también se identificó el uso de características 3D Haar-like que son extraídas de imágenes de profundidad. Dichas características capturan diferencias entre la intensidad promedio entre regiones

rectangulares adyacentes, estas son presentadas en el trabajo de Jimenez et al. (2017). También existen trabajos que emplean segmentación basada en color. Por ejemplo, en el estudio de Espejel-Cabrera et al. (2021), se utiliza el espacio de color HSV para segmentar las regiones de piel. Se entrenan redes neuronales para detectar el color de la piel y realizar una mejor segmentación así como reducir el tiempo de procesamiento. En Alejandro & Antonio (2021); Martínez-Sánchez et al. (2023) se optó por utilizar modelos de *deep learning* como CNN para extraer características relevantes de las imágenes. Además de aplicar *transfer learning* con modelos preentrenados como ResNet50 para mejorar la precisión con conjuntos de datos pequeños.

En ASL, el trabajo de Alsharif et al. (2023) mediante modelos CNNs y Transformers extraen características profundas sobre imágenes RGB, probando una gran variedad de modelos pre-entrenados. Metodología similar a la utilizada por Kothadiya et al. (2023), pero centrada únicamente en la arquitectura de un *vision transformer* para el reconocimiento del alfabeto de ISL. Por otro lado, el trabajo de Wei & Lan (2023) implementó un módulo de desplazamiento temporal en un modelo ResNet-50, también utilizó el modelo Action-net, que incluye módulos de atención espacial-temporal y de atención de movimiento para reconocer señas de la CSL.

3.2.5.3. Keypoints

En el capítulo anterior se mencionó que un *keypoint* es una característica de alto nivel que puede ser utilizado para diversas tareas de visión por computadora. Dentro del contexto de reconocimiento de la LSM, los *keypoints* en regiones como las manos, rostro y parte superior del cuerpo, son críticos para identificar y analizar gestos y movimientos. Con la llegada de librerías como *MediaPipe* y *OpenPose* (explicadas en la sección 2.2), resulta muy efectivo el utilizar procesamiento basado en *keypoints*. El costo monetario es mínimo a comparación de otros enfoques, esto abre la puerta para desarrollar aplicaciones más accesibles para la comunidad Sorda. En el trabajo de Ramírez Sánchez et al. (2021), se utiliza *MediaPipe* para capturar puntos de la cara, la posición del cuerpo y las manos en tiempo real. Además utiliza CNNs para codificar estos *keypoints* y extraer patrones, para posteriormente clasificarlos utilizando Modelos Ocultos de Markov. Por otro lado, el trabajo de González-Rodríguez et al. (2024) se centra en reducir las características, que consta del número de *keypoints*, logrando reducir la dimensionalidad a 50 *keypoints* esenciales para reconocer señas. En el trabajo de Morfín-Chávez et al. (2024) se extraen 21 *keypoints* de la mano, cada uno representado por coordenadas x, y, z. Los componentes se normalizan

según el ancho y la altura de la imagen para formar un vector de características para alimentar modelos de clasificación de señas estáticas. Por otro lado, el sensor *Leap Motion* también es capaz de capturar la posición tridimensional de cada dedo en ambas manos. Lo que permite modelar hasta 27 señas estáticas con alta precisión, como se observa en el trabajo de Estrivero-Chavez et al. (2019).

El uso de *keypoints* dentro de otros trabajos de lenguas de señas ha sido ampliamente explorado, donde trabajos como Tran et al. (2023) utilizan *MediaPipe* para reconocer señas de la ASL y los trabajos de Subramanian et al. (2022); Rajalakshmi et al. (2023) del mismo modo trabajan con *MediaPipe* para reconocer señas de ISL. Dentro de estos trabajos, destaca el de Rajalakshmi et al. (2023) que no sólo utiliza la información de los *keypoints* sino que estos son mencionados como características espaciales y utilizando arquitecturas CNN, aplicado en los frames de video, se extraen características temporales, lo que se podría decir que es una metodología híbrida entre técnicas de visión y técnicas basadas en *keypoints*.

3.3. Comparativa del trabajo relacionado

La Tabla 1 presenta un resumen comparativo de los trabajos revisados en esta tesis. A partir de este análisis, se concluye que el reconocimiento automático de lengua de señas abarca enfoques sumamente diversos, en los que se emplean distintas técnicas de visión por computadora y procesamiento de señales para la extracción de características. Una observación clave es que el reconocimiento en modalidad aislada ha sido ampliamente explorado, destacando el uso de características profundas extraídas mediante modelos basados en redes convolucionales (CNN). En contraste, se identifica una área de oportunidad en el tratamiento de problemas asociados a la modalidad continua, especialmente en lo que respecta a los métodos de evaluación. En la literatura, el trabajo de Borges-Galindo et al. (2024) es el único que propone una métrica específica para evaluar esta modalidad, mientras que los demás estudios se limitan a ofrecer resultados cualitativos. Asimismo, se observa que son escasos los trabajos que emplean corpus con más de 50 señas, lo que evidencia otra área de oportunidad relevante: el desarrollo de modelos capaces de reconocer vocabularios amplios y representativos, más cercanos al uso real de las lenguas de señas.

Tabla 1. Resumen de trabajos revisados sobre reconocimiento de lengua de señas, incluyendo aquellos desarrollados en la institución. Nótese que la columna de resultados corresponde exclusivamente a la modalidad aislada, evaluada mediante la métrica de *accuracy*.

Trabajo	Modalidad	Tarea	Características	Tipo de seña	No. clases	Resultados
Solis et al. (2015)	Aislado	Dactilología	Imagenes	Estática	24	95.00 %
Cervantes et al. (2016)	Aislado	Ideogramas	Imagenes	Dinámica	249	97.04 %
Garcia-Bautista et al. (2016)	Continuo	Dactilología	Imagenes	Dinámica	10	95.71 %
Pérez et al. (2017)	Aislado	Dactilología	Imagenes	Estática	21	91.00 %
Jimenez et al. (2017)	Aislado	Dactilología	Imagenes	Estática	10	95.00 %
García-Bautista et al. (2017)	Continuo	Ideogramas	Imagenes	Dinámica	20	98.57 %
Martínez-Seis et al. (2019)	Aislado	Ideogramas	Imagenes	Dinámica	27	95.23 %
Estrivero-Chavez et al. (2019)	Aislado	Dactilología	Sensores	Estática	27	99.80 %
Ocampo et al. (2020)	Aislado	Dactilología	Sensores	Dinámica	21	88.30 %
Alejandro & Antonio (2021)	Aislado	Dactilología	Imagenes	Estática	21	91.56 %

Continúa en la siguiente página

Tabla 1 – continuación de la página anterior

Trabajo	Modalidad	Tarea	Características	Tipo de seña	No. clases	Resultados
Varela-Santos et al. (2021)	Aislado	Ideogramas	Imagenes	Estática	20	89.00 %
Ramírez Sánchez et al. (2021)	Aislado	Ideogramas	*Keypoints*	Dinámica	75	94.90 %
Espejel-Cabrera et al. (2021)	Aislado	Ideogramas	Imagenes	Dinámica	249	94.90 %
Rios-Figueroa et al. (2022)	Aislado	Dactilología	Imagenes	Estática	21	99.73 %
Mejía-Peréz et al. (2022)	Aislado	Dactilología	*Keypoints*	Dinámica	30	97.11 %
Sosa-Jiménez et al. (2022)	Aislado	Ideogramas	Imagenes	Dinámica	43	99.50 %
Trujillo-Romero & García-Bautista (2023)	Continuo	Ideogramas	Imagenes	Dinámica	600	95.27 %
Martínez-Sánchez et al. (2023)	Aislado	Ideogramas	Imagenes	Dinámica	100	99.62 %
Morfín-Chávez et al. (2024)	Aislado	Dactilología	*Keypoints*	Estática	21	98.00 %
Miah et al. (2024)	Aislado	Dactilología	Imagenes	Dinámica	30	99.96 %

Continúa en la siguiente página

Tabla 1 – continuación de la página anterior

Trabajo	Modalidad	Tarea	Características	Tipo de seña	No. clases	Resultados
González-Rodríguez et al. (2024)	Continuo	Ideogramas	*Keypoints*	Dinámica	10	98.80 %
Borges-Galindo et al. (2024)	Aislado	Ideogramas	*Keypoints*	Dinámica	20	93.33 %
American Sign Language						
Alsharif et al. (2023)	Aislado	Dactilología	Imagenes	Estática	29	99.98 %
Tran et al. (2023)	Continuo	Ideogramas	*Keypoints*	Dinámica	20	99.50 %
Abdullahi et al. (2024)	Aislado	Ideogramas	Sensores	Dinámica	38	92.70 %
Chinese Sign Language						
Wei & Lan (2023)	Aislado	Ideogramas	Imagenes	Dinámica	500	99.30 %
Liu et al. (2023)	Continuo	Ideogramas	Sensores	Dinámica	48	98.54 %
Indian Sign Language						
Subramanian et al. (2022)	Aislado	Ideogramas	*Keypoints*	Dinámica	13	95.00 %
Rajalakshmi et al. (2023)	Aislado	Ideogramas	*Keypoints*	Dinámica	500	99.87 %
Kothadiya et al. (2023)	Aislado	Dactilología	Imagenes	Estática	36	99.29 %

Continúa en la siguiente página

Tabla 1 – continuación de la página anterior

Trabajo	Modalidad	Tarea	Características	Tipo de seña	No. clases	Resultados
Trabajo realizado dentro de la institución						
Sainos-Vizuett (2022)	Aislado	Ideogramas	*Keypoints*	Dinámica	100	81.00 %
Morfín (2023)	Continuo	Dactilología	*Keypoints*	Estática	21	91.00 %
Gortarez-Pelayo et al. (2023)	Aislado	Dactilología	*Keypoints*	Dinámica	27	87.70 %

Capítulo 4. Metodología

Tras la revisión del estado del arte presentada en el capítulo anterior, este capítulo describe la metodología adoptada para abordar el reconocimiento automático de la Lengua de Señas Mexicana (LSM), considerando tanto la modalidad aislada como la continua. La propuesta metodológica integra procesamiento de datos, extracción de características y el uso de modelos de aprendizaje profundo en las diferentes modalidades de reconocimiento.

4.1. Corpus de LSM utilizado

Para la realización de este trabajo, se utilizaron dos conjuntos de datos recolectados en un trabajo previo (Sainos-Vizuett, 2022). Ambos fueron obtenidos en un mismo ambiente controlado, cuya configuración fue diseñada cuidadosamente considerando aspectos como el fondo y la iluminación (ver Figura 17).

(a) Diagrama ilustrativo del ambiente controlado extraído de la tesis de Morfín (2023)

(b) Imagen del ambiente controlado extraído de la tesis de Sainos-Vizuett (2022)

Figura 17. Ambiente controlado para la captura de datos.

4.1.1. Diseño del diccionario

En la tesis de Sainos-Vizuett (2022), parte de su trabajo fue el diseño de un corpus, ante la carencia de recursos adecuados para el entrenamiento de modelos de aprendizaje de máquina. Se tomaron como

referencia 110 frases en español enfocadas hacia un contexto médico, emergencias y cotidianas. Con la asesoría de intérpretes de la Asociación Regional de Sordos Ensenadenses (ARSE) y de una especialista en LSM se validaron cada una de las frases.

4.1.2. Captura de datos

Dos expertos reprodujeron cada frase en LSM y se descartaron aquellas señas que presentaron variaciones significativas entre ellos, reflejo de los regionalismos existentes en la lengua. Después se convirtieron las frases en español a glosas de la LSM —recordando que una glosa es la representación de una seña mediante una palabra en español— i.e., segmentaron cada ideograma de la frase. Posteriormente se realizó una captura de datos de los Ideogramas a 10 participantes denominados **no expertos**.

Sainos-Vizuett (2022) estableció el siguiente protocolo para la captura de señas con participante no expertos:

1. Se le explica al participante que tiene que realizar las señas con la mano dominante y el uso de la mano de apoyo.
2. Se le presenta un video de muestra como instructivo para realizar la seña.
3. El participante debe imitar la seña vista en el video instructivo y practicarlo dos veces.
4. Se realiza la grabación de la seña.
5. Una vez finalizada la grabación se repiten los tres pasos anteriores para la grabar la siguiente seña.

El grupo de participantes no expertos tiene las siguientes características: ocho de ellos fueron hombres y dos mujeres, 4 participantes utilizaron lentes durante las grabaciones, todos los participantes son **diestros** (i.e. que su mano dominante es la derecha y su mano de apoyo la izquierda). La variabilidad física entre los participantes es lejana entre si, mencionando que la prioridad en el diseño de adquisición de datos es que los participantes realicen de forma correcta cada seña dentro del corpus. Todos los participantes firmaron un consentimiento informado en el que autorizaron el uso de su rostro con fines de investigación y divulgación.

Posteriormente, siguiendo el mismo protocolo de captura propuesto por Sainos-Vizuett (2022), se llevó a cabo una segunda sesión de adquisición de datos, centrada en la captura de las frases completas

previamente desarrolladas con el apoyo de personas expertas. A diferencia de la captura previa —en la que se registró cada glosa de forma aislada, i.e., el participante partía de una posición neutral, ejecutaba la seña y regresaba a dicha posición—, en esta ocasión las secuencias de señas se realizaron de forma continua, sin retornar a la posición neutral hasta completar la frase. Dentro de este conjunto de datos, se destacan ocho participantes a los cuales denominamos **semi expertos**.

Dentro de las características de estos participantes, siete participantes son mujeres y uno es hombre, del mismo modo existe una variabilidad física entre participantes. Las frases dentro del conjunto de datos varían en cuanto a número de glosas expresadas dentro de cada video, siendo desde uno hasta cinco señas. Contando con un total de 96 frases que fueron depuradas donde cada participante realiza la frase tres veces.

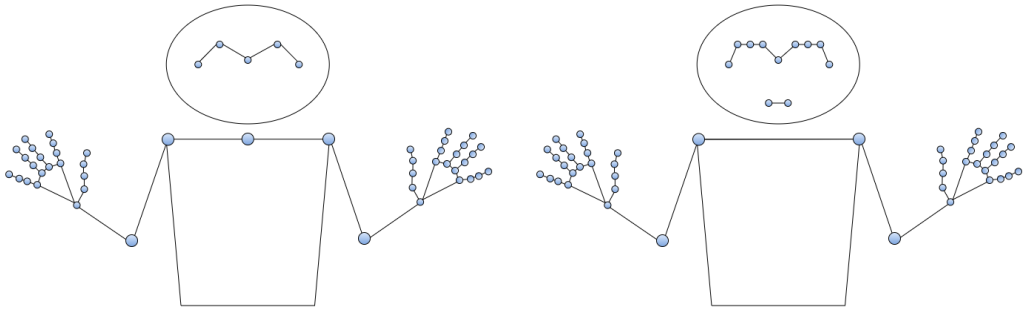
4.2. Extracción de características

El presente trabajo se basa en la extracción de *keypoints* correspondientes al cuerpo, el rostro y las manos en cada fotograma de los videos de señas capturados. Para ello se emplean los modelos *OpenPose* y *MediaPipe*, los cuales estiman las coordenadas en píxeles de las articulaciones de los participantes en cada imagen que compone el video. De ambas salidas se aprovechan las coordendas (x, y) y un valor c que es proporcionado nativamente por *OpenPose* y adicionado en *MediaPipe* el cual indica el estado del *keypoint* (activo o inactivo). Un *keypoint* inactivo denota que la articulación no es visible para el modelo y, en consecuencia, no puede ser estimada.

Selección de articulaciones. Para mitigar la variabilidad de los gestos faciales que no aportan información semántica a la seña, se decidió descartar la mayoría de los puntos del rostro y conservar únicamente aquellos que describen la rotación y la orientación de la cabeza. Asimismo, se priorizaron las articulaciones dentro del espacio señante de la LSM; en consecuencia, se retuvieron los *keypoints* de ojos, orejas, boca, hombros, codos y manos (ver Figura 18).

- **OpenPose:** se emplearon los 42 *keypoints* de las manos más los índices $\{0, 1, 2, 3, 5, 6, 15, 16, 17, 18\}$, correspondientes a cabeza y hombros, lo que da un total de $\mathbf{52}$ *keypoints* seleccionados.
- **MediaPipe:** se seleccionaron los *keypoints* con índices del 0–14 (región facial y parte superior del tronco) junto con los 42 puntos de ambas manos, obteniendo un total de $\mathbf{57}$ *keypoints*.

Esta selección proporciona un conjunto compacto y coherente, centrado en las regiones corporales más informativas para el reconocimiento de señas.



(a) Esqueleto de *OpenPose* (b) Esqueleto de *MediaPipe*

Figura 18. Diagrama de las articulaciones seleccionadas.

4.3. Estudio comparativo entre grupos de características

Se evaluó la consistencia de los datos obtenidos de los modelos, mediante un estudio comparativo en el que se midió la pérdida de datos que se obtiene al utilizar el valor c resultante de cada *keypoint* inferido por los modelos. Entonces, se recorre un vector de características K considerando que $k_c < 0.5$ significa que el keypoint está apagado. Por lo que se tiene:

$$\text{Pérdida} = \frac{\sum_{i=1}^{n} k_i}{n \cdot K} \tag{14}$$

donde $\sum_{i=1}^{n} k_i$ es la suma de todos los *keypoints* apagados en los n frames; y $n \cdot K$ es el total de *keypoints* esperados (Ver Figura 19).

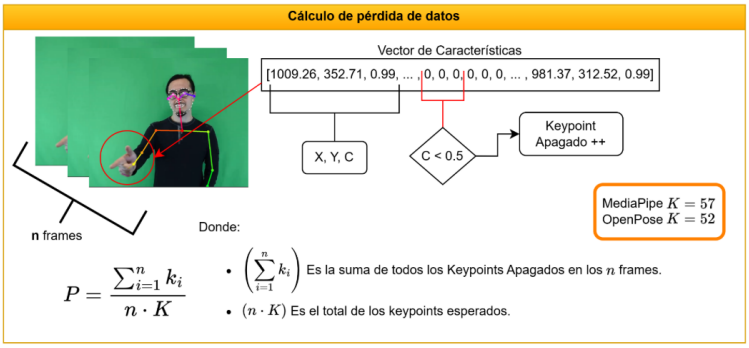


Figura 19. Diagrama ilustrativo del proceso de evaluación.

4.4. Aumento de datos

El conjunto de datos de glosas aisladas dispone de muy pocas muestras por ideograma, lo que resulta insuficiente para entrenar de forma efectiva modelos de aprendizaje profundo. Ko et al. (2019) propusieron un método para solucionar la falta de datos mediante la extracción de secuencias de frames utilizando una técnica denominada *Frame Skip Sampling* que realiza lo siguiente: Se tiene una secuencia $S = (f_1, \ldots, f_l)$ con l frames de longitud, al seleccionar un número arbitrario fijo de frames, $n = 15$ para este caso. La longitud promedio de espacios entre frames se calcula de la siguiente forma: $z = \lfloor \frac{l}{n-1} \rfloor$.

Considerando este factor z se extrae un conjunto de frames con índices de la siguiente secuencia $Y = (y, y + z, y + 2z, \ldots, y + (n-1)z)$ donde $y = \frac{l - z(n-1)}{2}$, a Y se le conoce como la **secuencia base**. Después se calcula una secuencia de números aleatorios $R = (r1, \ldots, r_n)$ con valores de $[1, z]$. Se crea una nueva secuencia Y_{new} sumando la secuencia base con la secuencia aleatoria $Y_{new} = (Y_1 + R_1, Y_2 + R_2, \ldots, Y_n + R_n)$. Siempre y cuando el factor $z > 1$ y $l > n$ es posible generar muestras distintas entre sí. Por lo que se generaron 50 nuevas secuencias de $n = 15$ frames cada uno por cada secuencia de video original (Ver Figura 20).

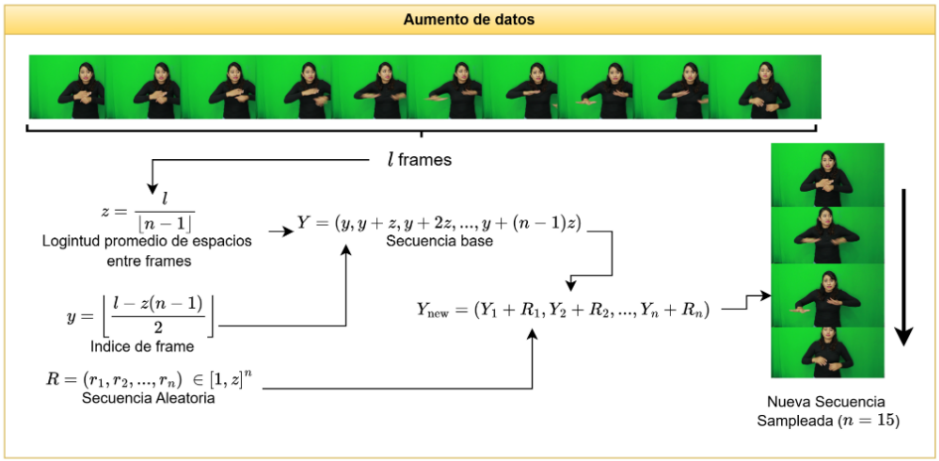


Figura 20. Diagrama ilustrativo del *Frame Skip Sampling* propuesto por Ko et al. (2019).

4.5. Estudio de similitud entre muestras de datos aumentadas

Con el objetivo de medir la similitud en los datos aumentados a modo de análisis exploratorio, se decidió calcular la distancia euclidiana de cada uno de los *keypoints* entre todas las muestras aumentadas a

través de los frames. Suponiendo que se tiene que calcular la similitud entre 2 muestras aumentadas:

$$Similitud = \frac{\sum_{i=1}^{n}\sum_{k=1}^{K}\sqrt{(x_{k,1,i} - x_{k,2,i})^2 + (y_{k,1,i} - y_{k,2,i})^2}}{n \cdot K} \tag{15}$$

donde n es el número de frames y K el número de *keypoints*. El mínimo posible es 0, cuando las dos muestras coinciden punto a punto, cuanto menos sea el valor resultante, mayor sera su similitud entre las muestras aumentadas (ver Figura 21).

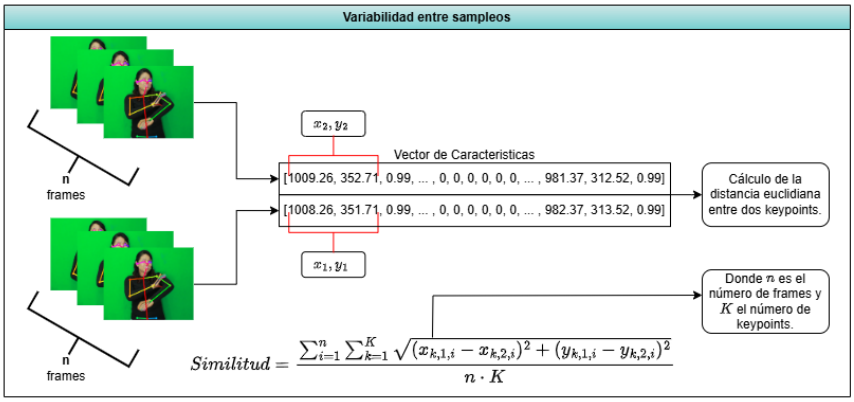


Figura 21. Diagrama ilustrativo del cálculo de similitud.

4.6. Preprocesamiento de datos

El preprocesamiento de datos constituye una etapa esencial en cualquier tarea de aprendizaje de máquina y, en el contexto de reconocimiento de señas, adquiere particular relevancia, ya que las características extraídas están sujetas a variaciones en las coordenadas espaciales (x, y) debido a desplazamientos en la posición, diferencias de escala (e.g., el tamaño aparente de la mano en función de la proximidad a la cámara) y la presencia de valores faltantes. En esta sección se describen los procedimientos empleados para mitigar dichas variaciones.

4.6.1. Normalización

La normalización, también conocida como escalado de datos, tiene como objetivo ajustar la escala de los atributos numéricos para que todos contribuyan de manera equitativa durante el entrenamiento del

modelo, en este caso los *keypoints* extraídos de las secuencias de imágenes. Este preprocesamiento busca hacer que los datos sean invariantes al tamaño, la proximidad y la posición del sujeto dentro del video, facilitando así la convergencia del algoritmo durante el entrenamiento y posteriormente en la inferencia.

No obstante, los métodos de escalado comúnmente utilizados tienden a aplicar una normalización global sobre todo el conjunto de datos, lo que puede resultar problemático cuando se trabaja con secuencias temporales. En estos casos, es fundamental preservar la dinámica del movimiento a través de los fotogramas.

Consideremos una secuencia temporal $S = (f_1, f_2, \ldots, f_n)$, donde cada fotograma f_n se representa mediante un vector de características compuesto por las coordenadas:

$$f_n = [V_x, V_y], \tag{16}$$

donde:

$$V_x = (v_1^x, v_2^x, \ldots, v_k^x), \quad V_y = (v_1^y, v_2^y, \ldots, v_k^y), \tag{17}$$

y k es el número total de *keypoints*.

El proceso de normalización consiste en transformar cada componente utilizando su respectiva media (μ) y desviación estándar (σ):

$$\dot{V}_x = \frac{V_x - \mu(V_x)}{\sigma(V_x)}, \qquad \dot{V}_y = \frac{V_y - \mu(V_y)}{\sigma(V_y)}. \tag{18}$$

De esta manera, se logra escalar los datos preservando las trayectorias de movimiento entre fotogramas, lo cual es esencial para conservar la información temporal relevante en tareas de clasificación de secuencias.

4.6.2. Manejo de datos faltantes

El conjunto de datos presenta casos de valores faltantes, lo cual puede deberse a fenómenos como la oclusión o la salida de las manos del encuadre del video, entre otros factores. Una estrategia empleada para mitigar este problema consiste en utilizar la técnica conocida como *forward fill*, la cual reemplaza los valores faltantes con la última observación no nula registrada previamente en la secuencia.

Este método se aplicó directamente sobre las características extraídas en cada fotograma. Sin embargo, presenta una limitación importante: si los primeros valores de la secuencia ya contienen datos faltantes, estos no son rellenados. Para abordar esta situación, se adoptó un enfoque complementario basado en la relación morfológica entre articulaciones. Específicamente, cuando se detecta un *keypoint* faltante, se reemplaza con el valor correspondiente de la articulación morfológicamente más cercana. Por ejemplo, si el *keypoint* de la mano con índice 12 está ausente en un fotograma arbitrario, dicho valor es rellenado con la información del *keypoint* 11.

4.7. Reconocimiento de señas dinámicas en modalidad aislada

En este apartado, se detallan los modelos construidos y evaluados para el reconocimiento de señas en modalidad aislada. Se explicarán los experimentos realizados y los parámetros a optimizar dentro de la construcción del mejor modelo resultado de la evaluación.

4.7.1. Modelos de clasificación

Se implementaron varias arquitecturas de aprendizaje profundo cuyo rol es procesar representaciones basadas en *keypoints* para aprovechar las dependencias temporales presentes en las señas. A diferencia de los enfoques tradicionales de aprendizaje automático, que requieren una ingeniería manual de características y suelen tener un rendimiento limitado cuando se trata de datos secuenciales complejos, el aprendizaje profundo permite modelar automáticamente patrones espaciales y temporales mediante capas jerárquicas de representación —el funcionamiento de las redes neuronales se explica en el Anexo .1—.

Dentro de las arquitecturas diseñadas se incluye una red residual (ResNet 1D) para la extracción de características espaciales y modelos recurrentes como RNN simple, LSTM, BiLSTM y GRU. Cabe mencionar que todos los modelos fueron entrenados a través de *Tensorflow*, una herramienta que facilita la creación de modelos basados en redes neuronales Abadi et al. (2015).

4.7.1.1. RNN simple

Este modelo utiliza una capa de RNN simple con 128 unidades, la cual procesa secuencias de entrada con forma (n, d), donde $n = 15$ corresponde al número de fotogramas por secuencia y d representa la dimensionalidad de las características extraídas de los *keypoints*. La capa de clasificación está compuesta por dos capas totalmente conectadas (densas) de 128 y 32 unidades, respectivamente, y se incluye una capa de *dropout* con una tasa de eliminación del 30 %. La clasificación final se realiza mediante una capa de salida softmax, cuya cantidad de neuronas coincide con el número total de señas a predecir (ver Figura 22).

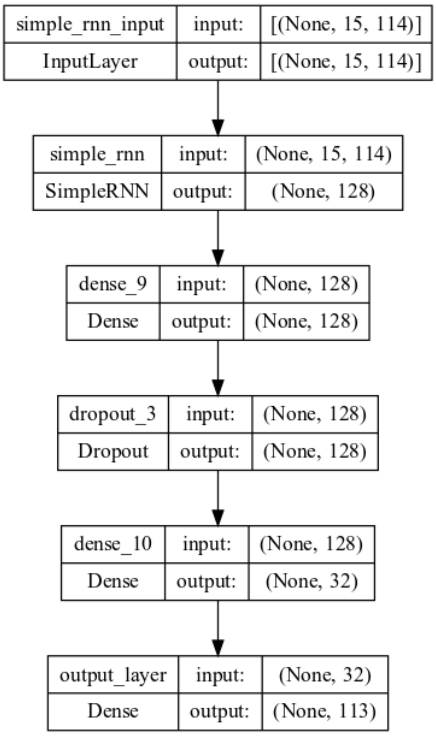


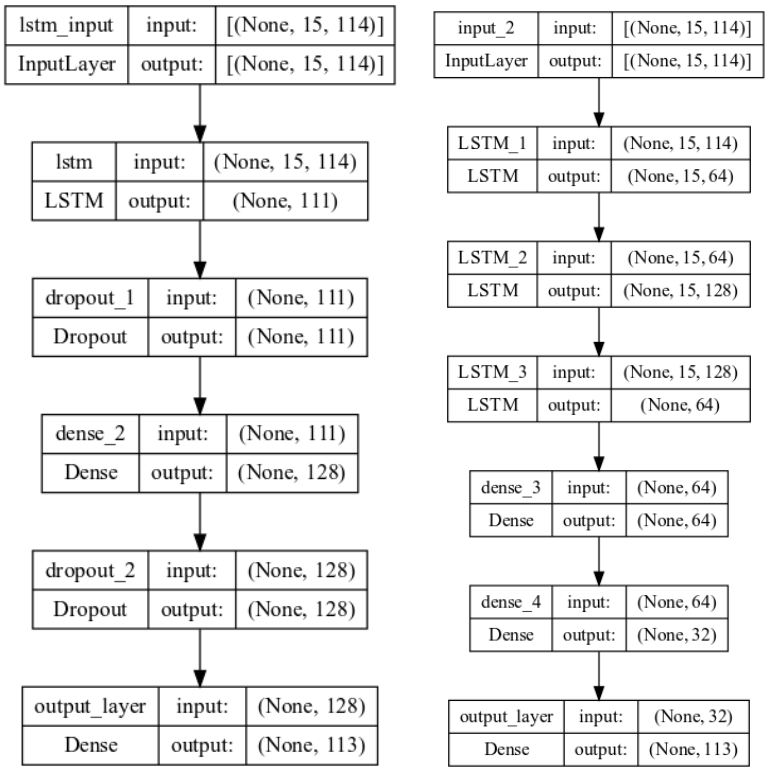
Figura 22. Arquitectura del modelo RNN simple.

4.7.1.2. LSTM

Se implementaron dos arquitecturas basadas en LSTM con el propósito de capturar de manera más efectiva las dependencias temporales presentes en las secuencias de señas. El primer modelo está conformado por una única capa LSTM de 111 unidades con función de activación ReLU, seguida por una capa de

dropout con una tasa del 80 %. Las características extraídas se procesan posteriormente en una capa densa de 128 unidades, acompañada por una segunda capa de *dropout* del 70 %. Finalmente, la salida se genera mediante una capa softmax para realizar la clasificación (ver Figura 23(a)).

El segundo modelo, adaptado de Samaan et al. (2022), presenta una estructura jerárquica más profunda compuesta por tres capas de LSTM con 64, 128 y 64 unidades, respectivamente. Estas capas están configuradas para conservarla secuencia de salida mediante la opción *return sequences*, permitiendo así preservar las relaciones temporales entre los fotogramas. La capa de clasificación consta de dos capas densas de 64 unidades y 32 unidades, seguidas por una cada de salida softmax (ver Figura 23(b)).



(a) Modelo propuesto

(b) Modelo adaptado de Samaan et al. (2022)

Figura 23. Arquitecturas de modelos LSTM.

4.7.1.3. LSTM Bidireccional (BiLSTM)

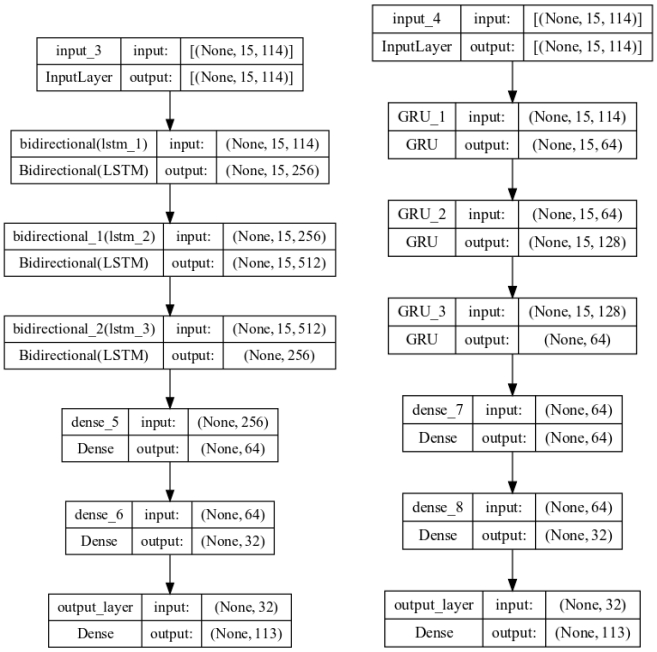
En un modelo recurrente tradicional, las secuencias se procesan únicamente en un sentido: desde el primer hasta el último estado. El procesamiento bidireccional permite que el modelo recorra la secuencia en dos

direcciones simultáneamente (hacia adelante y hacia atrás). Este enfoque enriquece la representación de cada paso temporal, ya que combina información del paso y del futuro en un mismo estado oculto.

Se implementó un modelo BiLSTM, inspirado en Samaan et al. (2022), con el objetivo de mejorar la extracción de características temporales mediante el procesamiento bidireccional de las secuencias. La arquitectura está compuesta por tres capas de BiLSTM apiladas: la primera con 128 unidades, la segunda con 256 y la tercera nuevamente con 128 unidades. La capa de clasificación está conformada por dos capas densas de 64 y 32 unidades, respectivamente, seguidas por una capa de salida softmax (ver Figura 24(a)).

4.7.1.4. GRU

Siguiendo el enfoque propuesto por Samaan et al. (2022), se implementó un modelo basado en GRU como una alternativa computacionalmente más eficiente frente a las LSTM. La arquitectura del modelo está compuesta por tres capas GRU apiladas: la primera con 64 unidades, la segunda con 128 y la tercera con 64 unidades nuevamente. La capa de clasificación se conforma por dos capas densas de 64 y 32 unidades, respectivamente, y con una capa de salida softmax (ver Figura 24(b)).



(a) BiLSTM (b) GRU

Figura 24. Arquitecturas de modelos BiLSTM y GRU.

4.7.1.5. ResNet 1D

El modelo está conformado por una serie bloques residuales que permiten la propagación de la información a través de capas profundas mediante conexiones de salto (*skip connections*). Cada bloque residual se estructura de la siguiente manera: primero se aplica una capa de convolución de una sola dimensión (Conv1D) con $k = 128$ filtros y un *stride* de $s = 2$, seguida de una capa de normalización por lotes (*Batch Normalization*) y una función de activación ReLU. Después se repite la operación convolucional anterior y el resultado se suma con la entrada original del bloque mediante una *skip connection* para nuevamente una capa de *Batch Normalization*, seguida de una función ReLU. Finalmente, se incorpora una operación de MaxPooling1D con un tamaño de ventana 2 y un *stride* 2 para reducir la dimensión de los datos antes de pasarlos a la siguiente capa.

La red completa está compuesta por tres bloques residuales, precedidos por una capa convolucional inicial. Posteriormente, los datos son transformados en una representación unidimensional mediante una capa de aplanado (*Flatten*). Las siguientes capas después de esto constituyen a las capas de clasificación, en donde este modelo base utiliza dos capas densas de 128 unidades cada una, activadas con ReLU y acompañadas por capas de *dropout* con una tasa del 80 % para prevenir el sobreajuste. La clasificación final se realiza mediante una capa de salida softmax, cuyo número de neuronas corresponde al total de clases de señas (ver Figura 25).

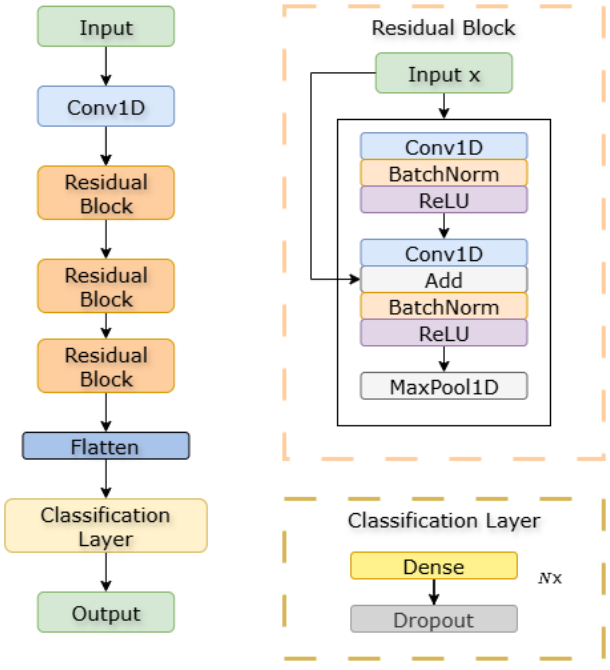


Figura 25. Arquitectura del modelo ResNet.

4.7.2. Experimentación

Con el objetivo de determinar cuál modelo presenta el mejor desempeño, se utilizó el conjunto de datos de glosas para entrenar los modelos, y se definieron tres experimentos principales. La estructura de estos experimentos se muestra en la Figura 26.

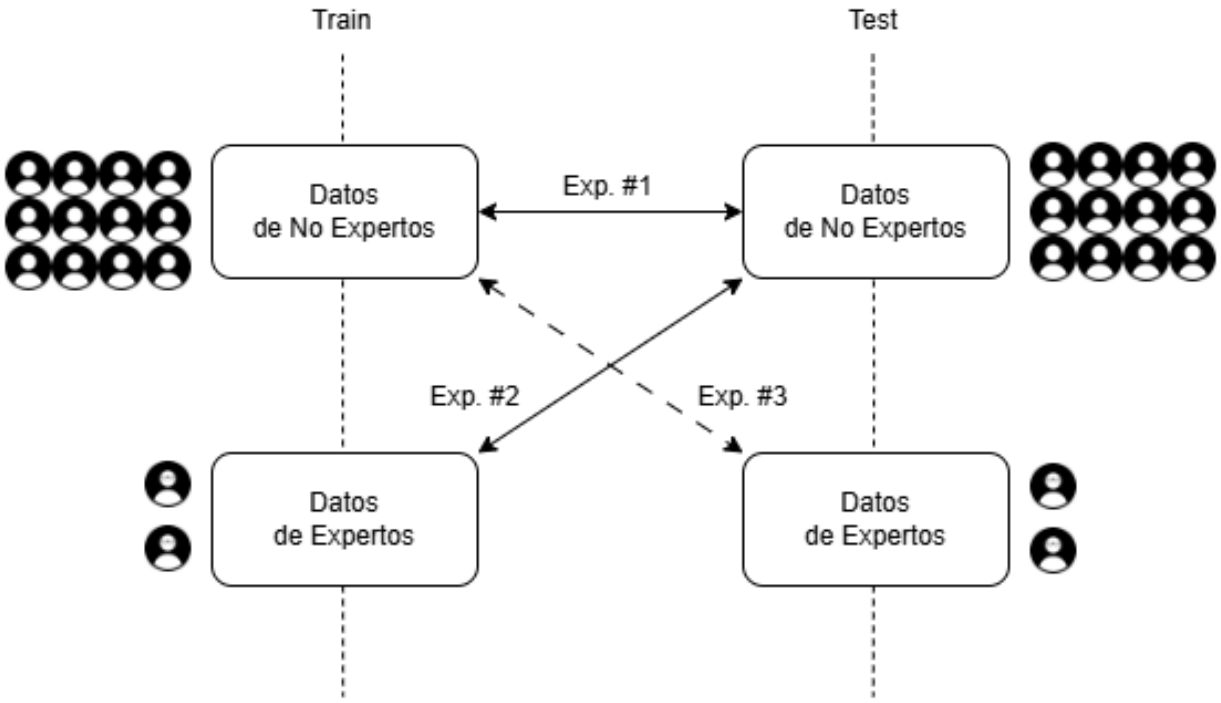


Figura 26. Esquema ilustrativo de los experimentos realizados.

- **Experimento 1:** Consiste en entrenar los modelos utilizando secuencias de señas realizadas por sujetos No Expertos, y posteriormente evaluarlos con los datos de un sujeto No Experto distinto. Este escenario busca replicar lo que comúnmente se observa en el estado del arte, donde en su mayoría, los conjuntos de datos son capturados por personas sin experiencia especializada.
- **Experimento 2:** Consiste en entrenar los modelos con los datos de los dos sujetos Expertos y evaluarlos con los datos de los sujetos No Expertos. Este experimento tiene como propósito analizar cómo se adapta el modelo cuando es expuesto a una baja variabilidad durante el entrenamiento, frente a datos provenientes de usuarios No Expertos.
- **Experimento 3:** En este caso, se entrena el modelo con datos de sujetos No Expertos y se evalúa con datos de Expertos. El objetivo es observar el nivel de ajuste que alcanzan los modelos

entrenados con un volumen amplio de datos con alta variabilidad, al ser expuestos a un conjunto de datos distinto y más estructurado.

Todos los experimentos se llevaron a cabo aplicando validación cruzada *Leave-One-Out* por sujeto, la cual consiste en excluir a un sujeto del conjunto de entrenamiento y utilizarlo exclusivamente para la fase de prueba. En los Experimentos 2 y 3, cuyos conjuntos de datos están separados según el nivel de experiencia, se realiza un único entrenamiento utilizando la totalidad de los datos del grupo correspondiente. Posteriormente, se efectúan pruebas individuales con la información de cada sujeto perteneciente al grupo contrario. Después de separar los datos de entrenamiento y prueba, aleatoriamente extraemos el 10 % de las instancias de entrenamiento para la validación durante el entrenamiento (ver Figura 27.

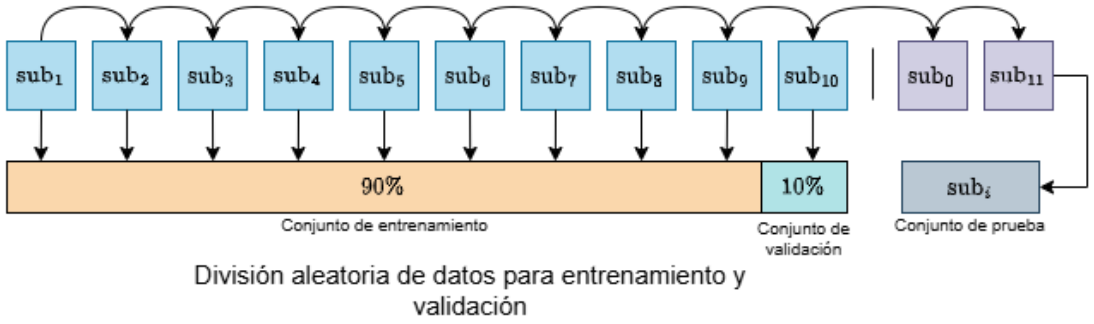


Figura 27. Esquema ilustrativo del proceso *Leave-One-Out* y la partición de datos.

4.7.3. Optimización de hiperparámetros

Una vez identificado el modelo con mejor desempeño en los experimentos, se procedió a realizar un proceso de optimización de hiperparámetros. Este consistió en una búsqueda aleatoria de combinaciones para los siguientes parámetros:

- Número de capas densas
- Número de unidades por capa
- Porcentaje de *dropout*
- Tasa de aprendizaje

Este proceso se enfocó exclusivamente en la modificación del módulo de clasificación del modelo, manteniendo intacta la arquitectura base previamente seleccionada. La optimización fue llevada a cabo mediante el uso de la herramienta *Keras Tuner* de *TensorFlow*, la cual permite explorar el espacio de hiperparámetros de manera eficiente.

Siguiendo el esquema de validación cruzada Leave-One-Out, se realizaron 10 pruebas con combinaciones aleatorias por cada iteración. Es decir, se buscaron los mejores hiperparámetros para cada división específica del conjunto de entrenamiento y prueba definida por el procedimiento de validación.

Todo este procedimiento fue necesario para obtener un modelo capaz de clasificar correctamente señas de forma aislada. Dicho modelo será posteriormente utilizado para la evaluación en la modalidad continua, la cual se describe en la siguiente sección.

4.8. Reconocimiento de señas dinámicas en modalidad continua

El reconocimiento de señas en modalidad continua, consiste en identificar cada una de las señas que aparecen dentro de una muestra, en este caso un video. En este apartado veremos el procedimiento propuesto para realizar el reconocimiento en modalidad continua, partiendo del análisis de regiones de interés, el método de adaptación para el modelo aislado, así como la manera en la que podemos evaluarlo.

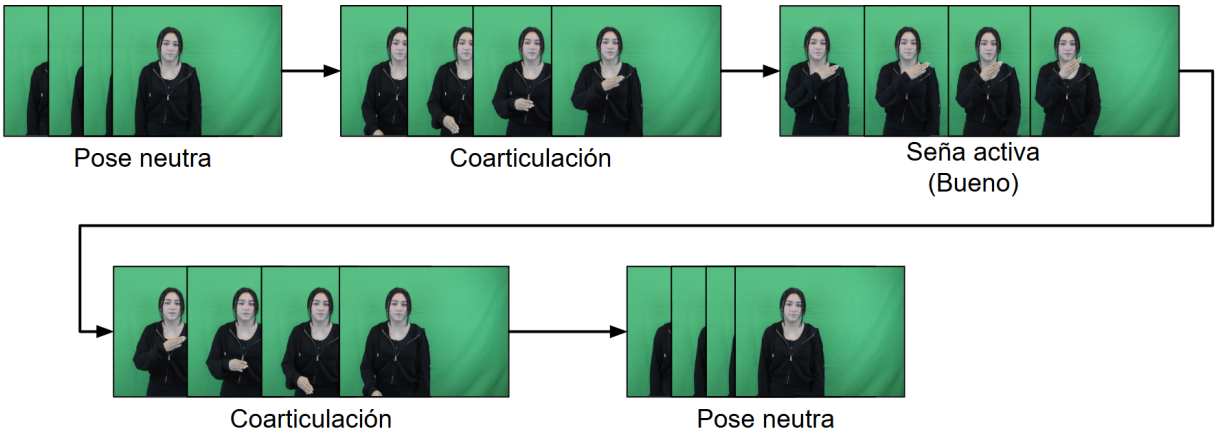
4.8.1. Regiones de interés dentro de un video a nivel temporal

Para llevar a cabo el reconocimiento continuo de señas dinámicas, se utilizó el conjunto de datos de frases capturado conforme al segundo corpus descrito en la Sección 4.1 de este capítulo. Tal como se menciona en dicha sección, los videos pueden contener entre una y cinco señas. Uno de los principales desafíos en esta modalidad es la segmentación, ya que, a diferencia del reconocimiento aislado —donde la seña inicia casi inmediatamente al comenzar el video—, en este caso se trabaja con secuencias que contienen una o más señas en ubicaciones temporales no definidas dentro del video.

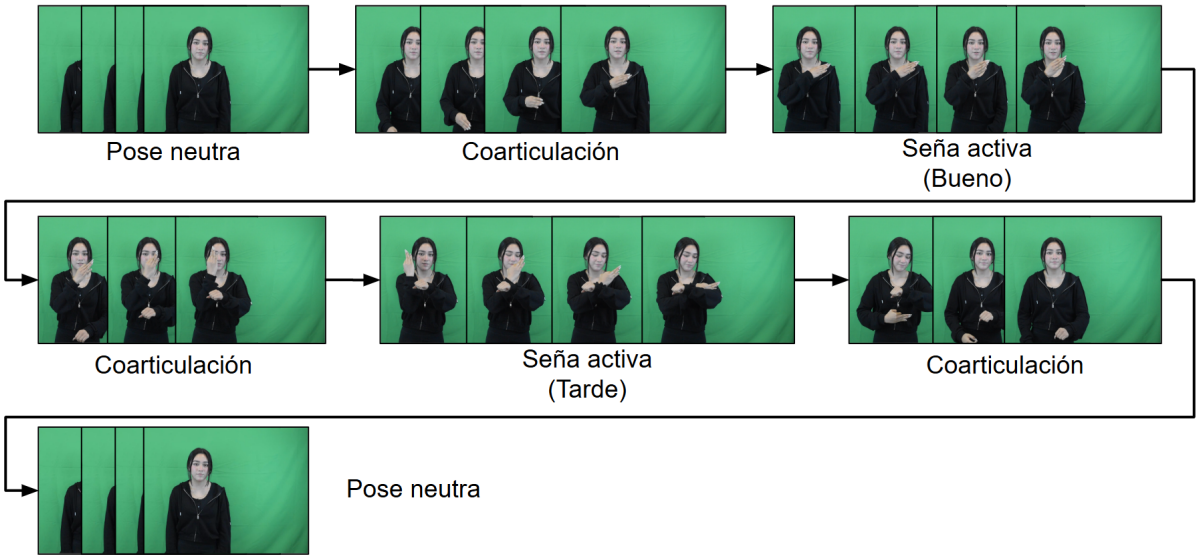
Considerando un video dentro del conjunto de datos en donde sólo incluya una seña o dos señas, partimos de los segmentos: Pose neutra → Coarticulación → Seña activa → Coarticulación → Pose Neutra (ver Figura 28(a)), para una seña; Pose neutra → Coarticulación → Seña activa → Coarticulación → Seña

activa → Coarticulación → Pose Neutra (ver Figura 28(b)), para dos señas consecutivas. Donde:

- **Pose neutra:** Representa la postura inicial y final del sujeto que realiza la seña.
- **Seña activa:** Es el segmento de mayor importancia dentro del video, dado que es la seña que buscamos reconocer.
- **Coarticulación:** Se refiere a la transición entre una seña activa y otra, o entre una pose neutra (ya sea inicial o final) y una seña activa. Este fenómeno es producto de la continuidad del movimiento en la ejecución natural de las señas.



(a) Segmentos para una seña (esta representación también aplica para la modalidad aislada).



(b) Segmentos para dos señas.

Figura 28. Ejemplos de segmentación dentro de videos de una y dos señas.

4.8.2. Desplazamiento dentro de un video de señas continuo

Para recorrer cada uno de los videos, se emplea un esquema de **ventaneo temporal**, definido por dos parámetros clave: el **tamaño de la ventana** N, que indica la cantidad de frames a considerar por segmento, y el **porcentaje de traslape** U, que establece la proporción de superposición entre ventanas consecutivas. Esta configuración, permite asegurar una cobertura adecuada de la información contenida en los videos. A través del Algoritmo 1, se generan los índices de inicio de cada ventana, de modo que, conociendo la constante N, es posible construir cada segmento temporal de forma controlada y sistemática. En la Figura 29, es posible observar una pequeña simulación del comportamiento de este ventaneo, donde cada linea azul representa una ventana con su respectivo identificador y sus índices de frames de comienzo y término, dado los parámetros: $N = 32, U = 0.5, L = 150$. Se puede observar que a mayor nivel de traslape se recorre una mayor cantidad de ventanas, pero garantizando una cobertura amplia de información. Es por esto, que una de las tareas importantes aquí es identificar el tamaño de ventana y porcentaje de traslape adecuados.

Algoritmo 1: Generación de ventanas con traslape

Input: Longitud de la secuencia L, tamaño de ventana N, traslape mínimo $U \in (0, 1]$

Output: Lista de posiciones iniciales de cada ventana

if $N > L$ **then**
    return *lista vacía*
end
$traslape \leftarrow N \cdot U$
$salto \leftarrow \lfloor N - traslape \rfloor$
if $salto < 1$ **then**
    $salto \leftarrow 1$
end
$inicios \leftarrow \emptyset$
$posicion \leftarrow 1$
while $posicion + N - 1 \leq L$ **do**
    Añadir $posicion$ a $inicios$
    $posicion \leftarrow posicion + salto$
end
return $inicios$

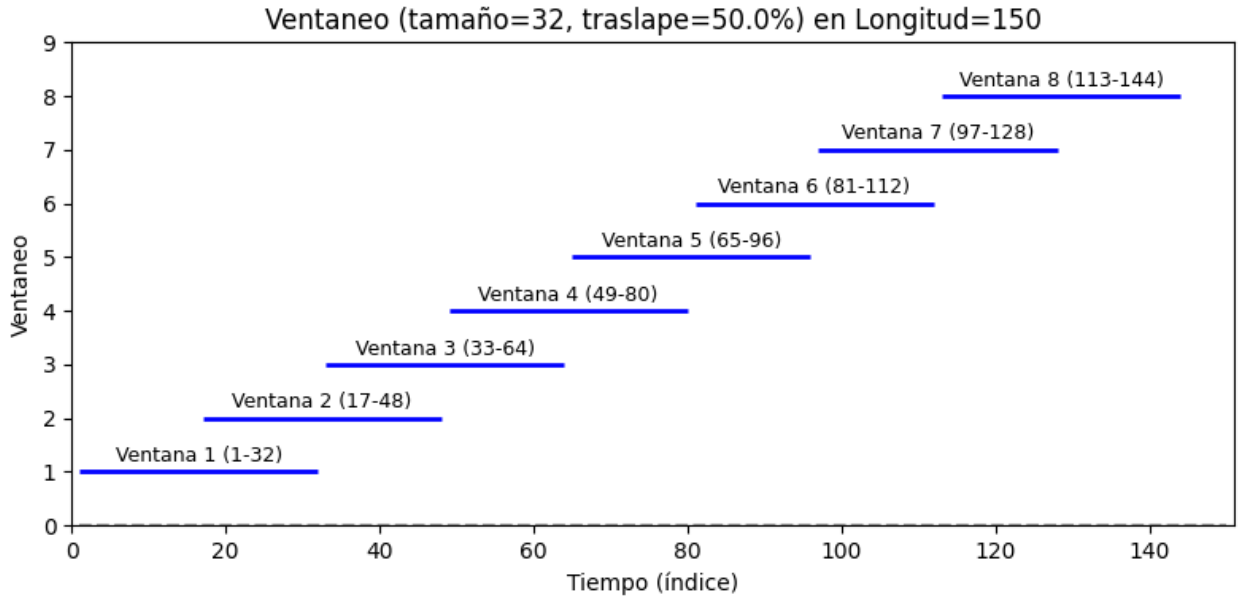


Figura 29. Gráfica ilustrativa de ventaneo dentro de un video.

4.8.2.1. Definición de la longitud de la ventana

Se llevó a cabo una búsqueda de parámetros con el fin de determinar el tamaño óptimo de la ventana y el porcentaje de traslape. Para evitar una definición arbitraria de estos valores, se realizó un muestreo pseudoaleatorio dentro del conjunto de datos de glosas aisladas. Esta estrategia se adoptó considerando que, en un entorno real, la duración de una seña puede variar entre sujetos; por lo tanto, se optó por calcular el tiempo promedio de ejecución de la seña activa.

Se seleccionaron cinco muestras aleatorias de señas activas realizadas con una sola mano y cinco muestras de señas realizadas con ambas manos, con el objetivo de obtener una media equilibrada respecto al número de manos involucradas. Una vez medido el tiempo, este fue convertido a su correspondiente longitud en fotogramas, considerando la tasa de cuadros por segundo del conjunto de datos. Por último, los porcentajes de traslape se definieron de manera arbitraria como 25 % y 50 %.

Con el propósito de determinar la mejor configuración de tamaño de ventana y procentaje de traslape, se utilizó como métrica objetivo la **métrica de aparición** (*appearance_metric*), definida como:

$$\text{appearance\_metric}(A, B) = \frac{|A \cap B|}{|B|}. \tag{19}$$

donde:

- **A** representa el conjunto de glosas únicas reconocidas por el modelo.
- **B** corresponde al conjunto de glosas únicas objetivo, es decir, aquellas que realmente están presentes en la secuencia evaluada.

Esta métrica evalúa qué proporción de las glosas verdaderamente presentes en la secuencia fueron correctamente detectadas por el modelo, sin considerar el orden o la frecuencia de aparición. Es especialmente útil en escenarios donde el reconocimiento continuo puede omitir, fusionar o fragmentar señas, permitiendo evaluar el grado de cobertura del sistema de forma global.

A partir de esta definición, se compararon múltiples combinaciones de tamaños de ventana y umbrales de probabilidad con el fin de identificar aquella que maximiza el valor de *appearance_metric*, lo cual indica una mayor recuperación de señas relevantes en la secuencia de prueba.

4.8.3. Adaptación del modelo aislado a la modalidad continua

Los modelos entrenados, en la modalidad aislada, no están preparados para afrontar adecuadamente ciertos fenómenos presentes en la modalidad continua, tales como la postura neutra, la coarticulación entre señas y la presencia de señas con movimientos suaves o prolongados. Esta última limitación se debe a la restricción de la longitud de entrada del modelo, fijada en 15 frames, como se definió en el proceso de aumento de datos descrito en la Sección 4.4.

Se consideró necesario incorporar al modelo una clase adicional denominada NEUTRO, compuesta por segmentos representativos de posturas neutras. Estos segmentos fueron extraídos del conjunto de datos de frases, seleccionando fragmentos identificados como posturas neutras sin incluir coarticulación. Esta decisión se fundamenta en el hecho de que dicho conjunto presenta secuencias con mayor longitud, lo que permite capturar de forma más precisa la presencia de posturas neutras entre señas.

Para adaptar las ventanas de mayor duración generadas en la modalidad continua, se extrae de cada una

la **secuencia base** utilizando el método *Frame Skip Sampling*, descrito en la Sección 4.4 de este capítulo. De este modo, se brinda al modelo secuencias más apegadas a las vistas durante su entrenamiento. En la Figura 30 se observa de forma ilustrativa este proceso, cabe mencionar que el preprocesamiento definido en la sección 4.6 es aplicado a la secuencia de *keypoints* extraída de la secuencia base.

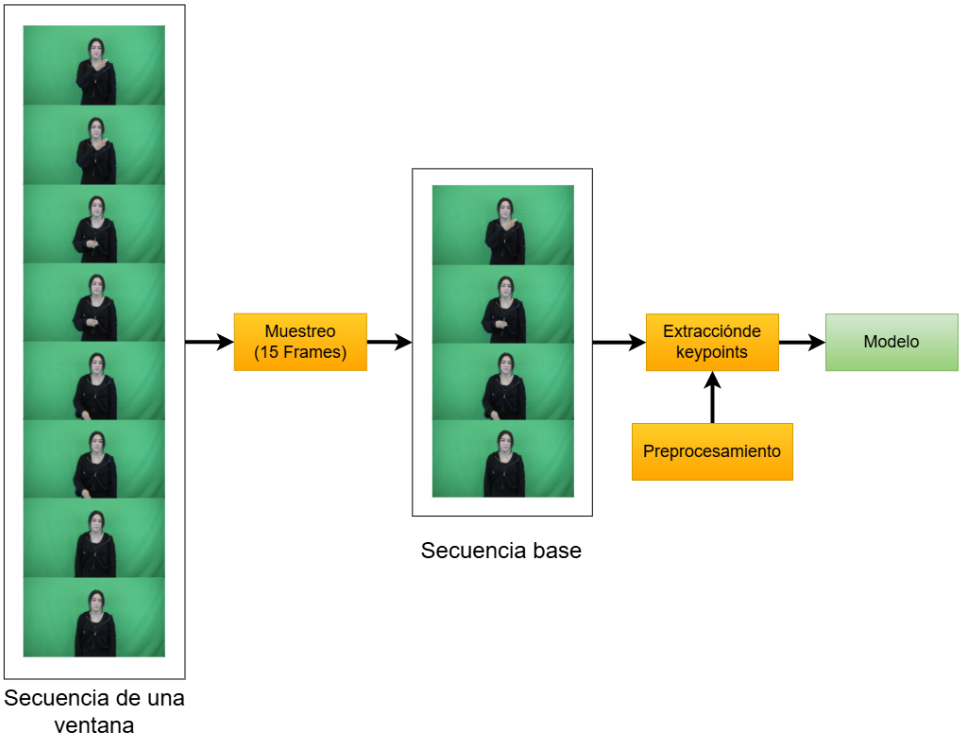


Figura 30. Diagrama ilustrativo del proceso de adaptación de la entrada para el modelo en modalidad continua.

4.8.4. Postprocesamiento de la salida

Dado que el modelo no ha sido adaptado para manejar adecuadamente el fenómeno de la coarticulación, se hace necesario aplicar un proceso de postprocesamiento orientado a depurar la salida del modelo antes de su evaluación. Para ello, es fundamental analizar el comportamiento de las predicciones a lo largo del tiempo y diseñar una estrategia que permita identificar y eliminar aquellas predicciones poco confiables atribuibles a la coarticulación.

La salida generada por el modelo para cada ventana corresponde a una distribución de probabilidad de longitud igual al número total de clases del sistema (112 clases). La predicción final se obtiene seleccionando la clase con la probabilidad más alta dentro de dicha distribución, es decir, mediante la

operación *argmax*, que indica la clase más probable dada la entrada.

El postprocesamiento consta de 3 pasos esenciales para limpiar y refinar la secuencia de glosas predichas por el modelo, buscando corregir errores comunes en el reconocimiento continuo, como repeticiones innecesarias, predicciones poco confiables y detección de clases irrelevantes.

Paso 1: Eliminación de la clase irrelevante . Se remueven todas las apariciones de la clases NEUTRO de la secuencia de salida. Su eliminación busca evitar que afecte la evaluación posterior o la segmentación de las verdaderas señas.

Paso 2: Agrupamiento de glosas repetidas. Dado a que el modelo puede predecir la misma glosa en ventanas consecutivas, estas predicciones se agrupan para consolidarlas en una única aparición. Existen dos estrategias:

- **Promedio de distribuciones:** Si se desea representar de forma balanceada la evidencia de todas las repeticiones, se promedian las distribuciones de probabilidad asociadas a cada instancia.
- **Selección de la más confiable:** Alternativamente, se conserva únicamente la ocurrencia con la mayor probabilidad máxima.

Paso 3: Filtrado por umbral de confianza. Finalmente, se aplica un filtro para eliminar glosas cuya probabilidad máxima no supera un umbral de tolerancia definida por el usuario. Este paso busca descartar predicciones poco confiables que podrían haber pasado las etapas anteriores, especialmente en presencia de ruido o ambigüedad.

Después de este proceso, obtenemos como resultado una secuencia final de glosas depuradas, acompañadas de sus respectivas distribuciones de probabilidad para ser evaluada posteriormente.

4.8.5. Generación de frases utilizando modelos de lenguaje

Como paso final, las glosas resultantes del postprocesamiento, se envían a LLMs para que generen oraciones gramaticalmente correctas en español. Los LLMs son sistemas de inteligencia artificial que pueden procesar y generar texto con comunicación coherente y generalizar a múltiples tareas (Naveed

et al., 2025). Es clave aclarar que el funcionamiento de un LLM de última generación no es abordado en esta tesis. El uso de LLMs se ve como una caja negra que sirve como postprocesamiento de un arreglo final de glosas a una frase con gramática en español.

Dado su buen desempeño en tareas de traducción entre lenguas orales-escritas, los LLMs se consideran un módulo independiente dentro de la metodología propuesta. A medida que estos modelos continúan evolucionando, es esperable que sus mejoras repercutan positivamente en la calidad de las traducciones generadas a partir de las secuencias de glosas.

Se utilizaron 3 LLMs comerciales, los cuales son *Chat-GPT 3.5-turbo*, *DeepSeek* (modelo por defecto) y *Gemini 2.0-flash*. Estos LLMs actualmente son los más utilizados para diferentes tareas, por lo que en este trabajo de propone utilizar el siguiente *prompt* para generar una oración en español a partir de una lista de glosas reconocidas: "*Eres un experto en español. A partir de una lista de glosas (palabras clave representando en señas), genera una oración completa y gramaticalmente correcta que exprese el mensaje que comunican dichas glosas.* $\backslash n$ *Glosas: [Lista de glosas]*".

Como instrucción adicional, se le pidió a los LLMs que la frase generada fuese escrita entre comillas, esto fue así, dado que los LLMs tienden a generar más información de lo solicitada, por lo que un proceso extra fue la extracción de las frases utilizando una expresión regular que extrae sólo el contenido interno entre comillas.

A modo de ejemplo hipotético, ante una secuencia de glosas como *YO*, *ANTES*, *MANEJAR*, *CARRO*, *ACCIDENTE*, se esperaría que un LLM generara una oración gramaticalmente correcta en español, como: "Sufrí un accidente automovilístico". Este tipo de salida representa la transformación ideal que se busca lograr mediante el módulo de postprocesamiento basado en lenguaje natural.

El uso de LLMs para la generación de oraciones en este trabajo responde a dos objetivos principales: (i) traducir la estructura gramatical de la LSM al español, y (ii) obtener una oración que pueda evaluarse frente a una frase de referencia, previamente traducida por expertos y disponible dentro del corpus. Para evaluar el desempeño de los LLMs, se utilizarán dos métricas principales: la **Distancia de Levenshtein Normalizada** (LDN), que se calcula como

$$\text{LDN} = \frac{\text{ED}(\hat{\mathbf{y}}, \mathbf{y})}{\text{máx}(|\hat{\mathbf{y}}|, |\mathbf{y}|)}, \tag{20}$$

donde ED representa la distancia de edición (Levenshtein) entre la oración generada $\hat{\mathbf{y}}$ y la oración de referencia $\mathbf{y}$, y la longitud mayor entre ambas secuencias se utiliza como factor de normalización; y el

Word Error Rate (WER), cuyo cálculo se describe en la Sección 2.4.2.

4.8.6. Evaluación en modalidad continua.

El reconocimiento continuo fue evaluado desde dos perspectivas, dado que se dispone de dos variables objetivo: (i) la secuencia de glosas objetivo y (ii) la frase generada en español. Para ello, se proponen dos métricas de evaluación complementarias. Por un lado, se introduce una métrica para medir la coherencia en el orden de las glosas predichas en comparación con la secuencia de referencia; por otro, se evalúa la calidad de la frase generada por los LLMs utilizando técnicas de evaluación de traducción descritas en la Sección anterior.

4.8.6.1. Métricas para evaluar presencia y orden en secuencias de glosas

En el contexto del reconocimiento continuo de señas, la evaluación del desempeño del sistema no debe limitarse únicamente a la identificación correcta de glosas individuales, sino que debe considerar también el orden en que estas aparecen dentro de la secuencia. Esto se debe a que, en lenguas de señas como la LSM, la disposición temporal de las glosas afecta directamente el significado de la frase. Por tanto, resulta necesario implementar métricas que permitan cuantificar no solo la cobertura de las glosas esperadas, sino también la precisión en su secuenciación.

A continuación, se presentan dos métricas diseñadas con dicho propósito. Por un lado, se describe el **AELVT** (*Average Error in Live Video Translation*), una métrica propuesta originalmente para evaluar el error promedio en la traducción de secuencias de video en tiempo real. Por otro lado, se introduce una métrica propuesta en este trabajo, denominada **POPE** (*Presence and Order Penalty Error*), la cual cuantifica los errores relacionados con omisiones, adiciones y desorden en las secuencias predichas de glosas. Ambas métricas resultan esenciales para evaluar de forma más realista y robusta el comportamiento de modelos en tareas de reconocimiento continuo.

La métrica **AELVT** fue propuesta por Borges-Galindo et al. (2024) con el propósito de cuantificar el error promedio en sistemas de traducción de video en tiempo real. Esta métrica se calcula en dos etapas. En primer lugar, se define el **AELV** (Average Error in Live Video) como:

$$\text{AELV} = 1 - \frac{N_W - M_{BT}}{N_W}, \tag{21}$$

donde:

- $AELV$: Error promedio en traducciones en vivo.
- N_W: Número total de palabras esperadas en la traducción.
- M_{BT}: Número de intentos incorrectos antes de producir la palabra correcta.

Luego, se obtiene el valor global de la métrica a lo largo de toda la secuencia mediante el cálculo del **AELVT** (Average Error in Live Video Translation):

$$\text{AELVT} = \frac{\sum \text{AELV}}{N_W}. \tag{22}$$

Esta medida permite evaluar la capacidad del sistema para converger rápidamente a la salida correcta, proporcionando una visión agregada del rendimiento de la traducción automática en contexto continuo.

Dado que el orden gramatical es fundamental en la Lengua de Señas Mexicana (LSM), la lista de glosas generadas por el modelo puede verse como una secuencia cuya alteración modifica el significado de la frase. Así, se vuelve necesario evaluar tanto la presencia como el orden de las glosas dentro de una predicción. Sin embargo, la literatura actual carece de métricas que cuantifiquen este comportamiento. Por tal motivo, se propone en este trabajo la métrica *Presence and Order Penalty Error*, la cual permite medir el error acumulado en la secuencia predicha considerando tanto la aparición como la posición de las glosas.

$$\text{POPE} = \frac{\sum_{i=1}^{\text{LS}} e(i)}{\text{LS}}, \quad e(i) = \begin{cases} \dfrac{|\text{Pred}_i - \text{Ref}_i|}{\text{LS} - n_{\text{app}} + 1}, & \text{si } \text{Pred}_i \in \text{Ref} \vee \text{Ref}i \in \text{Pred} \\ \dfrac{1}{\text{LS} - (||\text{Pred}|| + n_{\text{app}}) + 1}, & \text{si } \text{Pred}_i \notin \text{Ref} \vee \text{Ref}_i \notin \text{Pred} \end{cases} \tag{23}$$

donde:

- Pred_i: Glosa en la posición i de la predicción.

- Ref$i$: Glosa en la posición i de la referencia.
- LS $= max(||\text{Pred}||, ||\text{Ref}||)$: Longitud de la secuencia más larga
- $n_{\text{app}} = ||\text{Ref} \cap \text{Pred}||$: Número de glosas coincidentes entre predicción y referencia.
- $e(i)$: Penalización asociada a cada posición según pertenencia y posición relativa.
- $||\cdot||$: Longitud de una secuencia.

La métrica penaliza de forma diferenciada los errores por omisión, adición y orden, otorgando un valor entre 0 y 1, donde 0 representa una predicción perfecta en presencia y orden, y valores cercanos a 1 reflejan errores severos. Esta propuesta resulta adecuada para evaluar secuencias de glosas donde el significado depende tanto del contenido como del orden temporal.

Capítulo 5. Resultados

En este capítulo se presentan los resultados obtenidos a lo largo de las distintas etapas del estudio, incluyendo el análisis exploratorio del corpus, la extracción de características y los estudios relacionados con pérdida de datos y variabilidad. Asimismo, se reportan los desempeños alcanzados en las modalidades aislada y continua, así como los resultados del proceso de traducción de glosas reconocidas a frases en español mediante el uso de un LLM.

5.1. Características de los conjuntos de datos

En la Tabla 2 se presenta una descripción comparativa entre los conjuntos de datos utilizados para el reconocimiento en modalidad aislada (glosas) y modalidad continua (frases). Estos conjuntos presentan diferencias significativas, principalmente en que ninguno de los sujetos se repite entre ambas bases de datos, lo cual introduce una gran variabilidad entre los sujetos. Además, en el conjunto de frases no participan expertos, sin embargo, los sujetos fueron sensibilizados en la realización de señas (tomaron cursos de lengua de señas), por lo que destaca aún más la variabilidad en cuanto a movimiento, rapidez y coarticulación de señas. Desde otra perspectiva, dicha variabilidad puede considerarse beneficiosa, ya que amplía la cobertura del comportamiento gestual y contribuye al desarrollo de modelos de aprendizaje de máquina más robustos y generalizables.

Tabla 2. Descripción comparativa de los conjuntos de datos de glosas y frases.

Característica	Glosas	Frases
Datos generales		
Número de sujetos	10	8
Número de expertos	2	0
Número de clases	121	96
Número de repeticiones	1	3
Número de videos	1,415	2,304
Distribución temática		
Emergencias	24.0 % (n=29)	39.6 % (n=38)
Contexto médico	41.3 % (n=50)	38.6 % (n=37)
Cotidianas	9.1 % (n=11)	21.8 % (n=21)
Otras	25.6 % (n=31)	0.0 % (n=0)

El número de señas incluidas en el conjunto de glosas supera al número de frases por 28 clases, lo que garantiza una mejor cobertura al momento de aplicar un modelo entrenado con datos aislados sobre

secuencias continuas. Como se mencionó en la Sección 4.1.1, el corpus está enfocado en tres temáticas principales: **emergencias**, **contexto médico** y **temas cotidianos**. Dentro del corpus de glosas, se observa una distribución temática compuesta por cuatro categorías:

- **Emergencias**: Representa el 24.0 % del total (29 glosas), con señas relacionadas con situaciones críticas como *"EXPLOSION"*, *"FUEGO"*, *"GOLPE"*, entre otras.
- **Contexto médico**: Es la categoría predominante, con un 41.3 % (50 glosas), e incluye palabras como *"CANCER"*, *"CORAZON"*, *"MEDICINA"*, etc.
- **Cotidianas**: Representa un 9.1 % (11 glosas), conformada por expresiones comunes como *"GRACIAS"*, *"NO"*, *"POR_FAVOR"*, *"SI"*, entre otras.
- **Otros**: Se incluyó una cuarta categoría con un 25.6 % (31 glosas), correspondiente a señas cuya interpretación depende en gran medida del contexto. Aquí se agrupan elementos como *"NO_ENTENDER"*, *"MI"*, *"YO"*, *"LLAMAR"*, que podrían formar parte de más de una temática o funcionar como elementos gramaticales.

En el conjunto de frases, la distribución se concentra en tres categorías claramente definidas: emergencias (38 frases, 39.6 %), contexto médico (37 frases, 38.6 %) y cotidianas (21 frases, 21.8 %). A diferencia del conjunto de glosas, en este no se identificó ninguna frase interpretativa o de temática múltiple.

5.2. Elección del conjunto de características

Para seleccionar el grupo de características más confiable para la representación del cuerpo, se calculó la pérdida correspondiente a cada *framework* de extracción de *keypoints* y se realizó una comparación de pérdida de datos entre ellos. Esta comparación se enfocó en tres regiones corporales: el cuerpo, la mano izquierda y la mano derecha.

En la Figura 31 se presentan los diagramas de caja que ilustran la distribución de la tasa de pérdida para cada *framework* por región considerando todos los *keypoints* recuperados. Como se puede observar, *MediaPipe* presenta una tasa de pérdida menor en todas las regiones evaluadas. Esta observación se corrobora en la Tabla 3, donde se resume la media y desviación estándar de pérdida por grupo anatómico.

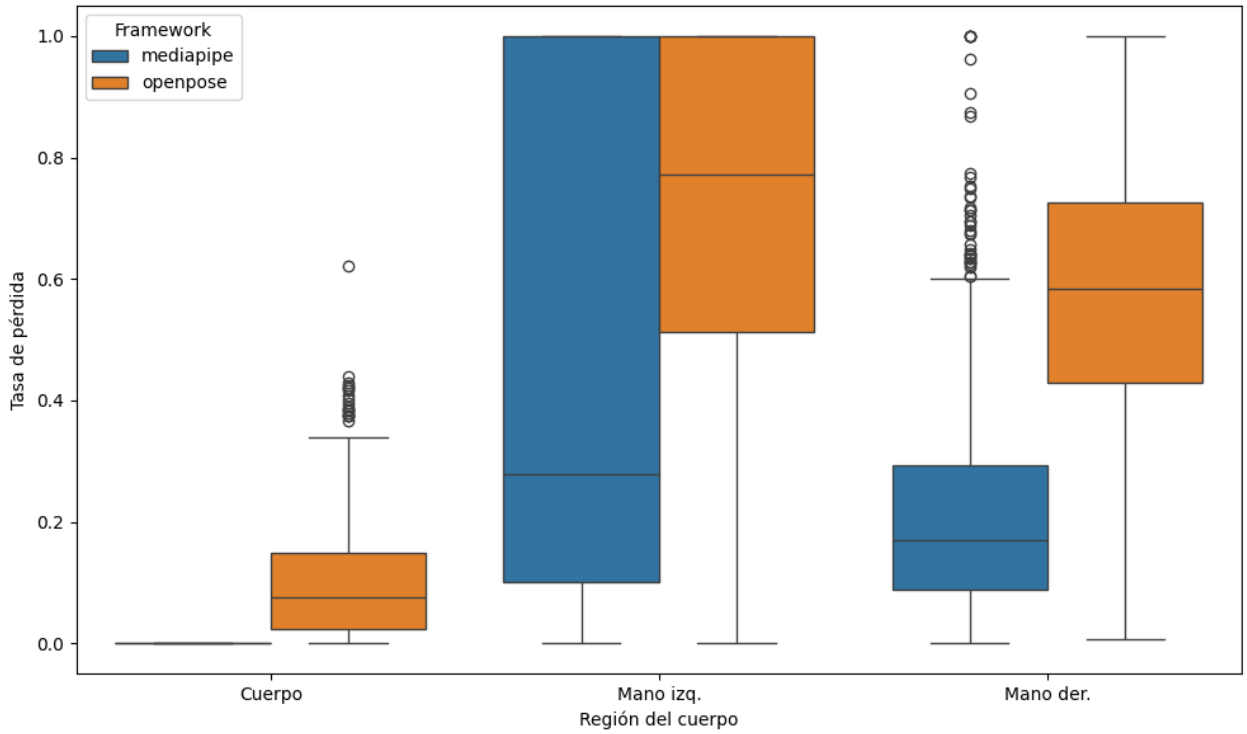


Figura 31. Comparación de perdida de datos entre *frameworks* enfocado a las regiones del cuerpo. Note que el diagrama de cajas presenta la mediana.

En términos de procentajes, *MediaPipe* no reportó pérdida alguna en la región del cuerpo (0.000 ± 0.000), mientras que *OpenPose* presentó una pérdida promedio del 9.6 % (± 8.8 %). En la región de la mano izquierda, *MediaPipe* obtuvo una media de pérdida de 48.5 % (± 42.2 %), frente a un 72.9 % (± 27.9 %) para *OpenPose*. De forma similar, en la mano derecha *MediaPipe* presentó una pérdida menor (20.7 % ± 16.6 %) en comparación con *OpenPose* (57.3 % ± 20.5 %).

Estos resultados permiten concluir que, en términos de robustez frente a la pérdida de información, *MediaPipe* ofrece una mayor confiabilidad, especialmente en la región del cuerpo. Es importante resaltar que, en los últimos años, Google ha continuado con el desarrollo de este *framework*, el cual puede recomendarse como la mejor opción para estudios de seguimiento de movimiento en el contexto de la lengua de señas. Por esta razón, estos datos fueron utilizados posteriormente para el entrenamiento de los modelos.

Tabla 3. Resumen de tasa de pérdida por región y *framework* (media ± desviación estándar)

Framework	Cuerpo	Mano Izquierda	Mano Derecha
Mediapipe	0.000 ± 0.000	0.485 ± 0.422	0.207 ± 0.166
OpenPose	0.096 ± 0.088	0.729 ± 0.279	0.573 ± 0.205

5.3. Aumento de datos y estudio de similitud

Dado que se cuenta con aproximadamente un video por seña y por sujeto dentro del conjunto de glosas, resultó conveniente emplear la técnica de *Frame Skip Sampling* para aumentar el número de muestras de 1 a 50. Tras aplicar este proceso, se generaron un total de 70,750 muestras de señas, y se añadieron 1,200 muestras correspondientes a la clase "NEUTRO". Esto da como resultado un conjunto de datos compuesto por 71,950 muestras, cada una con una longitud de $l = 15$ frames, utilizado posteriormente para el entrenamiento de modelos.

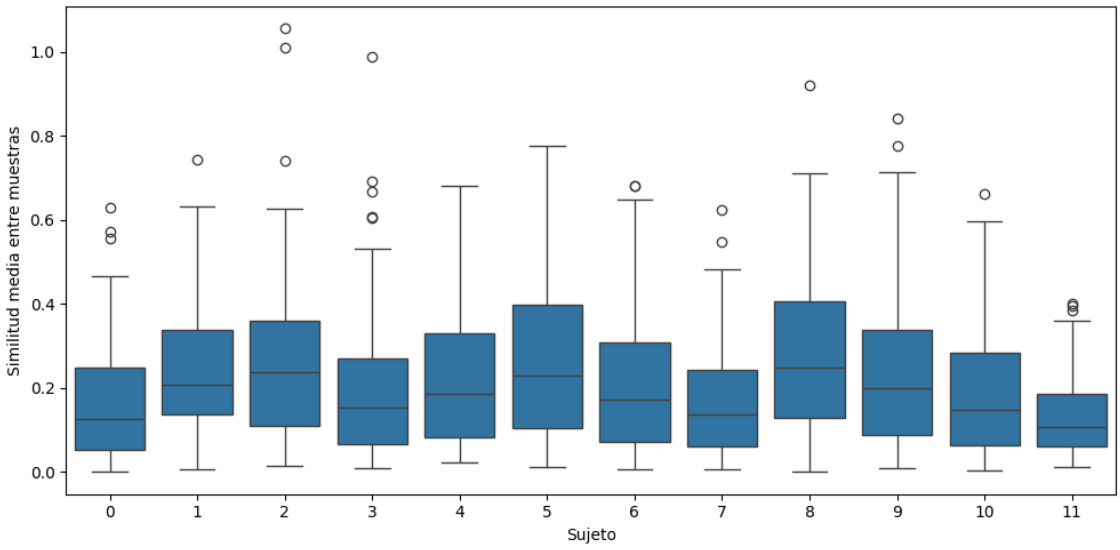


Figura 32. Comparación de similitud entre muestras aumentadas por sujetos

En el estudio de similitud entre muestras aumentadas, la medición se realizó de forma intra-glosa e intra-sujeto, i.e., se compararon únicamente muestras pertenecientes a la misma glosa y al mismo sujeto. Cabe recordar que un valor de similitud (calculada por la Ecuación 15) es cercano a 0 indica una alta similitud entre muestras. La Figura 32 presenta la distribución de la similitud entre muestras aumentadas por sujeto. En las muestras aumentadas de los sujetos 0, 7 y 11 presentan una menor distancia intra-sujeto, lo que sugiere una mayor similitud entre las muestras generadas. Por otro lado, los sujetos 2, 5, y 8 muestran una mayor distancia intra-sujeto, lo cual indica que sus muestras aumentadas son más diversas entre sí.

Esta observación se refuerza con los resultados resumidos en la Tabla 4, donde se presenta la media y desviación estándar de la similitud para cada sujeto. El sujeto con menor similitud fue 8 (0.280 ± 0.189),

seguido de 5 (0.265 ± 0.188). Por el contrario, los sujetos con mayor similitud fueron 11 (0.134 ± 0.102) y 0 (0.166 ± 0.145). Cabe destacar que estos dos sujetos corresponden a los expertos en lengua de señas, lo cual refuerza la hipótesis de que la experiencia influye en la consistencia de los datos aumentados.

Tabla 4. Similitud media entre muestras aumentadas por sujeto (media $\pm$ desviación estándar)

Sujeto	Similitud
0	0.166 $\pm$ 0.145
1	0.245 $\pm$ 0.149
2	0.257 $\pm$ 0.190
3	0.202 $\pm$ 0.173
4	0.229 $\pm$ 0.173
5	0.265 $\pm$ 0.188
6	0.210 $\pm$ 0.168
7	0.171 $\pm$ 0.136
8	0.280 $\pm$ 0.189
9	0.252 $\pm$ 0.194
10	0.192 $\pm$ 0.154
11	0.134 $\pm$ 0.102

Al realizar un análisis más exhaustivo de las similitudes, en la Figura 33 se presenta un mapa de calor que representa la similitud media entre las muestras aumentadas. Cada celda corresponde al valor promedio de similitud calculado para una glosa específica y un sujeto determinado. Es importante señalar que la escala empleada varía de 0 a 1, pero representa una distancia euclidiana, por lo que no corresponde a una proporción normalizada.

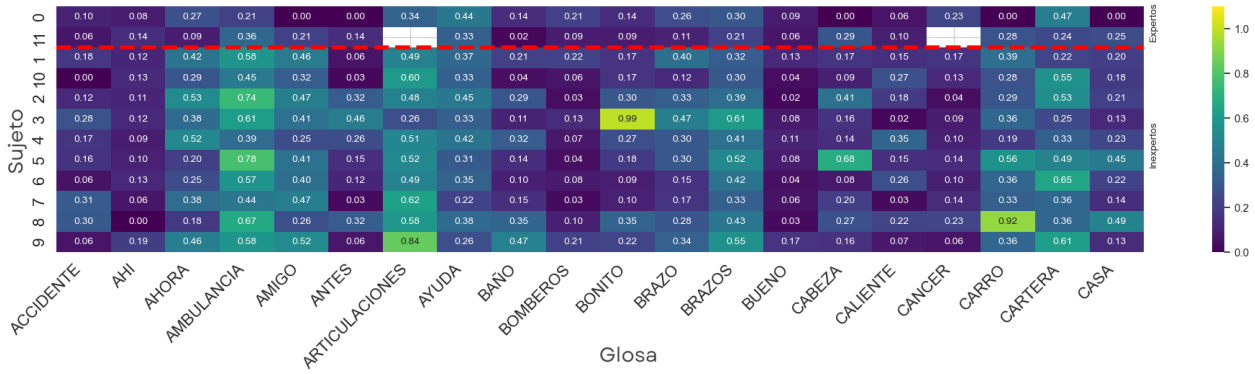
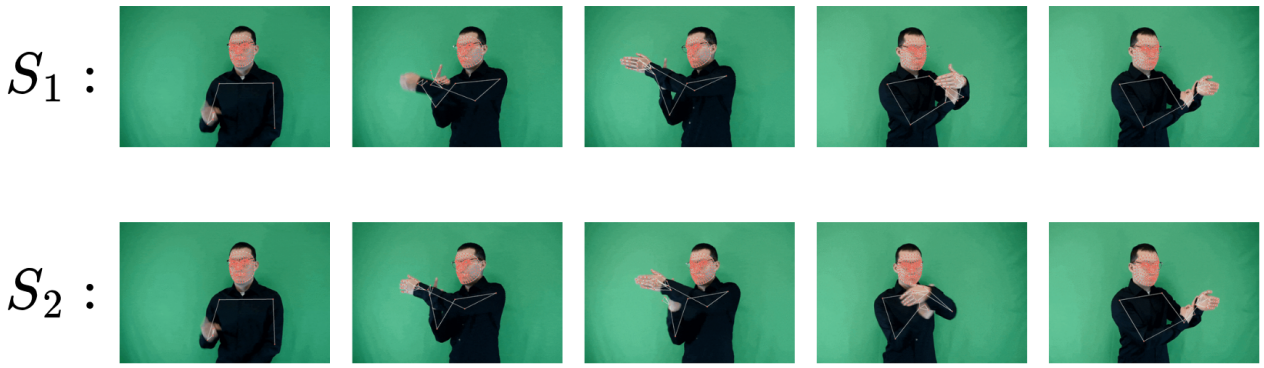


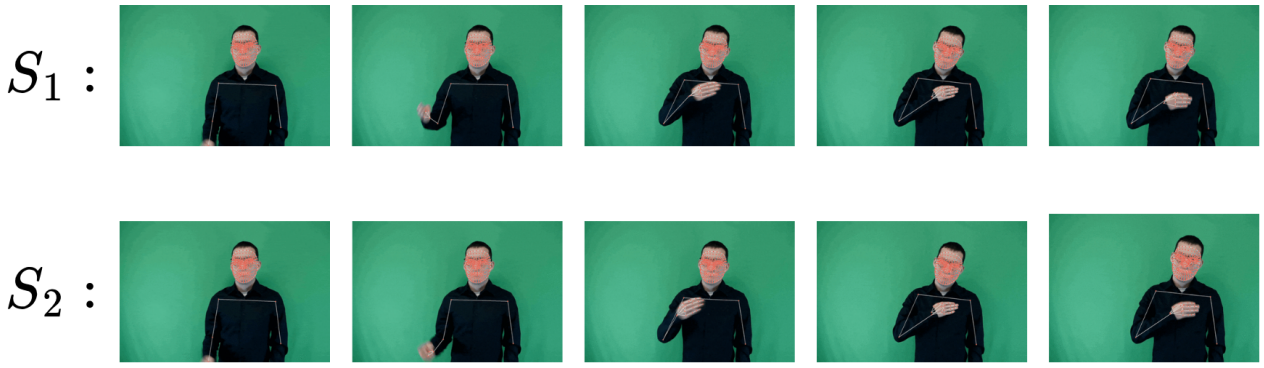
Figura 33. Mapa de calor de similitud por muestra y por sujeto (20 glosas)

El mapa de calor muestra únicamente las primeras 20 señas ordenadas alfabéticamente, las cuales reflejan casos relevantes que se repiten de forma representativa a lo largo del conjunto completo de 121 glosas. Por ejemplo, señas como "AMBULANCIA" (ver Figura 34(a)) tienden a presentar bajos niveles de

similitud debido a su naturaleza dinámica, independientemente del sujeto que las ejecute. En contraste, glosas como "BUENO" (ver Figura 34(b)) presentan un comportamiento opuesto, ya que al involucrar un movimiento más contenido, tienden a generar muestras con mayor consistencia y, por ende, mayor similitud.



(a) Fotogramas de "AMBULANCIA"



(b) Fotogramas de "BUENO"

Figura 34. Fotogramas representativos de glosas con diferentes niveles de variabilidad entre muestras aumentadas.

Este comportamiento respalda la observación previamente establecida: la experiencia del sujeto tiene una influencia directa en la coherencia del movimiento ejecutado, lo cual se refleja en una mayor similitud entre las muestras aumentadas. Es decir, los sujetos con mayor dominio de la lengua de señas tienden a realizar movimientos más consistentes y estructurados, lo que da como resultado secuencias sintéticas más homogéneas. Sin embargo, desde la perspectiva del aumento de datos, una mayor variabilidad entre muestras suele ser deseable para generar ejemplos más diversos y representativos. No obstante, esta relación entre experiencia y estabilidad en las muestras requiere ser explorada con mayor profundidad. Sería pertinente realizar un análisis complementario que considere otros factores como la complejidad gestual de cada seña, las trayectorias de movimiento y la precisión de los sistemas de captura, con el fin de caracterizar más rigurosamente el impacto de la experiencia en la generación de datos fiables.

5.4. Resultados en la modalidad aislada

En esta sección se presentan los resultados obtenidos en la modalidad de reconocimiento aislado de señas, en la que cada muestra corresponde a una única seña previamente segmentada. El análisis se centra en evaluar el rendimiento de distintos modelos de aprendizaje profundo bajo diversas condiciones de entrenamiento y prueba. En particular, se examina el impacto que tienen factores como la experiencia del sujeto y la arquitectura del modelo, sobre el desempeño de los clasificadores. Los experimentos realizados permiten identificar qué configuraciones son más efectivas para lograr una clasificación precisa y robusta en esta modalidad.

5.4.1. Experimentación según el nivel de experiencia

Dado que los sujetos que producen las señas pueden presentar diferentes niveles de experiencia en lengua de señas, se planteó una serie de experimentos para analizar cómo esta variabilidad influye en el rendimiento de los modelos. Para ello, se diseñaron tres escenarios experimentales en los que se controla el tipo de sujetos incluidos durante el entrenamiento y la evaluación. Esta experimentación busca responder si los modelos generalizan mejor cuando son expuestos a datos con alta variabilidad (como los de usuarios no expertos), o si se benefician más al ser entrenados con señales consistentes provenientes de usuarios expertos.

5.4.1.1. Experimento 1

El primer experimento fue diseñado como un escenario equivalente al que comúnmente se emplea en el estado del arte. En él, se recolectaron y procesaron datos exclusivamente de sujetos no expertos para el entrenamiento de modelos, los cuales fueron posteriormente evaluados utilizando datos inéditos también provenientes de sujetos no expertos. Los resultados, presentados en las Tablas 6 y 5, permiten observar diferencias claras entre los modelos evaluados tanto en términos de eficiencia computacional como en desempeño predictivo.

En cuanto a las métricas de desempeño (ver Tabla 5), el modelo *ResNet* obtuvo la mayor exactitud

de prueba (0.759 $\pm$ 0.177), así como una sensibilidad de 0.759 $\pm$ 0.177, una especificidad de 0.998 $\pm$ 0.002 y un F1-score de 0.717 $\pm$ 0.206. Aunque modelos como *BiLSTM* y *GRU* presentaron resultados comparables, *ResNet* mostró un desempeño más equilibrado en el conjunto de métricas evaluadas.

Tabla 5. Experimento 1, comparación de modelos: métricas de desempeño

Modelo	Test Acc	Especificidad	Sensibilidad	F1-score
BiLSTM (Samaan et al., 2022)	0.677±0.180	0.997±0.002	0.677±0.180	0.631±0.199
GRU (Samaan et al., 2022)	0.682±0.185	0.997±0.002	0.682±0.185	0.636±0.202
LSTM (Samaan et al., 2022)	0.688±0.199	0.997±0.002	0.688±0.199	0.643±0.216
LSTM	0.549±0.148	0.996±0.001	0.549±0.148	0.492±0.153
RNN simple	0.601±0.190	0.996±0.002	0.601±0.190	0.554±0.198
Resnet	0.759±0.177	0.998±0.002	0.759±0.177	0.717±0.206

Por otro lado, en términos de eficiencia computacional (ver Tabla 6), *ResNet* también se destacó por su bajo costo temporal, con un tiempo de entrenamiento de 254 ± 1.749 segundos y el menor tiempo de inferencia entre los modelos evaluados (0.435 ± 0.008 segundos). En contraste, modelos como *BiLSTM* y *LSTM* estándar requirieron tiempos significativamente mayores. Este balance entre rendimiento y eficiencia computacional sustenta su selección como modelo base para los siguientes experimentos.

Tabla 6. Experimento 1, comparación de modelos: tiempo de entrenamiento y prueba.

Modelo	Tiempo Train (s)	Tiempo Test (s)
BiLSTM (Samaan et al., 2022)	547±2.929	1.216±0.005
GRU (Samaan et al., 2022)	234±2.833	0.596±0.014
LSTM (Samaan et al., 2022)	253±1.301	0.595±0.008
LSTM	759±6.583	0.719±0.017
RNN Simple	315±2.579	0.521±0.015
ResNet	254±1.749	0.435±0.008

5.4.1.2. Experimento 2

Este experimento evalúa la capacidad de generalización de los modelos entrenados únicamente con datos de sujetos expertos cuando se enfrentan a muestras de usuarios no expertos. En cuanto al desempeño (ver Tabla 7), los modelos recurrentes muestran una reducción en su exactitud: *BiLSTM*, *GRU* y *LSTM*

se sitúan por debajo del 20 % (por ejemplo, *BiLSTM* con 0.130 ± 0.034 y *GRU* con 0.174 ± 0.029), lo que indica una limitada capacidad de generalización ante la variabilidad añadida. En contraste, *ResNet* logra una exactitud de prueba de 0.251 ± 0.048, acompañada del mismo valor de sensibilidad y un F1-score de 0.198 ± 0.041, por lo que se mantiene como la opción más consistente dentro de este conjunto experimental.

Tabla 7. Experimento 2, comparación de modelos: métricas de desempeño

Modelo	Test Acc	Especificidad	Sensibilidad	F1-score
BiLSTM (Samaan et al., 2022)	0.130±0.034	0.992±0.000	0.130±0.034	0.098±0.027
GRU (Samaan et al., 2022)	0.174±0.029	0.993±0.000	0.174±0.029	0.130±0.022
LSTM (Samaan et al., 2022)	0.182±0.041	0.993±0.000	0.182±0.041	0.142±0.033
LSTM	0.185±0.033	0.993±0.000	0.185±0.033	0.141±0.029
RNN Simple	0.175±0.042	0.993±0.000	0.175±0.042	0.127±0.038
ResNet	0.251±0.048	0.993±0.000	0.251±0.048	0.198±0.041

En términos de eficiencia computacional (ver Tabla 8), los modelos basados en *GRU* y *RNN Simple* destacan por sus reducidos tiempos de entrenamiento (56 s y 66 s, respectivamente) y por tiempos de inferencia igualmente bajos. No obstante, *ResNet* registra el menor tiempo de prueba (0.456 ± 0.016 s) con un entrenamiento de aproximadamente 60 s, lo que refuerza su balance entre rendimiento y costo computacional.

Tabla 8. Experimento 2, comparación de modelos: tiempo de entrenamiento y prueba

Modelo	Tiempo Train (s)	Tiempo Test (s)
BiLSTM (Samaan et al., 2022)	117±0.0	1.200±0.013
GRU (Samaan et al., 2022)	51±0.0	0.592±0.018
LSTM (Samaan et al., 2022)	56±0.0	0.590±0.007
LSTM	161±0.0	0.713±0.016
RNN Simple	66±0.0	0.522±0.014
ResNet	60±0.0	0.456±0.016

Estos resultados sugieren que las arquitecturas convolucionales como *ResNet* son más robustas ante cambios en la distribución de los datos, especialmente cuando se entrenan con datos limpios o de baja variabilidad y se evalúan con datos ruidosos o altamente variables. No obstante, es importante señalar que el proceso de entrenamiento se llevó a cabo con un volumen reducido de datos, dado que únicamente se contó con la participación de dos sujetos expertos.

5.4.1.3. Experimento 3

De modo contrario, este experimento evalúa la capacidad de los modelos entrenados con datos provenientes de sujetos no expertos para generalizar frente a muestras realizadas por sujetos expertos. Este escenario permite analizar el comportamiento de modelos expuestos a alta variabilidad durante el entrenamiento, cuando se enfrentan posteriormente a datos más estructurados.

En cuanto a las métricas de desempeño (ver Tabla 9), el modelo *ResNet* presentó nuevamente los resultados más consistentes dentro del conjunto evaluado. Alcanzó una exactitud de prueba de 0.396 $\pm$ 0.015, superando al resto de los modelos, cuyos valores oscilaron entre 0.274 y 0.336. Además, reportó la mayor sensibilidad (0.396 $\pm$ 0.015) y un F1-score de 0.301 $\pm$ 0.015, lo que refleja un mejor equilibrio entre precisión y recuperación de clases, incluso al haber sido entrenado con datos más variables.

Tabla 9. Experimento 3, comparación de modelos: métricas de desempeño

Modelo	Test Acc	Especificidad	Sensibilidad	F1-score
BiLSTM Samaan et al. (2022)	0.336±0.023	0.994±0.000	0.336±0.023	0.245±0.028
GRU Samaan et al. (2022)	0.301±0.021	0.994±0.000	0.301±0.021	0.222±0.002
LSTM Samaan et al. (2022)	0.331±0.028	0.994±0.000	0.331±0.028	0.243±0.032
LSTM	0.274±0.017	0.994±0.000	0.274±0.017	0.199±0.024
RNN Simple	0.299±0.007	0.994±0.000	0.299±0.007	0.224±0.020
ResNet	0.396±0.015	0.995±0.000	0.396±0.015	0.301±0.015

Respecto a la eficiencia computacional (ver Tabla 10), *ResNet* se mantiene como uno de los modelos más competitivos, con un tiempo de entrenamiento de 286 segundos y el menor tiempo de inferencia registrado (0.408 $\pm$ 0.035 segundos). En contraste, modelos recurrentes como *BiLSTM* y *LSTM* presentan tiempos de entrenamiento considerablemente más altos, alcanzando hasta 845 segundos, lo que implica un mayor costo computacional.

Tabla 10. Experimento 3, comparación de modelos: tiempo de entrenamiento y prueba

Modelo	Tiempo Train (s)	Tiempo Test (s)
BiLSTM Samaan et al. (2022)	606±0.0	1.125±0.136
GRU Samaan et al. (2022)	269±0.0	0.520±0.073
LSTM Samaan et al. (2022)	279±0.0	0.542±0.058
LSTM	845±0.0	0.658±0.081
RNN Simple	345±0.0	0.516±0.073
ResNet	286±0.0	0.408±0.035

No obstante, aunque el modelo *ResNet* se posiciona como una arquitectura robusta en este conjunto experimental, los resultados ponen de manifiesto la necesidad de profundizar en el análisis del papel que juega la experiencia del sujeto. En particular, es relevante investigar por qué modelos entrenados con datos altamente variables —que en principio deberían favorecer la generalización— pueden presentar dificultades al enfrentarse a muestras más estructuradas y precisas. Este comportamiento sugiere que, además de la variabilidad, factores como la coherencia del movimiento y la calidad de las muestras tienen un impacto determinante en el desempeño del modelo.

A partir de los resultados obtenidos en los tres experimentos, la arquitectura *ResNet* mostró un comportamiento estable y favorable en todos los escenarios analizados, tanto en precisión como en eficiencia computacional. Su capacidad para adaptarse a distintas condiciones de entrenamiento y evaluación, así como su tolerancia ante la variabilidad de los datos, la convierte en una opción adecuada para continuar con la siguiente fase del estudio. Por ello, se eligió *ResNet* como modelo base para llevar a cabo la optimización de hiperparámetros, con el objetivo de mejorar su desempeño.

5.4.2. Optimización del mejor modelo

Se realizó un proceso de optimización de hiperparámetros enfocado exclusivamente en su módulo de clasificación, cuyo objetivo fue maximizar el desempeño del mejor modelo medido mediante la métrica *F1-score*, considerada más representativa al balancear precisión y sensibilidad en contextos de clasificación multiclase. Esta etapa se llevó a cabo utilizando la metodología definida en el Experimento 1 siguiendo un esquema *Leave-One-Out*.

La Tabla 11 resume los mejores conjuntos de hiperparámetros obtenidos por cada iteración del procedimiento Leave-One-Out. Se observa una notable variabilidad en las combinaciones óptimas por sujeto, lo cual es consistente con las diferencias en la ejecución de las señas.

Entre los resultados, destaca el sujeto 4 con un *F1-score* de **0.925** y una pérdida de **0.210**, correspondiente a la configuración con tres capas densas de tamaños 256, 128 y 64 unidades respectivamente, y tasas de *dropout* de 0.4, 0.2 y 0.6. Esta configuración, junto con una tasa de aprendizaje de 0.0001, ofreció el mejor balance entre generalización y precisión en el reconocimiento de señas. Otras configuraciones con desempeño notable incluyen los sujetos 6 y 7, cuyos *F1-score* se mantuvieron por encima de 0.89. En contraste, el desempeño más bajo fue observado en el sujeto 2, con un *F1-score* de 0.718

y una pérdida elevada de 1.305, lo que sugiere mayor dificultad para aprender patrones consistentes en ese caso específico.

En función de los resultados obtenidos, el modelo optimizado correspondiente al sujeto 4 será seleccionado como base para la transición hacia la modalidad continua. Esta decisión se fundamenta en que dicho modelo alcanzó el mayor desempeño global, con un **F1-score de 0.925** y una pérdida mínima de **0.210**, lo cual refleja una capacidad destacada para discriminar entre clases de señas en un entorno de reconocimiento aislado.

Tabla 11. Mejores hiperparámetros y su desempeño por sujeto

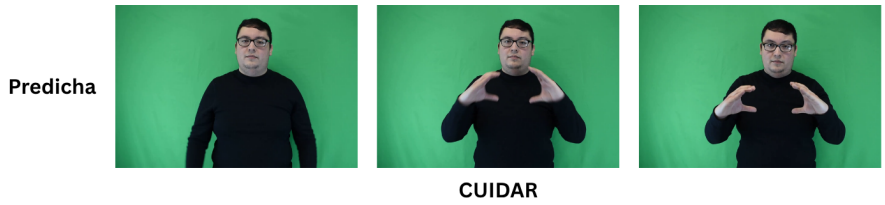
Sujeto	# Capas	Dense 1	Dropout 1	LR	Dense 2	Dropout 2	Dense 3	Dropout 3	F1	Loss
1	3	256	0.6	0.0001	256	0.5	128	0.1	0.855	0.364
2	3	256	0.5	0.0001	64	0.2	64	0.2	0.718	1.305
3	3	64	0.3	0.0001	128	0.5	256	0.2	0.885	0.378
4 ✓	3	256	0.4	0.0001	128	0.2	128	0.6	**0.925**	**0.210**
5	3	128	0.5	0.0001	64	0.2	128	0.8	0.887	0.384
6	2	64	0.2	0.0001	128	0.1	-	-	0.890	0.297
7	3	64	0.7	0.0001	256	0.0	256	0.8	0.892	0.337
8	1	256	0.4	0.0001	-	-	-	-	0.777	0.606
9	3	128	0.0	0.0001	256	0.1	64	0.3	0.866	0.394
10	3	128	0.0	0.0010	64	0.3	64	0.7	0.766	0.900

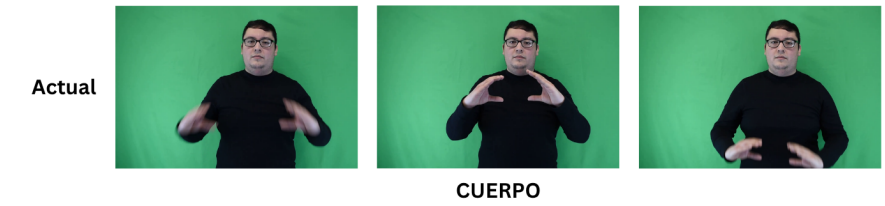
5.4.3. Análisis a detalle de los resultados

En esa sección se analizan las glosas con menor desempeño obtenido por el modelo optimizado. Entre las clases con peor rendimiento se encuentran OPRESIÓN_EN_PECHO, BUENO, CUERPO y TEMPERATURA, todas con valores de F1-score y sensibilidad de 0.000, lo que indica que el modelo no logró reconocer correctamente ninguna instancia de estas clases, asignándolas erróneamente a categorías como RESPIRAR, INFECCIÓN O FIEBRE. En la Figura 35 se pueden observar unos ejemplos.

Otras glosas como TEMBLOR y ELEVADO presentan un F1-score de apenas 0.125, con una sensibilidad de 0.067, lo cual también evidencia una alta confusión con otras clases similares. En el extremo inferior de la Tabla 12 —aunque no representan los mejores resultados globales— se encuentran clases como DIABETES (ver Figura 36), DIARREA, CORTAR_ABRIR y CARRO, con F1-scores por encima de 0.90, lo que refleja un mejor desempeño relativo dentro del conjunto de glosas con bajo rendimiento.

(a) Imagen comparativa entre la glosa BUENO y la glosa INFECCIÓN





(b) Imagen comparativa entre la glosa CUERPO y la glosa CUIDAR

Figura 35. Comparativas de glosas predichas contra glosas actuales.

5.5. Resultados en la modalidad continua

Una vez obtenido un modelo capaz de reconocer de manera efectiva 111 glosas (de un total de 121) en modalidad aislada, en esta sección se presentan los resultados relacionados con la definición de los parámetros del modelo para su aplicación en modalidad continua, así como los resultados del proceso de traducción, utilizando el conjunto de datos compuesto por 96 frases, que en conjunto abarcan 119 glosas.

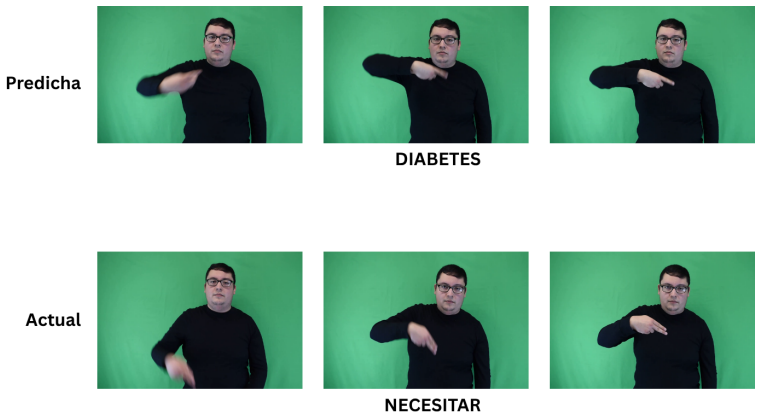


Figura 36. Imagen comparativa entre la glosa DIABETES y la glosa NECESITAR

Tabla 12. Clases con menor desempeño en clasificación (ordenadas por F1-Score). Note que son las clases cuyo valor de F1-score es menor a 1.

Glosa	F1-Score	Sensibilidad	Confusión
OPRESION_EN_PECHO	0.000	0.000	RESPIRAR
BUENO	0.000	0.000	INFECCIÓN
CUERPO	0.000	0.000	CUIDAR
TEMPERATURA	0.000	0.000	FIEBRE
TEMBLOR	0.125	0.067	INFECCIÓN
ELEVADO	0.125	0.067	ARTICULACIONES
TARDE	0.286	0.182	ENFERMO
INFECCION	0.391	1.000	—
NO_NADA	0.500	0.333	NADA
FIEBRE	0.571	0.800	TEMPERATURA
RESPIRAR	0.652	1.000	—
CUIDAR	0.667	1.000	—
NADA	0.750	1.000	—
ENFERMO	0.769	1.000	—
ARTICULACIONES	0.889	1.000	—
DAÑAR	0.900	1.000	—
NECESITAR	0.909	0.833	DIABETES
CARRO	0.923	1.000	—
CORTAR_ABRIR	0.938	1.000	—
DIABETES	0.938	1.000	—
DIARREA	0.938	1.000	—
DIA	0.966	0.933	TARDE
CARTERA	0.968	1.000	—
BOMBEROS	0.968	1.000	—
EMBARAZADA	0.968	1.000	—

5.5.1. Búsqueda de parámetros de desplazamiento

Con el objetivo de definir un tamaño de ventana adecuado para el reconocimiento continuo, se realizó una estimación experimental de la duración promedio de las señas activas dentro del conjunto de datos de glosas aisladas. Dado que este conjunto no tiene una tasa de refresco estandarizada, y que existen videos grabados a diferentes frecuencias (25, 30 y 60 *fps*), se optó por calcular primero la duración en segundos y luego convertirla a fotogramas según la tasa correspondiente.

Tabla 13. Duración de glosas seleccionadas por sujeto

Glosa	**Sujeto**	**Duración (s)**
"SU"	1	0.85
"PRESION_ARTERIAL"	7	1.33
"QUIMICOS ⋆"	1	1.63
"ROBAR ⋆"	3	1.30
"SENTIDO_GUSTO"	2	1.36
"NO_NADA"	4	1.20
"PULMONES ⋆"	5	1.40
"VOMITO"	1	0.81
"TEMBLOR ⋆"	6	1.72
"ROSTRO_HINCHADO ⋆"	8	0.85
Duración promedio	N/A	1.245±0.30

Para ello, se seleccionaron diez muestras representativas que cubren tanto señas unimanuales como bimanuales (estas últimas marcadas con ⋆ en la Tabla 13). La duración promedio obtenida fue de 1.245 ± 0.30 segundos, lo que permite estimar la longitud en fotogramas esperada bajo diferentes frecuencias:

- A 25 *fps*: aproximadamente 32 fotogramas.
- A 30 *fps*: aproximadamente 37 fotogramas.
- A 60 *fps*: aproximadamente 75 fotogramas.

Con base en estas estimaciones y con el objetivo de cubrir la variabilidad natural en la duración de las señas, se definieron los siguientes tamaños de ventana para ser evaluados en la modalidad continua: **22, 26, 32, 37, 51 y 75 frames**.

Los resultados de la búsqueda para identificar la mejor configuración de parámetros de desplazamiento fueron realizadas considerando como métrica objetivo la métrica de aparición (*appearance_metric*, Ecuación 19) y evaluándolo desde un enfoque top-k más probables. Específicamente, para cada ventana evaluada, el modelo produce una distribución de probabilidad sobre todas las clases posibles, y top-k indica cuántas de las clases con mayor probabilidad se retienen para su análisis. En este caso tenemos $k = 3$ más probables donde:

- **Top-1**: Solo considera la clase con la mayor probabilidad. Si esa predicción coincide con una glosa del conjunto objetivo, se considera correctamente reconocida.
- **Top-2**: Se consideran las dos clases más probables. Si cualquiera de ellas coincide con una glosa objetivo, se cuenta como aparición correcta.
- **Top-3**: Se permite un margen más amplio, considerando las tres clases con mayor probabilidad.

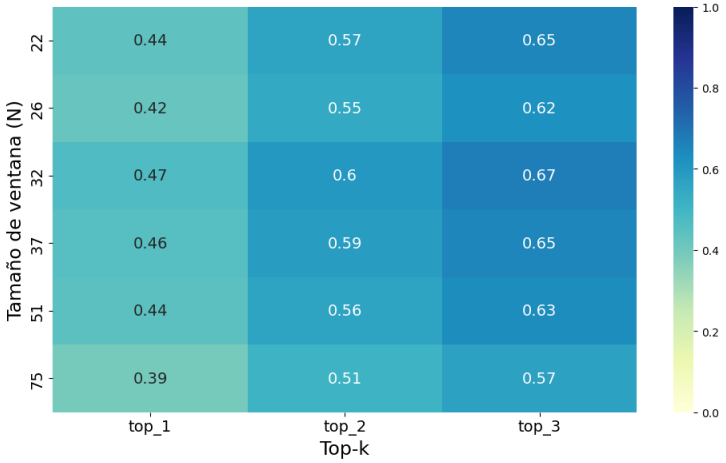
El uso de diferentes valores de top-k permite analizar cómo se modifica el desempeño del sistema al ajustar el criterio de decisión. A mayor k, mayor es la tolerancia del sistema a errores en el orden de probabilidad, lo que generalmente incrementa la métrica de aparición, aunque también implica una menor precisión en la predicción más confiable.

Como se muestra en la Figura 37(a), al analizar el comportamiento de la métrica de aparición respecto al tamaño de ventana (*window_size*), se observa que la mayor aparición se alcanza con una ventana de tamaño **32**, alcanzando un valor promedio de **0.67** bajo el esquema **top-3**. Este valor representa la mejor recuperación de glosas únicas objetivo en todas las configuraciones evaluadas. También se observa que tamaños intermedios, como 32 y 37, tienden a ofrecer un equilibrio entre ventanas muy cortas (como 22 o 26), que podrían omitir parte de la seña, y ventanas demasiado largas (como 75), que diluyen el movimiento relevante con ruido temporal.

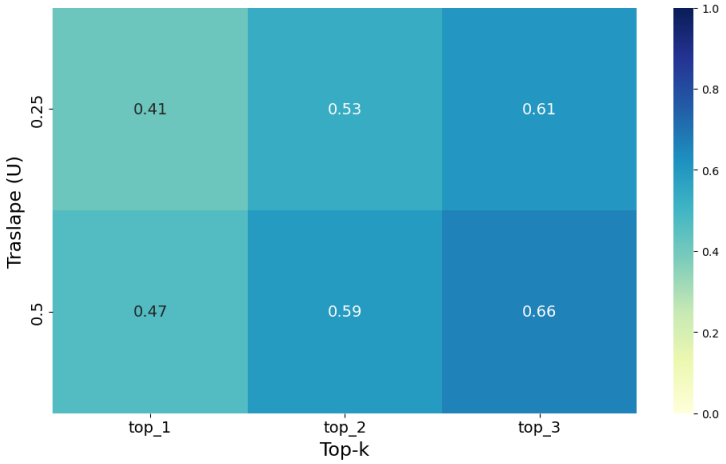
Por otro lado, la Figura 37(b) muestra los resultados promedio según el porcentaje de traslape (U). Se evaluaron dos valores: 0.25 y 0.5. En todos los casos, el mayor traslape (**0.5**) obtuvo mejores resultados, alcanzando un máximo de **0.66** en el esquema top-3. Este comportamiento sugiere que un mayor porcentaje de traslape permite una mejor cobertura temporal de las señas dentro del video, lo cual favorece la detección de glosas relevantes al evitar cortes abruptos en los segmentos analizados.

En conjunto, los resultados de ambas figuras indican que la combinación óptima de parámetros se alcanza utilizando un **tamaño de ventana de 32 fotogramas** y un **porcentaje de traslape del 50 %** (ver

Tabla 14), evaluado bajo el esquema de **top-3**. Esta configuración maximiza la métrica de aparición, permitiendo una recuperación más completa de glosas objetivo dentro del reconocimiento continuo.



(a) Resultados promedio por tamaño de ventana



(b) Resultados promedio por porcentaje de traslape

Figura 37. Mapas de calor que reflejan los resultados de la búsqueda de parámetros de desplazamiento.

5.5.2. Búsqueda de parámetros de postprocesamiento

Con el objetivo de mejorar la coherencia y precisión de las secuencias de glosas generadas en la modalidad continua, se exploraron distintas configuraciones del proceso de postprocesamiento descrito previamente. Específicamente, se evaluaron variaciones en dos parámetros clave: el umbral de tolerancia aplicado al filtro de confianza y la estrategia de consolidación de glosas repetidas (promediado de distribuciones o selección de la más confiable).

Tabla 14. Resultados promedio de aparición Top-k para diferentes tamaños de ventana y traslapes

Ventana	Traslape	Top-1	Top-2	Top-3
32	0.50	0.508 ± 0.342	0.633 ± 0.333	0.696 ± 0.322
37	0.50	0.489 ± 0.348	0.621 ± 0.341	0.685 ± 0.331
51	0.50	0.472 ± 0.345	0.600 ± 0.342	0.668 ± 0.329
22	0.50	0.459 ± 0.336	0.585 ± 0.335	0.674 ± 0.328
26	0.50	0.440 ± 0.340	0.562 ± 0.340	0.638 ± 0.332
32	0.25	0.436 ± 0.339	0.569 ± 0.339	0.645 ± 0.330
37	0.25	0.425 ± 0.339	0.549 ± 0.343	0.619 ± 0.333
75	0.50	0.424 ± 0.336	0.551 ± 0.340	0.614 ± 0.333
22	0.25	0.422 ± 0.336	0.546 ± 0.342	0.626 ± 0.334
51	0.25	0.412 ± 0.343	0.526 ± 0.341	0.601 ± 0.340
26	0.25	0.406 ± 0.331	0.528 ± 0.343	0.609 ± 0.337
75	0.25	0.361 ± 0.328	0.470 ± 0.340	0.531 ± 0.342

Tabla 15. Resultados comparativos según tolerancia y método de promediado

Tolerancia	Promediar	POPE	AELVT	LDN	WER
baseline	baseline	0.557 ± 0.232	0.998 ± 0.030	0.959 ± 0.075	3.455 ± 1.888
0.25	False	0.489 ± 0.265	0.849 ± 0.243	0.953 ± 0.088	3.286 ± 1.751
0.25	True	0.486 ± 0.264	0.848 ± 0.243	0.952 ± 0.089	3.276 ± 1.763
0.5	False	0.452 ± 0.278	0.790 ± 0.286	0.940 ± 0.117	2.590 ± 1.574
0.5	True	0.436 ± 0.270	0.790 ± 0.288	0.938 ± 0.123	2.485 ± 1.557
0.75	False	0.395 ± 0.264	0.780 ± 0.308	0.928 ± 0.148	1.902 ± 1.273
0.75	True	0.370 ± 0.242	0.813 ± 0.297	0.928 ± 0.147	1.723 ± 1.129

Los resultados se presentan en la Tabla 15, donde se reportan las métricas *POPE* y *AELVT* para cada configuración evaluada. El modelo sin posprocesamiento, considerado como el *baseline*, presenta un valor de *POPE* de $\mathbf{0.557 \pm 0.232}$ y un *AELVT* de $\mathbf{0.998 \pm 0.030}$, lo cual refleja un alto grado de error en la presencia y el orden de las glosas, así como una escasa capacidad del sistema para converger a frases coherentes. Al aplicar un umbral de tolerancia de 0.75 junto con la estrategia de promediado, se obtiene el menor valor de *POPE* ($\mathbf{0.370 \pm 0.242}$), lo que indica una mejora significativa en la precisión estructural de la secuencia. No obstante, esta configuración también presenta un ligero aumento en la métrica *AELVT* ($\mathbf{0.813 \pm 0.297}$), lo que sugiere una leve disminución en la capacidad de convergencia hacia frases lingüísticamente correctas. Estos resultados evidencian la importancia del posprocesamiento para optimizar el equilibrio entre precisión estructural y calidad semántica en el reconocimiento continuo.

La Figura 38 muestra la distribución del error *POPE* por sujeto. Se aprecia una tendencia general estable entre la mayoría de los participantes, con una mediana alrededor de 0.4 a 0.5. Sin embargo, el sujeto S5 presenta una mayor dispersión y valores extremos, lo que indica un comportamiento más inconsistente en la predicción de glosas en su caso. Esto podría estar relacionado con factores individuales, como la velocidad de ejecución o la variabilidad en las señas.

Por otro lado, la Figura 39 muestra cómo varía el error *POPE* en función del número de glosas por secuencia. Se observa que las secuencias de una sola glosa presentan una dispersión considerablemente mayor y una mediana más alta, mientras que las secuencias con mayor número de glosas (cuatro o cinco) tienden a mostrar menor variabilidad y menor error promedio. Esto sugiere que el sistema logra un mejor desempeño cuando la señal contiene mayor contexto temporal y semántico.

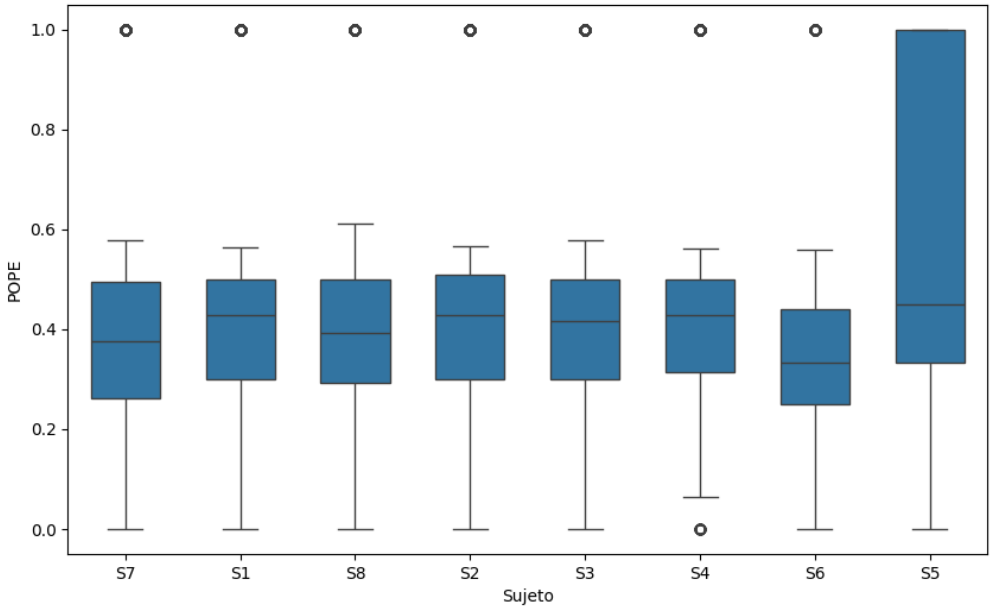


Figura 38. Distribución de errores de presencia y orden (*POPE*) por sujeto en la modalidad continua

Los valores obtenidos para las métricas Word Error Rate (WER) y Distancia de Levenshtein Normalizada (LDN) permiten estimar la fidelidad con la que los LLMs convierten secuencias de glosas en frases gramaticalmente correctas en español. Los resultados evidencian una tendencia clara: a medida que se incrementa la tolerancia en el posprocesamiento, ambas métricas disminuyen, reflejando una mayor coincidencia con las frases de referencia. El mejor desempeño se obtuvo con una tolerancia de 0.75 y utilizando el promediado de distribuciones, registrando un WER de 1.723 ± 1.129 y una LDN de 0.928 ± 0.147. En conjunto, estos resultados apoyan la selección de un umbral de tolerancia moderado (0.75) y el uso del promediado como estrategia de agrupamiento, lo cual contribuye a reducir el error sin comprometer de forma significativa la fidelidad general de las secuencias.

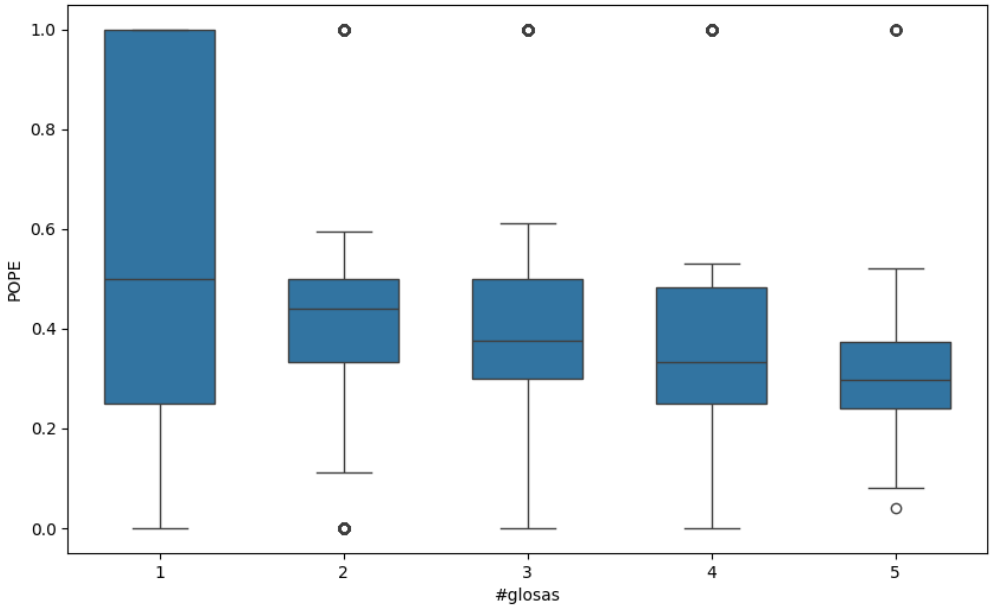


Figura 39. Distribución de errores de presencia y orden (*POPE*) por número de glosas en la modalidad continua

5.6. Análisis a detalle en la generación de frases con LLMs.

En esta sección veremos con mayor detenimiento las traducciones realizadas por los LLMs, realizando una comparación directa entre los resultados obtenidos por el postprocesamiento con mejor desempeño y los resultados ideales, donde las glosas objetivo de cada oración fueron utilizadas para generar una oración; estos datos representan el basal.

5.6.1. Modelos de LLM

Como se explicó en la Sección 4.8.5, para la generación de frases a partir de las glosas reconocidas se emplearon los modelos de lenguaje *Chat-GPT 3.5-turbo*, *DeepSeek* (utilizado como modelo por defecto) y *Gemini 2.0-flash*. En esta sección se presentan y comparan los resultados obtenidos por cada uno de ellos.

A partir de la Figuras 40 y 41, se comparan los resultados de los modelos de lenguaje *Chat-GPT*, *Gemini* y *DeepSeek* en la tarea de generación de frases a partir de glosas reconocidas, evaluando el desempeño

mediante las métricas WER y LDN. En ambos gráficos, se distingue entre la salida generada de forma directa (basal) y la obtenida tras aplicar un proceso de postprocesamiento —implementando la mejor estrategia definida anteriormente—.

Los resultados obtenidos en términos de WER (Figura 40) muestran un desempeño relativamente similar entre los tres modelos evaluados, destacando una ligera ventaja de *DeepSeek* cuando se aplica postprocesamiento. En la Figura 41, se observa una elevada concentración de valores en torno a LDN $\approx$ 1.0, especialmente en los datos basales, aunque con medianas ligeramente menores (alrededor de 0.8). Esto sugiere que la calidad de las traducciones generadas por los LLMs aún dista de las referencias proporcionadas por expertos en LSM, donde en ciertos casos se aproximan a las correctas.

Cabe considerar que la métrica LDN puede ser especialmente estricta ante errores como la omisión de tildes o la sustitución de palabras clave, lo que podría explicar en parte los altos valores observados. Finalmente, la variabilidad en los resultados postprocesados, particularmente el menor rendimiento de *ChatGPT*, apunta a la necesidad de un análisis cualitativo que permita comprender en qué contextos cada modelo falla o acierta.

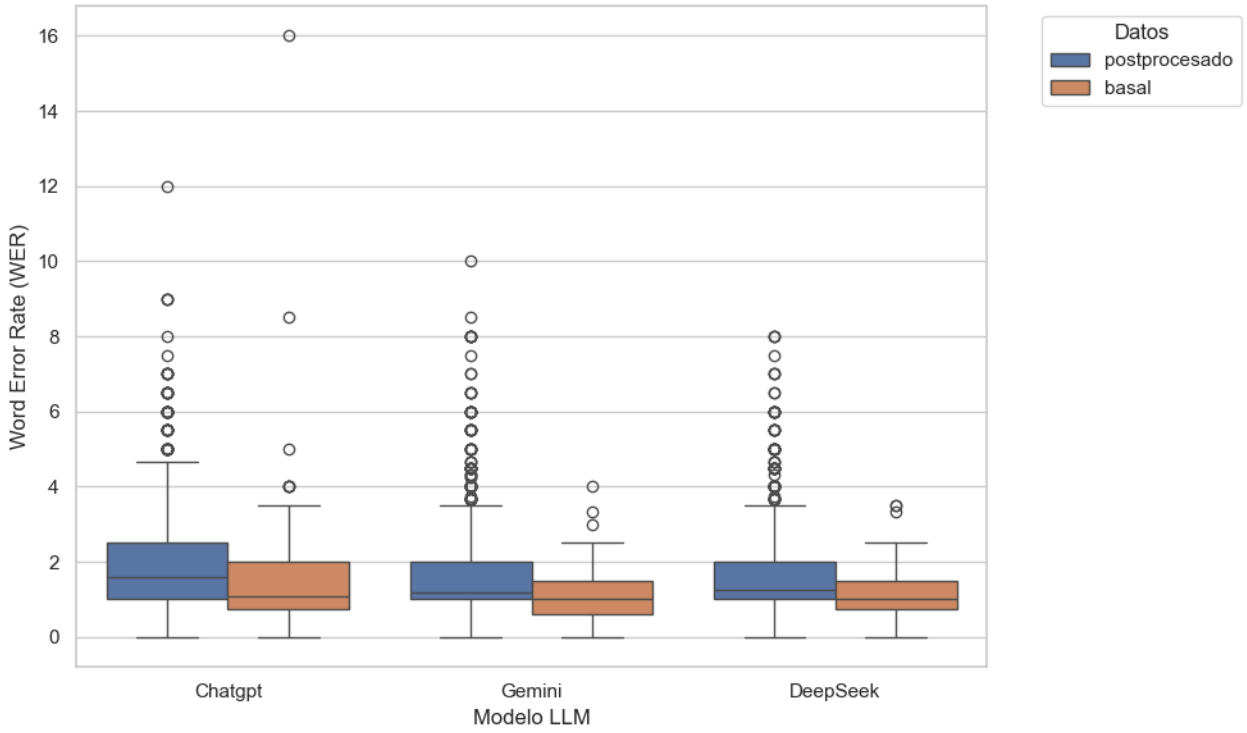


Figura 40. Resultado de la generación de frases por LLM medida con WER.

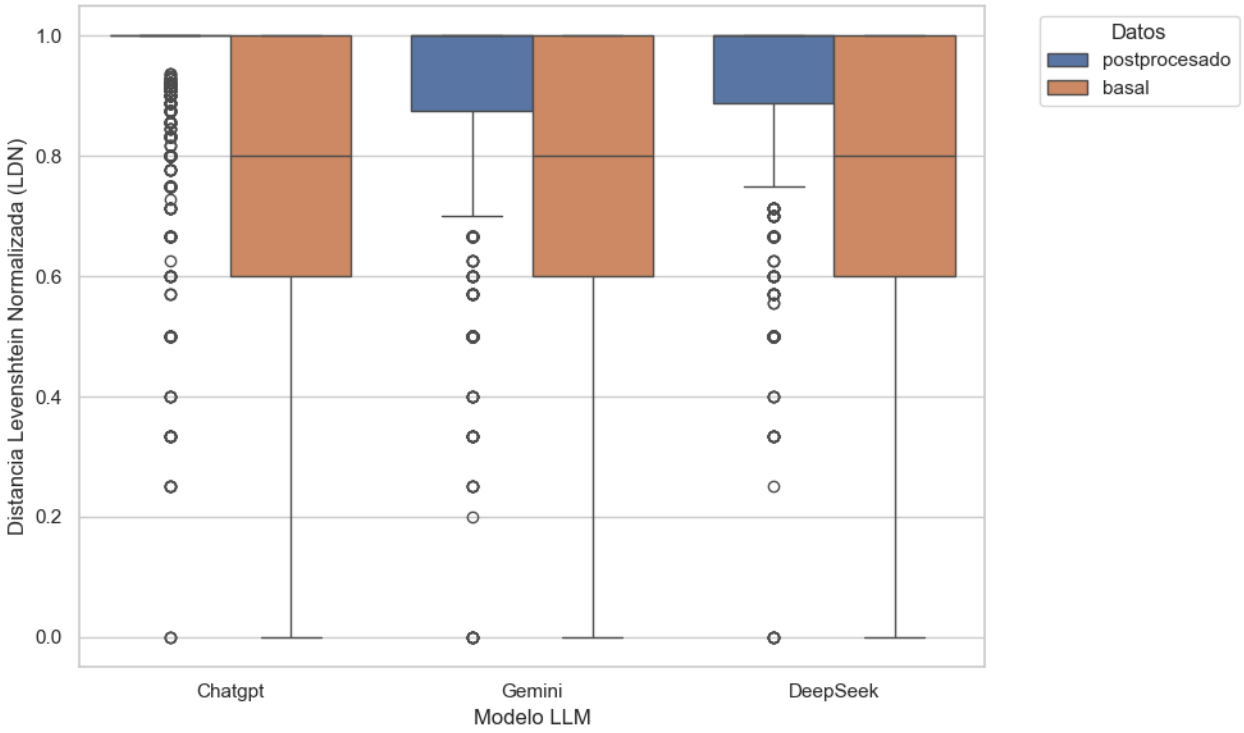


Figura 41. Resultado de la generación de frases por LLM medida con LDN.

5.6.2. Análisis por casos

En esta sección se analizan los casos con mayor y menor desempeño en la traducción, considerando frases compuestas por una, dos y tres glosas. La selección de ejemplos se realizó con base en la métrica POPE, mientras que la calidad de las traducciones generadas por los modelos de lenguaje fue evaluada mediante la métrica WER.

5.6.2.1. Traducción de una glosa

Los resultados de traducción para frases compuestas por una sola glosa (ver Tabla 16) muestran que los modelos LLM tienden a generar expresiones más completas o contextualizadas, lo cual eleva los valores del WER pese a la aparente corrección semántica. Por ejemplo, ante la glosa NO, ChatGPT generó la frase "¡No quiero!", mientras que Gemini y DeepSeek se limitaron a la forma "No.", obteniendo menores

valores de WER (1.0 frente a 2.0). Para otras glosas como POR_FAVOR, los tres modelos lograron coincidir con la frase objetivo, alcanzando un WER de 0.5, lo que representa un resultado favorable. Sin embargo, en casos de glosas mal clasificadas, como ESTAR frente al objetivo GRACIAS, o pares como SU y ESTO interpretados erróneamente como afirmaciones, se evidencian errores semánticos importantes, con tasas de error por palabras de hasta 4.0. En general, DeepSeek mostró mejor desempeño en los casos más ambiguos, con WER ligeramente inferior al de sus contrapartes, aunque ninguna de las generaciones fue completamente libre de errores en todos los ejemplos. Estos resultados evidencian la dificultad de generar traducciones precisas a partir de glosas aisladas, donde el contexto sintáctico y semántico suele ser limitado.

Tabla 16. Resultados representativos de traducción para una glosa. LLM: `CHG` = ChatGPT, `GEM` = Gemini, `DSK` = DeepSeek. **Glosa**: resultado del modelo en modalidad continua. **Glosa Objetivo**: glosa de referencia. **Frase Generada**: oración generada por el LLM. **Frase Objetivo**: traducción de referencia realizada por expertos.

Glosa	Glosa Objetivo	LLM	Frase Generada	Frase Objetivo	POPE	WER
['NO']	['NO']	CHG	¡No quiero!	No	0.0	2.0
['NO']	['NO']	DSK	No.	No	0.0	1.0
['AYUDA']	['AYUDA']	GEM	Necesito ayuda.	¡Auxilio!	0.0	2.0
['SI']	['SI']	CHG	Si quieres, podemos comer juntos mañana.	Si	0.0	5.0
['SI']	['SI']	DSK	Sí.	Si	0.0	1.0
['POR_FAVOR']	['POR_FAVOR']	GEM	Por favor.	Por favor	0.0	0.5
['ESTAR']	['GRACIAS']	CHG	¿Dónde está el baño?	Gracias	1.0	4.0
['ESTAR']	['GRACIAS']	DSK	Está.	Gracias	1.0	1.0
['NO_PODER' 'NO_NADA']	['NADA']	GEM	No puedo hacer nada.	De nada	1.0	2.0
['SU' 'ESTO']	['SI']	DSK	Eso es tuyo.	Si	1.0	3.0

5.6.2.2. Traducción de dos glosas

En el caso de frases compuestas por dos glosas, los resultados muestran una variabilidad considerable en el desempeño de los modelos. Algunos ejemplos evidencian una traducción precisa y completamente alineada con la frase objetivo, como el caso de la glosa `['FARMACIA', 'DONDE']`, traducida correctamente por ChatGPT con un WER de 0.00. Sin embargo, también se observan traducciones parcialmente correctas, donde la intención general se mantiene pero con diferencias léxicas o estructurales, como en `['OTRA_VEZ', 'POR_FAVOR']` (WER = 0.75) o `['BRAZO', 'BRAZO_HINCHADO']` (WER = 0.80).

Por otro lado, existen ejemplos con errores significativos en los que las frases generadas no coinciden

semánticamente con la frase objetivo, como ['SU', 'COMIDA'] frente a ['BONITO', 'DIA'] con WER = 2.00, lo que sugiere una interpretación completamente distinta. Asimismo, los casos con POPE = 1.0 reflejan un desajuste entre las glosas proporcionadas y las frases de referencia, lo cual se traduce en errores de traducción relevantes que elevan el WER por encima de 1.0. Estos resultados ponen de manifiesto que, aunque los modelos pueden generar frases coherentes desde el punto de vista gramatical, su fidelidad semántica depende fuertemente del contexto y de la correspondencia entre glosas y significado.

Tabla 17. Resultados representativos de traducción para dos glosas. LLM: CHG = ChatGPT, GEM = Gemini, DSK = DeepSeek. **Glosas**: resultado del modelo en modalidad continua. **Glosas Objetivo**: glosas de referencia. **Frase Generada**: oración generada por el LLM. **Frase Objetivo**: traducción de referencia realizada por expertos.

Glosas	Glosas Objetivo	LLM	Frase Generada	Frase Objetivo	POPE	WER
['FARMACIA' 'DONDE']	['FARMACIA', 'DONDE']	CHG	¿Dónde está la farmacia?	¿Dónde está la farmacia?	0.0	0.00
['OTRA_VEZ' 'POR_FAVOR']	['OTRA_VEZ', 'POR_FAVOR']	CHG	Otra vez, por favor.	Lo repites por favor	0.0	0.75
['BRAZO' 'BRAZO_HINCHADO']	['BRAZO', 'BRAZO_HINCHADO']	DSK	Tengo el brazo hinchado.	Tengo hinchazón en los brazos	0.0	0.80
['PULMONES' 'DOLOR']	['PULMONES', 'DOLOR']	GEM	Me duelen los pulmones.	Siento dolor pulmonar	0.0	1.33
['DOCTOR' 'LLAMAR']	['DOCTOR', 'LLAMAR']	GEM	Llama al doctor.	¡Llame a un doctor!	0.0	1.00
['TELEFONO' 'NO_ESCUCHAR']	['AMBULANCIA', 'LLAMAR']	DSK	No escucho el teléfono.	¡Llame a una ambulancia!	1.0	1.00
['ROSTRO_HINCHADO' 'FUEGO']	['ARTICULACIONES', 'DOLOR']	CHG	El rostro hinchado arde en fuego.	Siento dolor en las articulaciones	1.0	1.20
['NO_PODER' 'TELEFONO' 'NOCHE']	['BOMBEROS', 'LLAMAR']	CHG	Esta noche no podrás usar el teléfono.	¡Llame a los bomberos!	1.0	1.75
['SU' 'COMIDA']	['BONITO', 'DIA']	GEM	Su comida está lista.	Bonito día	1.0	2.00
['BUENO' 'TOS']	['CORAZON', 'PALPITACION']	DSK	Es bueno para la tos.	Tengo el pulso acelerado	1.0	1.25

5.6.2.3. Traducción de tres glosas

Los resultados obtenidos en la traducción de frases compuestas por tres glosas muestran un contraste significativo entre casos bien interpretados y otros con errores de traducción importantes. Por un lado, se identifican ejemplos de traducción exitosa con WER bajo o nulo, como en ['HOLA', 'COMO', 'ESTAR'] y ['MI', 'TELEFONO', 'ROBAR'], ambos correctamente interpretados por Gemini con valores de WER de 0.00 y 0.60, respectivamente. También se observan frases aceptables como ['MI',

'CABEZA', 'GOLPE'] con WER = 0.50 y ['YO', 'INFECCION', 'ESTOMAGO'] con WER = 0.75, lo que indica una buena correspondencia semántica con la frase objetivo.

Por otro lado, los ejemplos con POPE = 1.0 reflejan una discordancia total entre la intención de la glosa y la frase objetivo, lo que se traduce en valores altos de WER. Particularmente, el caso ['NO_ESCUCHAR', 'NOCHE', 'GUSTAR', 'VOZ'] genera frases que, aunque coherentes en estructura gramatical, se alejan completamente del significado esperado, alcanzando un WER de hasta 4.00. Asimismo, en ['ESTO', 'ACCIDENTE', 'COMPRAR', 'DIABETES'], los tres modelos muestran interpretaciones erróneas con WER que oscilan entre 1.33 y 3.67. Estos hallazgos evidencian que, a medida que aumenta la complejidad de la entrada (más glosas), también incrementa la dificultad para mantener la fidelidad semántica, especialmente cuando la correspondencia con la frase objetivo no es directa o depende de un contexto específico.

Tabla 18. Resultados representativos de traducción para tres glosas. LLM: CHG = ChatGPT, GEM = Gemini, DSK = DeepSeek. **Glosas**: resultado del modelo en modalidad continua. **Glosas Objetivo**: glosas de referencia. **Frase Generada**: oración generada por el LLM. **Frase Objetivo**: traducción de referencia realizada por expertos.

Glosas	Glosas Objetivo	LLM	Frase Generada	Frase Objetivo	POPE	WER
['HOLA' 'COMO' 'ESTAR']	['HOLA', 'COMO', 'ESTAR']	GEM	Hola, ¿cómo estás?	Hola, ¿cómo estás?	0.0	0.00
['MI' 'TELEFONO' 'ROBAR']	['MI', 'TELEFONO', 'ROBAR']	GEM	Me robaron mi teléfono.	Me han robado mi teléfono	0.0	0.60
['MI' 'CABEZA' 'GOLPE']	['MI', 'CABEZA', 'GOLPE']	CHG	Me golpeé la cabeza.	Me golpee la cabeza	0.0	0.50
['YO' 'INFECCION' 'ESTOMAGO']	['YO', 'INFECCION', 'ESTOMAGO']	GEM	Yo tengo una infección estomacal.	Tengo una infección estomacal	0.0	0.75
['SU' 'DIABETES' 'NO_ESCUCHAR' 'NO_ENTENDER']	['VOZ', 'NO_ESCUCHO', 'NO_ENTIENDO']	CHG	Su diabetes no se escucha, no se entiende.	No entiendo el español	1.0	2.00
['NO_ESCUCHAR' 'NOCHE' 'GUSTAR' 'VOZ']	['MUCHO', 'GUSTO', 'CONOCER']	GEM	No me gusta escuchar voces por la noche.	¡Mucho gusto!	1.0	4.00
['TOS' 'DINERO' 'NO_PODER']	['YO', 'SENTIR', 'MAL']	DSK	Tú no puedes gastar dinero.	Me siento mal	1.0	1.67
['ESTO' 'ACCIDENTE' 'COMPRAR' 'DIABETES']	['EL', 'DAÑAR', 'YO']	CHG	Esto accidente comprar diabetes.	Me han atacado	1.0	1.33
['ESTO' 'ACCIDENTE' 'COMPRAR' 'DIABETES']	['EL', 'DAÑAR', 'YO']	GEM	Este accidente fue causado por comprar alimentos con diabetes.	Me han atacado	1.0	3.67
['ESTO' 'ACCIDENTE' 'COMPRAR' 'DIABETES']	['EL', 'DAÑAR', 'YO']	DSK	Este accidente compró diabetes.	Me han atacado	1.0	1.33

Este capítulo presentó una evaluación integral del sistema de reconocimiento y traducción de señas propuesto, abarcando desde la caracterización de los conjuntos de datos utilizados hasta la implementación y optimización del modelo. Se emplearon dos corpus existentes para las modalidades aislada y continua, los cuales difieren principalmente en la participación de sujetos no coincidentes y en los niveles de experiencia de los mismos, lo que introduce una alta variabilidad gestual. Esta variabilidad, aunque representa un reto para el modelo, también enriquece el entrenamiento al fomentar una mayor capacidad de generalización. El análisis de pérdida evidenció que *MediaPipe* es un sistema confiable para la captura de movimiento, y se observó que la experiencia del usuario influye en la coherencia de los gestos, lo que repercute directamente en la estabilidad de los datos aumentados. La arquitectura *ResNet* fue seleccionada por su desempeño estable en todos los experimentos, y tras su proceso de optimización, fue utilizada como base en la modalidad continua, alcanzando un error promedio del 37 % con una configuración óptima de 32 fotogramas y 50 % de traslape.

Respecto a la traducción automática, los resultados revelan que los LLMs logran generar frases gramaticalmente correctas, pero enfrentan dificultades para mantener la fidelidad semántica, especialmente al aumentar la cantidad de glosas por frase. Con una sola glosa, las traducciones tienden a ser comprensibles, aunque no siempre precisas; mientras que con entradas más complejas, la relación entre glosa y frase objetivo se diluye, afectando la coherencia del resultado. Estos hallazgos permiten concluir que, si bien los LLMs tienen un papel prometedor, su implementación efectiva requiere considerar su sensibilidad al contexto y su limitada capacidad para resolver ambigüedades semánticas.

Capítulo 6. Discusión y Conclusiones

Dentro de este trabajo se propuso un sistema basado en visión por computadora, aprendizaje automático y LLMs para realizar la traducción de oraciones en español a partir de glosas reconocidas de la LSM, en contextos médicos, de emergencia y frases cotidianas. Para ello, se exploraron las capacidades de distintas arquitecturas de aprendizaje profundo para el reconocimiento de señas dinámicas, centradas en ideogramas que forman parte de la Lengua de Señas Mexicana (LSM). Se utilizaron dos conjuntos de datos de señas basados en un corpus de frases enfocadas en contextos médicos, situaciones de emergencia y expresiones cotidianas. Uno de los conjuntos estuvo compuesto por 121 señas dinámicas aisladas, y el otro por 96 frases cuyas glosas varían entre una y cinco. Los sujetos de ambos conjuntos fueron diversos y no se repitieron entre ellos.

Se evaluaron dos modelos de detección de posturas. El primero fue *OpenPose*, un modelo orientado a la detección de posturas en múltiples personas; el segundo fue *Holistic*, del paquete *MediaPipe*, un *framework* desarrollado por Google para tareas de aprendizaje automático. Ambos modelos fueron sometidos a un análisis de pérdida de datos, donde *MediaPipe* demostró ser más robusto ante la oclusión de puntos del cuerpo y el movimiento de las articulaciones.

Asimismo, se aplicó la técnica de aumento de datos *Frame Skip Sampling* para incrementar la cantidad de muestras y generar una entrada reducida y consistente en cuanto a la secuencialidad de las señas. Posteriormente, se estudió la similitud entre estas muestras, observándose que la experiencia de la persona señante podría influir en la calidad y la similitud de los datos. En particular, los datos provenientes de personas expertas mostraron mayor consistencia. Se realizó un análisis exhaustivo de los casos con mayor variabilidad, concluyendo que ciertas configuraciones complejas realizadas por participantes no expertos generan ruido en la detección con *MediaPipe* y presentan mayores índices de variabilidad.

Con base en este conocimiento, se diseñaron tres experimentos para evaluar distintas arquitecturas de aprendizaje profundo, específicamente modelos basados en RNN y CNN en modalidad aislada. Cada experimento se orientó a identificar la robustez de los clasificadores frente a la experiencia de los participantes, previamente observada en los datos. Se evidenció que dicha experiencia impacta directamente en el rendimiento de los clasificadores. La arquitectura *ResNet* se posicionó como la más robusta en todos los experimentos y, tras ser sometida a una optimización de hiperparámetros en su capa de clasificación, alcanzó un *F1-score* de **0.92**.

Este modelo fue posteriormente adaptado al dominio continuo, lo que requirió ajustar su entrada a 15 fotogramas y aplicar un sistema de desplazamiento basado en ventanas con traslape. Se determinó

que la mejor configuración correspondía a una ventana de 32 fotogramas y un traslape del 50 %. No obstante, en el reconocimiento continuo no existen métricas que midan explícitamente aspectos como el orden y la presencia de glosas dentro de una secuencia reconocida. Por tal motivo, en este trabajo se diseñó la métrica *Presence and Order Penalty Error* (POPE), la cual considera ambos aspectos. Esta métrica fue implementada y utilizada como función objetivo para identificar la mejor estrategia de posprocesamiento de las secuencias de glosas generadas por el sistema continuo. Se observó que la mejor estrategia consistió en aplicar un umbral de decisión alto (0.75 de probabilidad) y promediar las distribuciones de las predicciones repetidas, lo cual permitió reducir el error promedio de 0.55 a 0.37.

Finalmente, se evaluó la capacidad de los LLMs para traducir secuencias de glosas reconocidas por el sistema continuo. Los resultados evidenciaron que, si bien estos modelos pueden generar frases gramaticalmente coherentes, su fidelidad semántica depende en gran medida del número de glosas, la claridad del contexto y la relación directa entre glosa y significado. A pesar de estas limitaciones, estrategias de posprocesamiento permitieron reducir los errores en casos particulares, indicando que una integración adecuada entre sistemas de reconocimiento y LLMs puede facilitar la generación automática de traducciones funcionales, aunque aún se requiere investigación adicional para alcanzar traducciones con precisión cercana a la humana.

6.1. Limitaciones

Este trabajo presenta diversas limitaciones que deben ser consideradas. En primer lugar, la dependencia de conjuntos de datos previamente recolectados impidió controlar parámetros críticos como la tasa de refresco (fps) y la consistencia en la ejecución de las glosas, lo que derivó en secuencias de longitud variable. La alta variabilidad entre sujetos también representa un desafío, ya que puede inducir al modelo a sobreajustarse a datos menos diversos y afectar su capacidad de generalización.

En cuanto al reconocimiento, la precisión del sistema está fuertemente condicionada por el desempeño del *framework MediaPipe Holistic*, del cual depende la calidad de las características extraídas. La presencia de errores en la detección de poses o articulaciones puede degradar notablemente el rendimiento, sobre todo en condiciones de oclusión o movimientos rápidos. Por su parte, la estrategia de posprocesamiento basada únicamente en niveles de confianza también introduce riesgos, al aceptar predicciones erróneas con alta probabilidad o descartar aciertos con baja certeza.

Respecto a la traducción, los LLMs muestran limitaciones semánticas importantes: aunque generan frases gramaticalmente correctas, su precisión se deteriora cuando la secuencia de glosas carece de contexto claro o cuando la relación con la frase objetivo es indirecta, especialmente en entradas de mayor longitud. Finalmente, el sistema enfrenta una barrera de escalabilidad, ya que el aumento progresivo de señas en el corpus implicará un incremento sustancial en la complejidad del reconocimiento, lo cual demanda estrategias más robustas y adaptativas a futuro.

6.2. Trabajo a futuro

Los resultados obtenidos en este trabajo fueron satisfactorios, y es pertinente dar seguimiento a diversas áreas de oportunidad identificadas a lo largo del estudio. A continuación, se enumeran algunas líneas de trabajo a futuro:

- Profundizar en el estudio de la relación entre el nivel de experiencia de los sujetos (expertos y no expertos en LSM) y la calidad de los datos extraídos.
- Ampliar y estructurar de mejor manera el corpus, incorporando una mayor diversidad de contextos y dominios temáticos.
- Explorar estrategias de clasificación jerárquica basadas en la descomposición de los ideogramas en sus unidades fonéticas fundamentales (quierema, toponema y kinema), como una alternativa escalable a largo plazo.
- Mejorar la calidad de las frases generadas por los LLMs mediante técnicas como *fine-tuning* o *prompt-tuning*, con el objetivo de incorporar de forma más precisa la gramática propia de la LSM.
- Integrar otros enfoques, como el procesamiento del lenguaje natural (NLP) y el uso de arquitecturas avanzadas como *Transformers* que combine información multimodal.
- Evaluación de técnicas de segmentación automática de señas continuas, con énfasis en la detección de límites de glosa en flujos de video reales.

Literatura citada

Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G. S., Davis, A., Dean, J., Devin, M., Ghemawat, S., Goodfellow, I., Harp, A., Irving, G., Isard, M., Jia, Y., Jozefowicz, R., Kaiser, L., ... , & Zheng, X. (2015). Tensorflow: Large-scale machine learning on heterogeneous systems. *Mountain View, CA: Tensorflow*. https://www.tensorflow.org/.

Abdullahi, S. B., Chamnongthai, K., Bolon-Canedo, V., & Cancela, B. (2024). Spatial–temporal feature-based end-to-end fourier network for 3d sign language recognition. *Expert Systems with Applications*, *248*, 123258. https://doi.org/https://doi.org/10.1016/j.eswa.2024.123258.

Alejandro, S.-M. & Antonio, N.-C. J. (2021). A real-time deep learning system for the translation of mexican signal language into text. In *2021 Mexican International Conference on Computer Science (ENC)*, 1–7. https://doi.org/10.1109/ENC53357.2021.9534825.

Alsharif, B., Altaher, A. S., Altaher, A., Ilyas, M., & Alalwany, E. (2023). Deep learning technology to recognize american sign language alphabet. *Sensors*, *23*(18), 7970. https://doi.org/10.3390/s23187970.

Bazarevsky, V., Grishchenko, I., Raveendran, K., Zhu, T., Zhang, F., & Grundmann, M. (2020). Blazepose: On-device real-time body pose tracking. *arXiv preprint arXiv:2006.10204*. https://doi.org/10.48550/arXiv.2006.10204.

Bazarevsky, V., Kartynnik, Y., Vakunov, A., Raveendran, K., & Grundmann, M. (2019). Blazeface: Sub-millisecond neural face detection on mobile gpus. *arXiv preprint arXiv:1907.05047v2*. https://doi.org/10.48550/arXiv.1907.05047.

Belcic, I. (2024). What is classification in machine learning? *IBM*. https://www.ibm.com/think/topics/classification-machine-learning.

Bishop, C. M. (2006). *Pattern Recognition and Machine Learning*. Information Science and Statistics. Springer, New York.

Borges-Galindo, E. A., Morales-Ramírez, N., González-Lee, M., García-Martínez, J. R., Nakano-Miyatake, M., & Perez-Meana, H. (2024). Sign language interpreting system using recursive neural networks. *Applied Sciences*, *14*(18). https://doi.org/10.3390/app14188560.

Cao, Z., Hidalgo, G., Simon, T., Wei, S.-E., & Sheikh, Y. (2021). Openpose: Realtime multi-person 2d pose estimation using part affinity fields. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *43*(1), 172–186. https://doi.org/10.1109/TPAMI.2019.2929257.

Cervantes, J., García-Lamont, F., Rodríguez-Mazahua, L., & López-Chau, A. (2016). Recognition of mexican sign language from frames in video sequences. *Intelligent Computing Theories and Application - Springer Verlag*, *9772*, 353–362. https://doi.org/10.1007/978-3-319-42294-7_31.

Cruz Aldrete, M. (2008). *Gramática de la Lengua de Señas Mexicana*. [Tesis de Doctorado en Lingüística, Centro de Estudios Lingüísticos y Literarios, El Colegio de México]. Repositorio colmex. https://hdl.handle.net/20.500.11986/COLMEX/10001268.

Cruz Aldrete, M. (2018). Una aproximación al estudio de la adquisición de la lengua de señas mexicana. *Inventio. La génesis de la cultura universitaria en Morelos*. https://riaa.uaem.mx/handle/20.500.12055/198.

Cruz Aldrete, M. & Serrano, J. (2018). La comunidad sorda mexicana. vivir entre varias lenguas: Lsm, asl, lsm, español, inglés, maya. *Convergencias. Revista de educación*, *1*(2). https://revistas.uncu.edu.ar/ojs3/index.php/convergencias/article/view/1386.

Escobedo, C. (2017). *Diccionario de lengua de señas mexicana de la ciudad de méxico*. Instituto para las Personas con Discapacidad de la Ciudad de México (INDEPEDI CDMX). Ciudad de México, México. https://pdh.cdmx.gob.mx/storage/app/media/banner/Dic_LSM%202.pdf.

Espejel-Cabrera, J., Cervantes, J., García-Lamont, F., Ruiz Castilla, J. S., & D. Jalili, L. (2021). Mexican sign language segmentation using color based neuronal networks to detect the individual skin color. *Expert Systems with Applications*, *183*. https://doi.org/https://doi.org/10.1016/j.eswa.2021.115295.

Estrivero-Chavez, C., Contreras-Teran, M., Miranda-Hernandez, J., Cardenas-Cornejo, J., Ibarra-Manzano, M., & Almanza-Ojeda, D. (2019). Toward a mexican sign language system using human computer interface. In *2019 International Conference on Mechatronics, Electronics and Automotive Engineering (ICMEAE)*, 13–17. https://doi.org/10.1109/ICMEAE.2019.00010.

Garcia-Bautista, G., Trujillo-Romero, F., & Diaz-Gonzalez, G. (2016). Advances to the development of a basic mexican sign-to-speech and text language translator. In *Applications of Digital Image Processing XXXIX*, volume *9971*, 99713E. International Society for Optics and Photonics, SPIE. https://doi.org/10.1117/12.2238281.

García-Bautista, G., Trujillo-Romero, F., & Caballero-Morales, S. O. (2017). Mexican sign language recognition using kinect and data time warping algorithm. In *2017 International Conference on Electronics, Communications and Computers (CONIELECOMP)*, 1–5. https://doi.org/10.1109/CONIELECOMP.2017.7891832.

González, M. Á. R. (1991). *Lenguaje de signos*. Confederación Nacional de Sordos de España, Madrid. https://aprendelenguadesignos.com/wp-content/uploads/2013/02/Lenguajedesignos-libro.pdf.

González-Rodríguez, J.-R., Córdova-Esparza, D.-M., Terven, J., & Romero-González, J.-A. (2024). Towards a bidirectional mexican sign language–spanish translation system: A deep learning approach. *Technologies*, *12*(1). https://doi.org/10.3390/technologies12010007.

Gortarez-Pelayo, J. J., Morfín-Chávez, R. F., & Lopez-Nava, I. H. (2023). Daktilos: An interactive platform for teaching mexican sign language (lsm). In *Proceedings of the 15th International Conference on Ubiquitous Computing & Ambient Intelligence (UCAmI 2023)*, 264–269. Springer Nature Switzerland. https://doi.org/10.1007/978-3-031-48642-5_25.

Herrera, F. C. & Cantu, C. R. (2020). La glosa como una hipótesis: apuntes metodológicos para el estudio de la lsp. *Sorda & Sonora*, *3*, 57–83. https://www.researchgate.net/publication/355214004.

Jimenez, J., Martin, A., Uc, V., & Espinosa, A. (2017). Mexican sign language alphanumerical gestures recognition using 3d haar-like features. *IEEE Latin America Transactions*, *15*(10), 2000–2005. https://doi.org/10.1109/TLA.2017.8071247.

Ko, S.-K., Kim, C. J., Jung, H., & Cho, C. (2019). Neural sign language translation based on human keypoint estimation. *Applied sciences*, *9*(13), 2683. https://doi.org/10.3390/app9132683.

Kothadiya, D. R., Bhatt, C. M., Saba, T., Rehman, A., & Bahaj, S. A. (2023). Signformer: deepvision transformer for sign language recognition. *IEEE Access*, *11*, 4730–4739. https://doi.org/10.1109/ACCESS.2022.3231130.

Kumar Attar, R., Goyal, V., & Goyal, L. (2023). State of the art of automation in sign language: A systematic review. *ACM Trans. Asian Low-Resour. Lang. Inf. Process.*, *22*(4). https://doi.org/10.1145/3564769.

Lara-Cázares, A., Moreno-Armendáriz, M. A., & Calvo, H. (2024). Advanced hybrid neural networks for accurate recognition of the extended alphabet and dynamic signs in mexican sign language (msl). *Applied Sciences*, 14(22). https://doi.org/10.3390/app142210186.

Levenshtein, V. (1966). Binary codes capable of correcting deletions, insertions, and reversals. In *Soviet physics doklady*, volume *10*, 707–710. Soviet Union. http://profs.sci.univr.it/~liptak/ALBioinfo/2012_2013/2011_2012/files/levenshtein66.pdf.

Liu, Y., Jiang, X., Yu, X., Ye, H., Ma, C., Wang, W., & Hu, Y. (2023). A wearable system for sign language recognition enabled by a convolutional neural network. *Nano Energy*, *116*, 108767. https://doi.org/https://doi.org/10.1016/j.nanoen.2023.108767.

Martínez-Seis, B., Pichardo-Lagunas, O., Rodriguez-Aguilar, E., & Saucedo-Diaz, E.-R. (2019). Identification of static and dynamic signs of the mexican sign language alphabet for smartphones using deep learning and image processing. *Research in Computing Science*, *148*, 199–211. https://www.rcs.cic.ipn.mx/2019_148_11/Identification%20of%20Static%20and%20Dynamic%20Signs%20of%20the%20Mexican%20Sign%20Language%20Alphabet.pdf.

Martínez-Sánchez, V., Villalón-Turrubiates, I., Cervantes-Álvarez, F., & Hernández-Mejía, C. (2023). Exploring a novel mexican sign language lexicon video dataset. *Multimodal Technologies and Interaction*, *7*(8). https://doi.org/10.3390/mti7080083.

Mejía-Peréz, K., Córdova-Esparza, D.-M., & Terven, J. (2022). Automatic recognition of mexican sign language using a depth camera and recurrent neural networks. *Applied Sciences (Switzerland)*, *12*(11). https://doi.org/10.3390/app12115523.

Miah, A., Hasan, M., Okuyama, Y., Tomioka, T., Hossain, M., & Rahman, M. (2024). Spatial–temporal attention with graph and general neural network-based sign language recognition. *Pattern Analysis and Applications*, *27*(2). https://doi.org/10.1007/s10044-024-01229-4.

Morfín, C. R. F. (2023). *Reconocimiento continuo de la Lengua de Señas Mexicana*. [Tesis de Maestría, Centro de Investigación Científica y de Educación Superior de Ensenada, Baja California]. Repositorio Intitucional. https://cicese.repositorioinstitucional.mx/jspui/handle/1007/3998.

Morfín-Chávez, R., Gortarez-Pelayo, J., & Lopez-Nava, I. (2024). Fingerspelling recognition in mexican sign language (lsm) using machine learning. In *Advances in Computational Intelligence*, volume *14391*, 110–120. Springer Nature Switzerland. https://doi.org/10.1007/978-3-031-47765-2_9.

Morris, A. C., Maier, V., & Green, P. D. (2004). From wer and ril to mer and wil: improved evaluation measures for connected speech recognition. In *Interspeech 2004*, 2765–2768. https://doi.org/0.21437/Interspeech.2004-668.

Naveed, H., Khan, A. U., Qiu, S., Saqib, M., Anwar, S., Usman, M., Akhtar, N., Barnes, N., & Mian, A. (2025). A comprehensive overview of large language models. *ACM Trans. Intell. Syst. Technol.* https://doi.org/10.1145/3744746.

Ocampo, J. C. C., León, M. A. C., Bringas, J. A. S., Encinas, I. D., & Muñoz, J. G. S. (2020). Design of a glove like support for the learning of the mexican sign language. In *2020 3rd International Conference of Inclusive Technology and Education (CONTIE)*, 167–172. Institute of Electrical and Electronics Engineers Inc. https://doi.org/10.1109/CONTIE51334.2020.00038.

Pérez, L. M., Rosales, A. J., Gallegos, F. J., & Barba, A. V. (2017). Lsm static signs recognition using image processing. In *2017 14th International Conference on Electrical Engineering, Computing Science and Automatic Control (CCE)*, 1–5. Institute of Electrical and Electronics Engineers Inc. https://doi.org/10.1109/ICEEE.2017.8108885.

Rajalakshmi, E., Elakkiya, R., Subramaniyaswamy, V., Alexey, L. P., Mikhail, G., Bakaev, M., Kotecha, K., Gabralla, L. A., & Abraham, A. (2023). Multi-semantic discriminative feature learning for sign gesture recognition using hybrid deep neural architecture. *IEEE Access*, *11*, 2226–2238. https://doi.org/10.1109/ACCESS.2022.3233671.

Ramírez Sánchez, J., Rodríguez, A., Mendoza, M., & Terven, J. (2021). Real-time mexican sign language interpretation using cnn and hmm. In *Advances in Computational Intelligence*, volume *13067*, 55–68. Springer International Publishing. https://doi.org/10.1007/978-3-030-89817-5_4.

Rios-Figueroa, H., Sánchez-García, A., & Sosa-Jiménez, C. (2022). Use of spherical and cartesian features for learning and recognition of the static mexican sign language alphabet. *Mathematics*, *10*(16). https://doi.org/10.3390/math10162904.

Rosenblatt, F. (1958). The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review*, *65*(6), 386–408. https://doi.org/10.1037/h0042519.

Sainos-Vizuett, M. (2022). *Traducción de Lenguaje de Señas Mexicano a texto mediante aprendizaje profundo*. [Tesis de Maestría, Centro de Investigación Científica y de Educación Superior de Ensenada, Baja California]. Repositorio Intitucional. https://cicese.repositorioinstitucional.mx/jspui/handle/1007/3780.

Samaan, G. H., Wadie, A. R., Attia, A. K., Asaad, A. M., Kamel, A. E., Slim, S. O., Abdallah, M. S., & Cho, Y.-I. (2022). Mediapipe's landmarks with rnn for dynamic sign language recognition. *Electronics*, *11*(19), 3228. https://doi.org/10.3390/electronics11193228.

Serafín de Fleischmann, M. E. & González, P. R. (2011). *Manos con voz: Diccionario de Lengua de Señas Mexicana*. Libre Acceso, A.C. y Consejo Nacional para Prevenir la Discriminación (CONAPRED), Ciudad de México, México. https://educacionespecial.sep.gob.mx/storage/recursos/2023/05/xzrfl019nV-4Diccionario_lengua_%20Senas.pdf.

Solis, F. J., Toxqui, C., & Martinez, D. (2015). Mexican sign language recognition using jacobi-fourier moments. *Engineering*, *07*, 700–705. https://doi.org/10.4236/eng.2015.710061.

Sosa-Jiménez, C. O., Ríos-Figueroa, H. V., & Solís-González-Cosío, A. L. (2022). A prototype for mexican sign language recognition and synthesis in support of a primary care physician. *IEEE Access*, *10*, 127620–127635. https://doi.org/10.1109/ACCESS.2022.3226696.

Subramanian, B., Olimov, B., Naik, S. M., Kim, S., Park, K.-H., & Kim, J. (2022). An integrated mediapipe-optimized gru model for indian sign language recognition. *Scientific Reports*, *12*(1), 11964. https://doi.org/10.1038/s41598-022-15998-7.

Sánchez-Vicinaiz, T. J., Camacho-Pérez, E., Castillo-Atoche, A. A., Cruz-Fernandez, M., García-Martínez, J. R., & Rodríguez-Reséndiz, J. (2024). Mediapipe frame and convolutional neural networks-based fingerspelling detection in mexican sign language. *Technologies*, *12*(8). https://doi.org/10.3390/technologies12080124.

Tran, K. B., Nguyen, U. D., & Huynh, Q. T. (2023). Continuous sign language recognition using mediapipe. In *2023 International Conference on Advanced Technologies for Communications (ATC)*, 493–498. IEEE. https://doi.org/10.1109/ATC58710.2023.10318855.

Trujillo-Romero, F. & García-Bautista, G. (2023). Mexican sign language corpus: Towards an automatic translator. *ACM Trans. Asian Low-Resour. Lang. Inf. Process.*, *22*(8). https://doi.org/10.1145/3591471.

Varela-Santos, H., Morales-Jiménez, A., Córdova-Esparza, D.-M., Terven, J., Mirelez-Delgado, F. D., & Orenday-Delgado, A. (2021). Assistive device for the translation from mexican sign language to verbal language. *Computación y Sistemas*, *25*(3), 451–464. https://doi.org/10.13053/cys-25-3-3459.

Wei, S. & Lan, Y. (2023). A two-way translation system of chinese sign language based on computer vision. *arXiv preprint arXiv:2306.02144*. https://doi.org/10.48550/arXiv.2306.02144.

Yoo, H., Goncharenko, I., & Gu, Y. (2023). Real-time dynamic sign language recognition using lstm based on mediapipe hand data. In *2023 International Conference on Consumer Electronics - Taiwan (ICCE-Taiwan)*, 17–18. https://doi.org/10.1109/ICCE-Taiwan58799.2023.10226687.

Zhang, A., Lipton, Z. C., Li, M., & Smola, A. J. (2023). *Dive into Deep Learning*. Cambridge University Press. https://D2L.ai.

Zhang, F., Bazarevsky, V., Vakunov, A., Tkachenka, A., Sung, G., Chang, C.-L., & Grundmann, M. (2020). Mediapipe hands: On-device real-time hand tracking. *arXiv preprint arXiv:2006.10214*. https://doi.org/10.48550/arXiv.2006.10214.

Anexos

.1. Funcionamiento de las redes neuronales artificiales

.1.1. Perceptrón

El perceptrón, propuesto por Rosenblatt (1958), constituye uno de los primeros modelos de clasificación binaria basados en discriminantes lineales. El algoritmo transforma el vector de entrada $\mathbf{x}$ mediante una función no lineal fija para obtener un vector de características $\boldsymbol{\phi}(\mathbf{x})$ y, a partir de él, construye un modelo lineal generalizado de la forma

$$y(\mathbf{x}) \;=\; f\big(\mathbf{w}^{\mathsf{T}}\,\boldsymbol{\phi}(\mathbf{x})\big), \tag{24}$$

donde la función de activación $f(\cdot)$ es la función escalón

$$f(a) = \begin{cases} +1, & a \geq 0, \\ -1, & a < 0. \end{cases} \tag{25}$$

El vector $\boldsymbol{\phi}(\mathbf{x})$ suele incluir un término de sesgo, $\phi_0(\mathbf{x}) = 1$. Se adoptan valores objetivo $t_n \in \{+1, -1\}$ para cada patrón $\mathbf{x}_n$, de modo que $t_n = +1$ represente la clase C_1 y $t_n = -1$ la clase C_2, armonizando así la elección de la función de activación con la notación de las clases. En la Figura 42 se observa una representación gráfica del perceptrón.

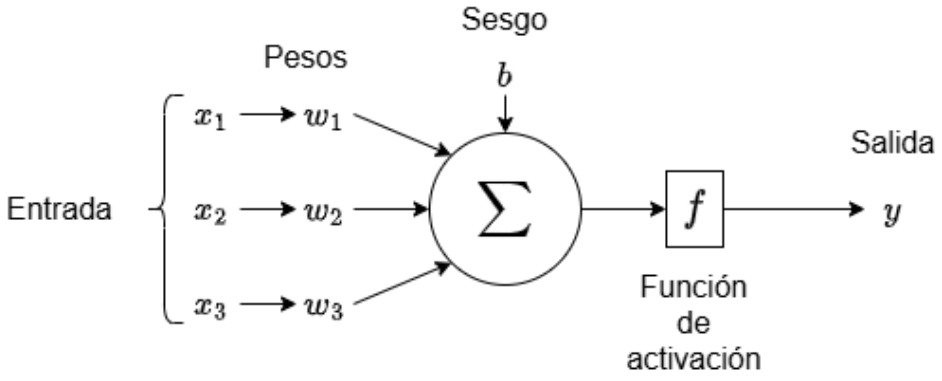


Figura 42. Esquema de arquitectura de un perceptrón.

Criterio del perceptrón Para estimar los parámetros $\mathbf{w}$ se recurre a la minimización del *criterio del perceptrón*. Sea $\mathcal{M}$ el conjunto de patrones mal clasificados por un vector de pesos $\mathbf{w}$. La función de

error se define como

$$E_P(\mathbf{w}) \;=\; -\sum_{n\in\mathcal{M}} \mathbf{w}^{\mathsf{T}}\boldsymbol{\phi}(\mathbf{x}_n)\, t_n, \tag{26}$$

la cual es lineal en $\mathbf{w}$ dentro de las regiones del espacio de pesos donde cada patrón está mal clasificado, y nula en las regiones donde todos los patrones se clasifican correctamente. El objetivo es hallar un $\mathbf{w}$ que anule $E_P(\mathbf{w})$, lo que equivale a clasificar correctamente todos los patrones.

Regla de aprendizaje Aplicando descenso estocástico del gradiente al criterio anterior se obtiene la clásica regla de actualización

$$\mathbf{w}^{(\tau+1)} \;=\; \mathbf{w}^{(\tau)} \;+\; \eta\,\boldsymbol{\phi}(\mathbf{x}_n)\, t_n, \tag{27}$$

donde $\eta > 0$ es la tasa de aprendizaje y τ indica la iteración. Para cada patrón $\mathbf{x}_n$:

- Si $t_n\,\mathbf{w}^{\mathsf{T}}\boldsymbol{\phi}(\mathbf{x}_n) > 0$ (clasificación correcta), $\mathbf{w}$ no cambia.
- Si $t_n\,\mathbf{w}^{\mathsf{T}}\boldsymbol{\phi}(\mathbf{x}_n) < 0$ (clasificación incorrecta), $\mathbf{w}$ se desplaza *hacia* $\boldsymbol{\phi}(\mathbf{x}_n)t_n$, reduciendo el error en al menos

$$-\mathbf{w}^{(\tau+1)\mathsf{T}}\boldsymbol{\phi}(\mathbf{x}_n)\, t_n \;=\; -\mathbf{w}^{(\tau)\mathsf{T}}\boldsymbol{\phi}(\mathbf{x}_n)\, t_n - \|\boldsymbol{\phi}(\mathbf{x}_n)\|_2^2. \tag{28}$$

Teorema de convergencia El *teorema de convergencia del perceptrón* garantiza que, si el conjunto de entrenamiento es linealmente separable, el algoritmo encontrará en un número finito de pasos un vector $\mathbf{w}$ que clasifique perfectamente todos los patrones. En la práctica, el número de iteraciones puede ser sustancial, y la solución final depende tanto de la inicialización de $\mathbf{w}$ como del orden de presentación de los patrones. Para conjuntos no separables, el algoritmo no converge y oscila indefinidamente, motivo por el cual se emplean extensiones como el perceptrón con márgenes o métodos basados en funciones de pérdida suaves (Bishop, 2006).

.1.2. Redes Neuronales

Las *redes neuronales artificiales* (ANN, por sus siglas en inglés) constituyen un modelo computacional inspirado en la conectividad del cerebro humano. Están formadas por unidades de procesamiento elementales, llamadas *neuronas*, organizadas en capas y enlazadas mediante conexiones dirigidas con pesos

ajustables. Cada neurona realiza la operación

$$\mathbf{a} = \mathbf{w}^\top \mathbf{x} + b, \qquad y = \phi(\mathbf{a}), \tag{29}$$

donde $\mathbf{x} \in \mathbb{R}^D$ es el vector de entrada, $\mathbf{w}$ el vector de pesos, b el sesgo y $\phi(\cdot)$ la función de activación. El aprendizaje consiste en encontrar el conjunto de parámetros $\{\mathbf{w}, b\}$ que minimiza una *función de pérdida* $\mathcal{L}$ mediante retro-propagación del gradiente (*Backward Propagation*) y un algoritmo de optimización. Su capacidad para aproximar funciones no lineales complejas las convierte en el pilar de numerosos avances recientes en visión por computadora, procesamiento del lenguaje y análisis de datos espaciales. Según Bishop (2006), la arquitectura típica de una red neuronal comprende la **capa de entrada**, una o varias **capas ocultas** y la **capa de salida**.

Capa de entrada. Presenta el vector de características $\mathbf{x}$ a la red. Por conveniencia, suele añadirse un término constante $x_0 = 1$ para incorporar el sesgo en la misma matriz de pesos.

Capas ocultas. Transforman progresivamente la representación de los datos mediante

$$\mathbf{a}^{(l)} = W^{(l)}\mathbf{z}^{(l-1)} + \mathbf{b}^{(l)}, \qquad \mathbf{z}^{(l)} = \phi\big(\mathbf{a}^{(l)}\big), \qquad l = 1, \ldots, L-1, \tag{30}$$

donde $W^{(l)} \in \mathbb{R}^{H_l \times H_{l-1}}$ son los pesos, $\mathbf{b}^{(l)} \in \mathbb{R}^{H_l}$ los sesgos y $\phi(\cdot)$ la función de activación. La profundidad (número de capas) y la anchura (número de neuronas por capa) determinan la capacidad expresiva del modelo.

Capa de salida. Traduce la representación interna al espacio de la tarea:

$$\mathbf{a}^{(L)} = W^{(L)}\mathbf{z}^{(L-1)} + \mathbf{b}^{(L)}, \qquad \mathbf{y} = g\big(\mathbf{a}^{(L)}\big), \tag{31}$$

donde $g(\cdot)$ se elige según la aplicación: la sigmoide (véase la Ec. 25) para clasificación binaria o la *softmax* para clasificación multiclase,

$$y_k = \frac{\exp(a_k)}{\sum_j \exp(a_j)}. \tag{32}$$

En conjunto, la red implementa la función compuesta

$$\mathbf{y}(\mathbf{x}; \boldsymbol{\theta}) = g\Big(W^{(L)}\,\phi\big(W^{(L-1)}\,\phi\big(\ldots\phi\big(W^{(1)}\mathbf{x} + \mathbf{b}^{(1)}\big)\ldots\big) + \mathbf{b}^{(L-1)}\big) + \mathbf{b}^{(L)}\Big), \tag{33}$$

donde $\boldsymbol{\theta} = \{W^{(l)}, \mathbf{b}^{(l)}\}_{l=1}^{L}$ agrupa todos los parámetros entrenables. Gracias a esta estructura de capas diferenciables, las redes neuronales actúan como *aproximadores universales* capaces de representar funciones arbitrariamente complejas siempre que dispongan de suficiente capacidad y datos de entrenamiento, lo que explica su relevancia en las aplicaciones modernas de *deep learning*. La representación grafica más común de una red neuronal es mediante grafos, en la Figura 43 podemos ver este esquema.

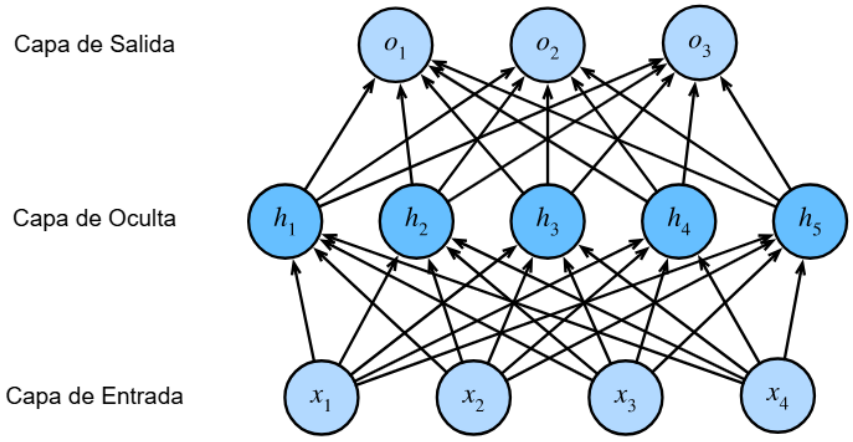


Figura 43. Esquema de arquitectura de un red neuronal tomada de Zhang et al. (2023).

Forward Propagation. La propagación hacia delante (*forward propagation*) es el recorrido determinista que, a partir de un patrón de entrada $\mathbf{x}$, calcula secuencialmente las salidas de cada capa y almacena las variables intermedias que luego se requieren para la *backward propagation*. Para una red con una sola capa oculta, los pasos son: $\mathbf{z} = \mathbf{W}^{(1)}\mathbf{x}, \qquad \mathbf{h} = \phi(\mathbf{z}), \qquad \mathbf{o} = \mathbf{W}^{(2)}\mathbf{h}$, donde $\mathbf{W}^{(1)} \in \mathbb{R}^{h\times d}$ y $\mathbf{W}^{(2)} \in \mathbb{R}^{q\times h}$ son las matrices de pesos, $\mathbf{z}$ la pre-activación, y $\mathbf{h}$ la activación de longitud h tras aplicar la función no lineal $\phi(\cdot)$. Obtenida la predicción $\mathbf{o}$, se evalúa la pérdida sobre un ejemplo $(\mathbf{x}, y)$ mediante $L = \ell(\mathbf{o}, y)$, y, si se emplea regularización L_2, se añade

$$s = \frac{\lambda}{2}\left(\|\mathbf{W}^{(1)}\|_F^2 + \|\mathbf{W}^{(2)}\|_F^2\right), \tag{34}$$

donde λ es un hiperparámetro de la regularización, de modo que la función objetivo total queda $J = L+s$.

Este paso hacia delante proporciona tanto la salida $\mathbf{o}$ como las variables $\mathbf{z}$ y $\mathbf{h}$, indispensables para el cálculo eficiente de los gradientes durante la fase de *backward propagation*.

Backward Propagation. Es el mecanismo que permite calcular de manera sistemática y eficiente los gradientes de la función objetivo J con respecto a todos los parámetros entrenables de la red. Su punto

de partida es el error en la capa de salida, $\frac{\partial J}{\partial \mathbf{o}}$, obtenido tras evaluar la pérdida entre la predicción y la etiqueta. A partir de ahí, el algoritmo recorre la red en sentido inverso al *forward pass*, aplicando la regla de la cadena para: (i) retro-propagar el error a la capa anterior y (ii) combinar dicho error con las activaciones almacenadas durante la propagación directa, generando los gradientes de cada matriz de pesos y vector de sesgos. Este flujo inverso se repite capa por capa hasta llegar a los primeros parámetros, reutilizando los valores intermedios ya computados y manteniendo así un coste computacional del mismo orden que el recorrido hacia delante. Al concluir, se obtiene un conjunto completo de gradientes $\nabla_{\boldsymbol{\theta}} J$; éstos se entregan al algoritmo de optimización (e.g. Adam), que actualiza los parámetros con el fin de reducir la pérdida en la siguiente iteración y, en última instancia, aprender el modelo a partir de los datos.

Dropout. Es una técnica de regularización estocástica que reduce el sobreajuste en redes neuronales al **anular** ("desconectar´´) aleatoriamente una fracción p de las unidades (neuronas) y sus conexiones durante cada iteración de entrenamiento. En la práctica, cada activación intermedia h se reemplaza por

$$h' = \begin{cases} 0, & \text{con probabilidad } p, \\ \dfrac{h}{1-p}, & \text{con probabilidad } 1-p. \end{cases} \tag{35}$$

de modo que el valor esperado se mantiene inalterado $(E[h'] = h)$ y no es necesario escalar la salida en la fase de inferencia. Al forzar que las capas posteriores no dependan excesivamente de patrones especificos de activación, el modelo aprende representaciones más robustas, equivalentes a promediar muchas subredes ligeras, lo que se traduce en **mejor capacidad de generalización** frente a datos no vistos.

.1.2.1. Funciones de activación

ReLU. La función más popular, debido a su simplicidad de implementación como a su buen rendimiento en variedad de tareas, ReLU proporciona una transformación no lineal muy simple. Dado un elemento x, la función se define como el máximo de ese elemento y 0:

$$ReLU(x) = \max(x, 0) \tag{36}$$

Informalmente, la función ReLU retiene sólo los elementos positivos y destacar todos los negativos poniendo las activaciones correspondientes a 0 (Zhang et al., 2023).

Softmax. Convierte un vector de valores reales, en una distribución de probabilidad discreta, de modos que cada componente es positivo y la suma total vale 1. Sea $\mathbf{a} = (a_1, \cdots, a_k)^\mathsf{T}$ el vector de valores reales que produce la última capa lineal de la red; la función softmax es definida por la Ec. 32, por lo que $\mathbf{y}$ puede interpretarse como la probabilidad de pertenecía a cada clase.

.1.2.2. Función de pérdida

La función *softmax* produce el vector de probabilidades predichas $\hat{\mathbf{y}} = (\hat{y}_1, \ldots, \hat{y}_K)^\mathsf{T}$, que interpretamos como la estimación de las probabilidades condicionales de cada clase dado el patrón de entrada $\mathbf{x}$; por ejemplo, $\hat{y}_1 = P\big(y = C_1 \mid \mathbf{x}\big)$, donde C_1 es una de las K clases posibles. Para medir la discrepancia entre esta distribución pronosticada y la distribución real de la etiqueta se utiliza la **Categorical Croos-Entropy Loss**. Sea $\mathbf{t} = (t_1, \ldots, t_K)^\mathsf{T}$ el vector *one-hot* que codifica la clase verdadera —es decir, $t_{k^*} = 1$ si el ejemplo pertenece a la clase C_{k^*} y $t_k = 0$ en otro caso— la pérdida por patrón se define como

$$\mathcal{L}_{\mathrm{CE}}\big(\hat{\mathbf{y}}, \mathbf{t}\big) = -\sum_{k=1}^{K} t_k \, \log \hat{y}_k. \tag{37}$$

Esta expresión equivale al negativo del logaritmo de la verosimilitud de un modelo softmax; minimizarla implica, por tanto, ajustar los parámetros de la red bajo el principio de máxima verosimilitud. Además, su derivada con respecto a los *logits*[1] es particularmente simple:

$$\frac{\partial \mathcal{L}_{\mathrm{CE}}}{\partial a_k} = \hat{y}_k - t_k, \tag{38}$$

lo que proporciona gradientes estables y eficientes para la *Backward Propagation* en tareas de clasificación multiclase excluyente.

[1] Los *logits* son las salidas lineales $\mathbf{a}^{(L)}$ antes de aplicar la *softmax*.

.1.2.3. Algoritmos de optimización

Una vez definida la función de pérdida, el siguiente paso consiste en seleccionar un algoritmo capaz de recorrer el espacio de parámetros para minimizarla. En optimización, la función de pérdida se denomina **función objetivo**. Sin embargo, conviene remarcar que (aunque estrechamente relacionados) los objetivos de la optimización y del aprendizaje profundo no son idénticos: la primera busca el mínimo global de $\mathcal{L}$, mientras que el segundo se conforma con encontrar un modelo que **generalice** bien a partir de un conjunto finito de datos y, por tanto, no necesariamente requiere la solución óptima absoluta Zhang et al. (2023). A continuación se resumen dos algoritmos fundamentales.

Descenso del gradiente. Sea $\boldsymbol{\theta}_t$ el vector que agrupa todos los parámetros en la iteración t. El descenso del gradiente actualiza estos parámetros en la dirección de la pendiente descendente de la función de pérdida:

$$\boldsymbol{\theta}_{t+1} \;=\; \boldsymbol{\theta}_t \;-\; \eta\, \nabla_{\boldsymbol{\theta}} \mathcal{L}\big(\boldsymbol{\theta}_t\big), \tag{39}$$

donde $\eta > 0$ es la tasa de aprendizaje. En la práctica se emplea casi siempre su variante *mini-batch* (mSGD), que evalúa el gradiente sólo sobre un subconjunto aleatorio del conjunto de entrenamiento en cada paso, reduciendo el coste computacional y añadiendo un ruido beneficioso que ayuda a escapar de los puntos de silla (*saddle points*). El algoritmo es sencillo, pero su rendimiento depende críticamente de una buena elección de η y de la escala de las variables, lo que ha motivado numerosos métodos adaptativos posteriores.

Adam. *Adaptive Moment Estimation* combina las ideas de *momentum* y *RMSProp* para ajustar automáticamente la magnitud y la dirección de cada paso de actualización. Para cada componente $g_t = \nabla_\theta \mathcal{L}(\boldsymbol{\theta}_t)$ del gradiente calcula:

$$\begin{aligned} \mathbf{m}_t &= \beta_1\, \mathbf{m}_{t-1} + (1-\beta_1)\, g_t, \\ \mathbf{v}_t &= \beta_2\, \mathbf{v}_{t-1} + (1-\beta_2)\, g_t^{\odot 2}, \end{aligned} \tag{40}$$

donde $\mathbf{m}_t$ y $\mathbf{v}_t$ son estimaciones de la **media** (primer momento) y la **varianza** (segundo momento) del gradiente, $0 < \beta_1, \beta_2 < 1$ son factores de decaimiento exponencial y $\odot$ denota producto elemento a elemento. Para corregir el sesgo inicial se usan $\hat{\mathbf{m}}_t = \mathbf{m}_t/(1-\beta_1^t)$ y $\hat{\mathbf{v}}_t = \mathbf{v}_t/(1-\beta_2^t)$, tras lo cual el

vector de parámetros se actualiza como

$$\boldsymbol{\theta}_{t+1} \;=\; \boldsymbol{\theta}_t \;-\; \eta \,\frac{\hat{\mathbf{m}}_t}{\sqrt{\hat{\mathbf{v}}_t} \;+\; \varepsilon}, \tag{41}$$

siendo ε un término pequeño ($\sim 10^{-8}$) que estabiliza la división. Adam destaca por requerir poca sintonización manual, converger con rapidez inicial y funcionar bien en problemas con gradientes poco frecuentes o de gran variabilidad.

.2. Lista de Glosas y Frases

Tabla 19. Glosas agrupadas por categoría temática

Categoría	Glosas
Emergencias (29 glosas)	*"ACCIDENTE"*, *"AMBULANCIA"*, *"AYUDA"*, *"BOMBEROS"*, *"BRAZO_HINCHADO"*, *"CORTAR_ABRIR"*, *"DAÑAR"*, *"DESLIZAR_EN_CUERPO"*, *"DESMAYAR"*, *"DOLOR"*, *"EPILEPSIA"*, *"EXPLOSION"*, *"FUEGO"*, *"GOLPE"*, *"HERIDA"*, *"INFECCION"*, *"OPRESION_EN_PECHO"*, *"PALPITACION"*, *"POLICIA"*, *"PRESIONAR"*, *"RESPIRAR"*, *"ROBAR"*, *"ROSTRO_HINCHADO"*, *"SUCESO"*, *"TEMBLOR"*, *"TEMPERATURA"*, *"TOS"*, *"URGENCIA"*, *"VOMITO"*
Contexto Médico (50 glosas)	*"AHORA"*, *"ANTES"*, *"ARTICULACIONES"*, *"BONITO"*, *"BRAZO"*, *"BRAZOS"*, *"BUENO"*, *"CABEZA"*, *"CALIENTE"*, *"CANCER"*, *"CORAZON"*, *"CUERPO"*, *"CUIDAR"*, *"DIA"*, *"DIABETES"*, *"DIARREA"*, *"DIFICIL"*, *"DOCTOR"*, *"EDIFICIO"*, *"ELEVADO"*, *"EMBARAZADA"*, *"ENFERMERO"*, *"ENFERMO"*, *"ESPECIAL"*, *"ESTAR"*, *"ESTOMAGO"*, *"FARMACIA"*, *"FIEBRE"*, *"GARGANTA"*, *"GRIPE"*, *"HABER"*, *"HOSPITAL"*, *"IR"*, *"MAL"*, *"MAREADO"*, *"MEDICINA"*, *"MUCHO"*, *"NECESITAR"*, *"NOCHE"*, *"NO_PODER"*, *"PERDER"*, *"PIERNA"*, *"PRESION_ARTERIAL"*, *"PROXIMO"*, *"PULMONES"*, *"SENTIDO_DEL_GUSTO"*, *"SENTIR"*, *"SORDO"*, *"TARDE"*, *"TENER"*
Cortesía (11 glosas)	*"COMO"*, *"CONOCER"*, *"ESPAÑOL"*, *"GRACIAS"*, *"GUSTAR"*, *"HOLA"*, *"NO"*, *"NOS_VEMOS"*, *"PLATICAR"*, *"POR_FAVOR"*, *"SI"*
Ambiguas (31 glosas)	*"AHI"*, *"AMIGO"*, *"BAÑO"*, *"CARRO"*, *"CARTERA"*, *"CASA"*, *"CITA"*, *"COMIDA"*, *"COMPRAR"*, *"CUANTO"*, *"CUERPO_CORTADO"*, *"DINERO"*, *"DONDE"*, *"EL"*, *"ESTO"*, *"LLAMAR"*, *"LSM"*, *"MANEJAR"*, *"MI"*, *"NADA"*, *"NO_ENTENDER"*, *"NO_ESCUCHAR"*, *"NO_NADA"*, *"OTRA_VEZ"*, *"QUIMICOS"*, *"ROSTRO"*, *"SU"*, *"TELEFONO"*, *"TRABAJO"*, *"VOZ"*, *"YO"*

Tabla 20. Distribución temática de frases por categoría

Categoría	Frases
Emergencias (38 frases)	*"Hubo una explosión"*, *"Hay una emergencia"*, *"Ha ocurrido un accidente"*, *"Estoy herido"*, *"Tengo quemaduras en el cuerpo"*, *"Tengo quemaduras químicas"*, *"¡Auxilio!"*, *"¡Llame a una ambulancia!"*, *"¡Llame a los bomberos!"*, *"Mi amigo no está respirando"*, *"Tengo una herida abierta"*, *"Tengo cuerpo cortado"*, *"El edificio está en llamas"*, *"¡Está temblando!"*, *"Me han atacado"*, *"Me han robado mi cartera"*, *"Me han robado mi teléfono"*, *"Sufrí un accidente automovilístico"*, *"Sufrí un accidente en casa"*, *"Sufrí convulsiones"*, *"Me golpeé la cabeza"*, *"Mi amigo se golpeó la cabeza"*, *"Mi amigo está herido"*, *"Mi amigo está inconsciente"*, *"Necesito un médico"*, *"Por favor, ayúdenme"*, *"Mi amigo necesita ayuda"*, *"¡Llame a un doctor!"*, *"¡Llame a la policía!"*, *"Me cuesta trabajo respirar"*, *"Siento opresión en el pecho"*, *"Tengo el pulso acelerado"*, *"Tengo hinchazón en los brazos"*, *"Tengo hinchazón en el rostro"*, *"Sufrí un accidente"*, *"Hay un incendio"*, *"Mi amigo necesita un médico"*, *"He perdido mi cartera"*
Contexto médico (37 frases)	*"Tengo fiebre"*, *"Tengo tos"*, *"Tengo gripe"*, *"Tengo vómito"*, *"Tengo diarrea"*, *"Tengo una infección estomacal"*, *"Tengo una infección respiratoria"*, *"Me siento mareado"*, *"He perdido el sentido del gusto"*, *"Soy alérgico a un medicamento"*, *"Tengo hipertensión"*, *"Tengo diabetes"*, *"Tengo problemas cardíacos"*, *"Tengo cáncer"*, *"Estoy embarazada"*, *"Siento dolor de garganta"*, *"Siento dolor de estómago"*, *"Siento dolor pulmonar"*, *"Siento dolor de cabeza"*, *"Siento dolor en el corazón"*, *"Siento dolor en las articulaciones"*, *"Tengo dolor en la pierna"*, *"Tengo dolor en el brazo"*, *"¿Dónde está el doctor?"*, *"¿Dónde está el hospital?"*, *"¿Dónde está la farmacia?"*, *"Necesito comprar medicamento"*, *"Tengo que ir al hospital"*, *"Tengo que ir al doctor"*, *"Me siento mal"*, *"Me siento enfermo"*, *"Mi amigo se siente enfermo"*, *"Mi amigo es alérgico a un medicamento"*, *"Mi amigo tiene hipertensión"*, *"Mi amigo tiene problemas cardíacos"*, *"Tengo una cita con el doctor"*, *"Necesito hablar con un especialista"*
Cortesía (21 frases)	*"Hola, ¿cómo estás?"*, *"Buenos días"*, *"Buenas tardes"*, *"Buenas noches"*, *"Sí"*, *"No"*, *"Por favor"*, *"Gracias"*, *"De nada"*, *"Bonito día"*, *"Nos vemos pronto"*, *"Cuídate mucho"*, *"¡Mucho gusto!"*, *"No entiendo el español"*, *"No hablo español"*, *"Hablo lengua de señas"*, *"Soy sordo"*, *"Lo repites, por favor"*, *"No entiendo español"*, *"¿Cuánto cuesta este producto?"*, *"¿Dónde está el baño?"*

.3. Resultados individuales del modelo ResNet optimizado

Tabla 21. Resultados por clase: F1-Score, Sensibilidad y Confusión.

Glosa	F1-Score	Sensibilidad	Confusión
ACCIDENTE	1.000	1.000	—
AHI	1.000	1.000	—
AHORA	1.000	1.000	—
AMBULANCIA	1.000	1.000	—
AMIGO	1.000	1.000	—
ANTES	1.000	1.000	—
ARTICULACIONES	0.889	1.000	—
AYUDA	1.000	1.000	—
BAÑO	1.000	1.000	—
BOMBEROS	0.968	1.000	—
BONITO	1.000	1.000	—
BRAZO	1.000	1.000	—
BRAZOS	1.000	1.000	—
BRAZO_HINCHADO	1.000	1.000	—
BUENO	0.000	0.000	INFECCION
CABEZA	1.000	1.000	—
CALIENTE	1.000	1.000	—
CANCER	1.000	1.000	—
CARRO	0.923	1.000	—
CARTERA	0.968	1.000	—
CASA	1.000	1.000	—
CITA	1.000	1.000	—
COMIDA	1.000	1.000	—
COMO	1.000	1.000	—
COMPRAR	1.000	1.000	—
CONOCER	1.000	1.000	—
CORAZON	1.000	1.000	—

Continúa en la siguiente página

Tabla 21. Resultados por glosa: F1-Score, Sensibilidad y Confusión.

Glosa	F1-Score	Sensibilidad	Confusión
CORTAR_ABRIR	0.938	1.000	—
CUANTO	1.000	1.000	—
CUERPO	0.000	0.000	CUIDAR
CUERPO_CORTADO	1.000	1.000	—
CUIDAR	0.667	1.000	—
DAÑAR	0.900	1.000	—
DESLIZAR_EN_CUERPO	1.000	1.000	—
DESMAYAR	1.000	1.000	—
DIA	0.966	0.933	TARDE
DIABETES	0.938	1.000	—
DIARREA	0.938	1.000	—
DIFICIL	1.000	1.000	—
DINERO	1.000	1.000	—
DOCTOR	1.000	1.000	—
DOLOR	1.000	1.000	—
DONDE	1.000	1.000	—
EDIFICIO	1.000	1.000	—
EL	1.000	1.000	—
ELEVADO	0.125	0.067	ARTICULACIONES
EMBARAZADA	0.968	1.000	—
ENFERMERO	1.000	1.000	—
ENFERMO	0.769	1.000	—
EPILEPSIA	1.000	1.000	—
ESPAÑOL	1.000	1.000	—
ESPECIAL	1.000	1.000	—
ESTAR	1.000	1.000	—
ESTO	1.000	1.000	—
ESTOMAGO	1.000	1.000	—
EXPLOSION	1.000	1.000	—

Continúa en la siguiente página

Tabla 21. Resultados por glosa: F1-Score, Sensibilidad y Confusión.

Glosa	F1-Score	Sensibilidad	Confusión
FARMACIA	1.000	1.000	—
FIEBRE	0.571	0.800	TEMPERATURA
FUEGO	1.000	1.000	—
GARGANTA	1.000	1.000	—
GOLPE	1.000	1.000	—
GRACIAS	1.000	1.000	—
GRIPE	1.000	1.000	—
GUSTAR	1.000	1.000	—
HABER	1.000	1.000	—
HERIDA	1.000	1.000	—
HOLA	1.000	1.000	—
HOSPITAL	1.000	1.000	—
INFECCION	0.391	1.000	—
IR	1.000	1.000	—
LLAMAR	1.000	1.000	—
LSM	1.000	1.000	—
MAL	1.000	1.000	—
MANEJAR	1.000	1.000	—
MAREADO	1.000	1.000	—
MEDICINA	1.000	1.000	—
MI	1.000	1.000	—
MUCHO	1.000	1.000	—
NADA	0.750	1.000	—
NECESITAR	0.909	0.833	DIABETES
NEUTRO	1.000	1.000	—
NO	1.000	1.000	—
NOCHE	1.000	1.000	—
NOS_VEMOS	1.000	1.000	—
NO_ENTENDER	1.000	1.000	—

Continúa en la siguiente página

Tabla 21. Resultados por glosa: F1-Score, Sensibilidad y Confusión.

Glosa	F1-Score	Sensibilidad	Confusión
NO_ESCUCHAR	1.000	1.000	—
NO_NADA	0.500	0.333	NADA
NO_PODER	1.000	1.000	—
OPRESION_EN_PECHO	0.000	0.000	RESPIRAR
OTRA_VEZ	1.000	1.000	—
PALPITACION	1.000	1.000	—
PERDER	1.000	1.000	—
PIERNA	1.000	1.000	—
PLATICAR	1.000	1.000	—
POLICIA	1.000	1.000	—
POR_FAVOR	1.000	1.000	—
PRESIONAR	1.000	1.000	—
PRESION_ARTERIAL	1.000	1.000	—
PROXIMO	1.000	1.000	—
PULMONES	1.000	1.000	—
QUIMICOS	1.000	1.000	—
RESPIRAR	0.652	1.000	—
ROBAR	1.000	1.000	—
ROSTRO	1.000	1.000	—
ROSTRO_HINCHADO	1.000	1.000	—
SENTIDO_DEL_GUSTO	1.000	1.000	—
SENTIR	1.000	1.000	—
SI	1.000	1.000	—
SORDO	1.000	1.000	—
SU	1.000	1.000	—
SUCESO	1.000	1.000	—
TARDE	0.286	0.182	ENFERMO
TELEFONO	1.000	1.000	—
TEMBLOR	0.125	0.067	INFECCION

Continúa en la siguiente página

Tabla 21. Resultados por glosa: F1-Score, Sensibilidad y Confusión.

Glosa	F1-Score	Sensibilidad	Confusión
TEMPERATURA	0.000	0.000	FIEBRE
TENER	1.000	1.000	—
TOS	1.000	1.000	—
TRABAJO	1.000	1.000	—
URGENCIA	1.000	1.000	—
VOMITO	1.000	1.000	—
VOZ	1.000	1.000	—
YO	1.000	1.000	—