



UNIVERSIDAD
DE LA REPÚBLICA
URUGUAY



FACULTAD DE
INGENIERÍA

Reconocimiento automático de glosas LSU en discurso continuo

Informe de Proyecto de Grado presentado por

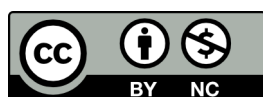
Manu Bervejillo, Facundo Techera, Emilio Valle Lisboa

en cumplimiento parcial de los requerimientos para la graduación de la carrera
de Ingeniería en Computación de Facultad de Ingeniería de la Universidad de
la República

Supervisores

Luis Chiruzzo
Ariel E. Stassi

Montevideo, 29 de junio de 2026



Reconocimiento automático de glosas LSU en discurso continuo por Manu Bervejillo, Facundo Techera, Emilio Valle
Lisboa tiene licencia [CC Atribución - No Comercial 4.0](#).

Agradecimientos

En primer lugar, queremos agradecer a nuestras familias, parejas, amigos, y compañeros de la facultad. Su compañía ha sido fundamental en estos años de carrera. Asimismo, agradecemos a los docentes de la facultad de ingeniería.

Queremos valorar también a nuestro tutor Luis por su ayuda y buena disposición durante este proyecto, y a Ariel por ser parte fundamental y siempre compartir con nosotros sus conocimientos y puntos de vista.

Finalmente, queremos agradecer a los integrantes del equipo de CIBPsi por su trabajo, que permite la existencia de proyectos como el nuestro.

Resumen

Los avances recientes en aprendizaje profundo han mejorado significativamente las soluciones a problemas de clasificación y traducción automática, impactando también en el desarrollo de sistemas de reconocimiento de lenguas de señas. Sin embargo, la investigación en este campo se ha visto limitada por la escasez de datos anotados, especialmente para lenguas poco documentadas como la Lengua de Señas Uruguaya (LSU). Recientemente, un equipo del CIBPsi ha generado el primer conjunto de datos de glosas de videos de intérpretes de LSU, habilitando por primera vez la investigación computacional sobre esta lengua.

Este proyecto aborda el reconocimiento automático de señas LSU, tanto en su variante aislada como en la detección en video continuo (*sign spotting*). A partir de 27 videos anotados con glosas, se construye un *dataset* curado de 446 glosas y 17.793 instancias mediante un pipeline de normalización, filtrado y extracción de *keypoints* corporales con MediaPipe.

Para el reconocimiento aislado se entrenan y comparan dos arquitecturas basadas en *keypoints*: SPOTER (Transformer) y MSG3D (red convolucional de grafos), obteniendo esta última los mejores resultados con una *accuracy* Top-1 de 32,7% sobre 446 clases. Para el reconocimiento continuo se implementan dos enfoques: un pipeline con segmentador temporal (MS-TCN) seguido de clasificación, y un enfoque de ventana deslizante. La evaluación se realiza mediante un sistema jerárquico de tres niveles que mide detección temporal, reconocimiento Top-5 y reconocimiento Top-1, utilizando métricas de F_1 e IoU.

Si bien el rendimiento obtenido no es suficiente para su aplicación directa en un entorno real, el trabajo constituye el primer antecedente de entrenamiento y evaluación de modelos de reconocimiento automático de LSU sobre el nuevo *dataset*, identificando fortalezas y limitaciones de cada enfoque y sentando bases para futuras investigaciones en el área.

Palabras clave: Lengua de señas uruguaya, LSU, Computer vision, Clasificación, Keypoints, Segmentador

Índice general

1. Introducción	1
1.1. Objetivos	1
1.1.1. Objetivo General	1
1.1.2. Objetivos Específicos	2
1.2. Estructura del documento	2
2. Marco teórico	3
2.1. Lenguas de señas	3
2.1.1. Lengua de Señas Uruguay	4
2.1.2. Representación mediante glosas	4
2.2. Reconocimiento de señas	5
2.2.1. Tecnologías para el procesamiento automático de lenguas de señas	5
2.2.2. Tipos de reconocimiento automático de señas	7
2.2.3. Estrategias para utilizar un modelo de reconocimiento aislado en sistemas de detección de señas	8
2.2.4. Lenguas de señas como lenguas de bajos recursos	9
2.3. Estado del arte: Modelos y arquitecturas	9
2.3.1. Modelos utilizados en este trabajo	11
2.3.2. SPOTER	11
2.3.3. Arquitecturas basadas en Grafos: GCN y MSG3D	12
2.3.4. Trabajo relacionado: reconocimiento en Lengua de Signos Catalana	13
2.3.5. Modelos de Segmentación	14
3. Pipeline de preprocesamiento	17
3.1. Dataset inicial	17
3.1.1. Videos	18
3.1.2. Archivos ELAN	19
3.2. Procesamiento de videos	19
3.2.1. Recorte y seccionado	19
3.2.2. Extracción de <i>frames</i> y <i>keypoints</i>	20
3.3. Procesamiento de archivos EAF	20
3.4. Procesado y limpieza de los datos	21

3.4.1.	Dataset antes del procesado	21
3.4.2.	Transformaciones aplicadas	23
3.4.3.	Estadísticas del dataset LSU-446	25
4.	Reconocimiento aislado: entrenamiento y experimentación	27
4.1.	SPOTER	27
4.1.1.	Formato de entrada de SPOTER	27
4.1.2.	Configuración experimental y criterios de evaluación	28
4.1.3.	Adaptaciones implementadas en el entrenamiento	30
4.1.4.	Resultados y análisis experimental	33
4.2.	Implementación MSG3D	39
4.2.1.	Formato de dataset	39
4.2.2.	Adaptaciones implementadas en el entrenamiento	40
4.2.3.	Resultados entrenando desde cero	41
4.2.4.	Resultados de <i>transfer learning</i>	43
4.3.	Análisis comparativo	45
5.	<i>Sign Spotting</i>: experimentación	47
5.1.	Metodología de evaluación	47
5.1.1.	Métricas principales	48
5.1.2.	Características del conjunto de prueba	49
5.2.	Modelos de reconocimiento aislado utilizados	50
5.3.	Modelo de segmentación para escenarios reales	51
5.3.1.	Selección del modelo de segmentación	51
5.3.2.	Integración con el reconocimiento de señas aisladas	52
5.3.3.	Resultados	53
5.4.	Enfoque de ventana deslizante	55
5.4.1.	Descripción del método	55
5.4.2.	Configuración de la ventana	56
5.4.3.	Resultados en video de prueba	57
5.4.4.	Comparación con el enfoque basado en segmentación	60
6.	Conclusiones y Trabajo Futuro	63
6.1.	Resumen de objetivos cumplidos	63
6.2.	Descubrimientos	64
6.3.	Dificultades encontradas	64
6.4.	Trabajo futuro	64
6.4.1.	Mejoras en el dataset	65
6.4.2.	Optimización de hiperparámetros	65
6.4.3.	Experimentación con nuevos modelos	65
6.4.4.	Aplicaciones al mundo real	66
	Referencias	67

A. Detalles complementarios de arquitecturas y modelos	71
A.1. Detalle de SPOTER	71
A.1.1. Datasets utilizados	71
A.1.2. Arquitectura detallada	72
A.1.3. Preprocesamiento de <i>keypoints</i>	72
A.1.4. Resultados y performance	74
A.2. Fundamentos de Redes Convolucionales de Grafos	75
A.2.1. Representación del cuerpo como grafo	75
A.2.2. Convolución en grafos	75
A.2.3. Representación matemática	75
A.3. Arquitectura MSG3D	77
A.3.1. El problema de las redes previas	77
A.3.2. El módulo G3D: unificación espacio-temporal	77
A.3.3. Agregación multi-escala desentrelazada	77
A.3.4. Bloque STGC y arquitectura general	78
A.4. Detalle del trabajo en Lengua de Signos Catalana	78
A.4.1. Adquisición y naturaleza de los datos	78
A.4.2. Preprocesamiento y curación del dataset	80
A.4.3. Experimentos y resultados completos	80
A.4.4. Contribución general	80
B. Métricas adicionales para enfoques continuos	83
B.1. Enfoque con segmentador	83
B.1.1. IoU Condicional	83
B.1.2. Variantes de Robustez para el Solapamiento	84
B.2. Enfoque de ventana deslizante	85
B.2.1. Ratios por <i>frame</i>	85
B.2.2. Impacto del filtro de suavizado	86
B.2.3. Evaluación sobre señas del vocabulario	87

Capítulo 1

Introducción

Debido a su condición y a la muy reducida cantidad de usuarios de Lengua de Señas Uruguaya (LSU), la población sorda de nuestro país enfrenta importantes barreras comunicativas en su vida cotidiana. En paralelo, los avances recientes en las áreas de Procesamiento de Lenguaje Natural y Visión por Computadora permiten hoy considerar posible el desarrollo de sistemas informáticos basados en modelos de aprendizaje profundo capaces de comprender y procesar la LSU. Esto adquiere especial relevancia teniendo en cuenta que la LSU se considera una lengua de bajos recursos, caracterizada por una disponibilidad limitada de datos anotados, herramientas y sin contar con modelos computacionales previos. Este tipo de tecnologías tiene el gran potencial de contribuir de forma significativa a la inclusión y a la mejora de la calidad de vida de la población sorda e hipoacúsica en Uruguay.

1.1. Objetivos

1.1.1. Objetivo General

Este proyecto tiene como objetivo principal el desarrollo, adaptación y validación de modelos de reconocimiento de señas tanto para el problema de reconocimiento aislado como para la tarea de *sign spotting*, que consiste en la identificación y clasificación de señas en un discurso continuo, utilizando como insumo un conjunto de videos de Lengua de Señas Uruguaya (LSU) anotados y segmentados manualmente mediante glosas. A través de este trabajo, se busca contribuir al avance de la línea de investigación que procura el desarrollo de tecnologías de procesamiento automático de lenguas de señas, aportando herramientas que apoyen el desarrollo de soluciones tecnológicas orientadas a la inclusión de la población sorda.

1.1.2. Objetivos Específicos

Para alcanzar el objetivo general, se plantean los siguientes objetivos específicos, los cuales guían las distintas etapas del proyecto:

- **Construir un dataset limpio de videos de intérpretes LSU con sus correspondientes glosas:** A partir de grabaciones en bruto y archivos de anotación, se procesarán y normalizarán los datos para eliminar inconsistencias lógicas y temporales.
- **Realizar experimentos de reconocimiento aislado de señas LSU:** Se implementarán y entrenarán diferentes arquitecturas de aprendizaje profundo (como modelos basados en grafos y transformers) utilizando los subconjuntos de datos procesados. El fin es evaluar la capacidad de estas redes para clasificar correctamente una única seña cuando sus límites temporales ya están dados, estableciendo así una línea base de rendimiento.
- **Realizar experimentos de *sign spotting* para LSU:** Se adaptarán los modelos entrenados aisladamente para que operen sobre flujos de video continuos que imitan escenarios reales. Para ello, se aplicarán estrategias como segmentadores automáticos y ventanas deslizantes, buscando no solo clasificar las señas correctamente, sino también delimitar sus fronteras temporales en discursos fluidos.

1.2. Estructura del documento

El presente documento se estructura en varios apartados. En primer lugar, se introducen los conceptos clave relacionados con el contexto de este trabajo, con el objetivo de facilitar la comprensión de las etapas posteriores. A continuación, se presentan distintos trabajos del estado del arte que comparten objetivos similares, aunque aplicados a lenguas de señas de otros países. Luego, se describe la etapa de procesamiento de los datos, cuyo propósito principal fue la limpieza, normalización y adecuación de las señas y glosas disponibles. Los dos siguientes apartados detallan los experimentos realizados y sus resultados para los escenarios de reconocimiento aislado y *sign spotting*, respectivamente. Por último, se exponen las conclusiones del trabajo y se discuten posibles líneas de mejora y extensiones futuras.

Capítulo 2

Marco teórico

En este capítulo se explican los conceptos más relevantes para el entendimiento del proyecto, también se presentan los trabajos relacionados que trabajan con lenguas de señas de otros países y constituyen el estado del arte del problema que se ataca.

2.1. Lenguas de señas

Las lenguas de señas son lenguas naturales utilizadas por las comunidades sordas e hipoacúsicas para la comunicación cotidiana. A diferencia de lo que comúnmente se asume, no constituyen una representación gestual de las lenguas orales, sino que poseen estructuras lingüísticas propias, incluyendo gramática, sintaxis y vocabulario específicos. Estas lenguas utilizan el canal visual-gestual para transmitir información, combinando configuraciones, movimientos y orientaciones de manos, expresiones faciales y posiciones corporales para construir significado.

Una característica relevante de las lenguas de señas es que no existe una lengua de señas universal. Cada región o país suele desarrollar su propia lengua, con diferencias significativas en vocabulario, gramática y formas de expresión, a pesar de que, eventualmente, el lenguaje oral más comúnmente hablado en estos países coincida.

Desde el punto de vista tecnológico, el procesamiento automático de lenguas de señas presenta desafíos particulares. A diferencia de las lenguas orales o escritas, donde existen grandes cantidades de datos textuales o de audio disponibles, las lenguas de señas requieren datos visuales anotados, lo cual implica procesos de captura, segmentación y etiquetado considerablemente más complejos. Como consecuencia, las lenguas de señas son consideradas lenguas de bajos recursos (*low-resource languages*) dentro del aprendizaje automático, ya que existe una disponibilidad limitada de datos anotados y su recolección requiere procesos complejos y costosos (Moryossef, Yin, Neubig, y Goldberg, 2021).

2.1.1. Lengua de Señas Uruguaya

La Lengua de Señas Uruguaya (LSU) constituye un caso representativo de esta situación. En Uruguay, según datos del Ministerio de Salud Pública basados en el último censo nacional, existen aproximadamente 120.000 personas hipoacúsicas y 30.000 personas sordas ([Ministerio de Salud Pública de Uruguay, 2023](#)). A pesar de esto, la disponibilidad de datos anotados, herramientas tecnológicas y estudios computacionales sobre LSU continúa siendo limitada.

Esta situación representa un desafío para el desarrollo de sistemas automáticos de reconocimiento de señas, pero al mismo tiempo evidencia la necesidad y el potencial impacto de nuevas herramientas tecnológicas que contribuyan a mejorar la accesibilidad y la inclusión de la comunidad sorda uruguaya.

2.1.2. Representación mediante glosas

En el estudio y procesamiento computacional de las lenguas de señas es habitual utilizar un sistema de representación denominado *glosa*. Una glosa consiste en una palabra o expresión escrita en una lengua oral que se utiliza como etiqueta para identificar una seña específica dentro de un video. De esta forma, cada ocurrencia de una seña en el material audiovisual puede asociarse a una etiqueta textual que facilita su anotación, análisis y posterior utilización en tareas de aprendizaje automático.

En la práctica, las glosas suelen escribirse utilizando palabras en mayúsculas que representan de forma aproximada el significado de la seña, por ejemplo *CASA*, *COMER* o *TRABAJAR*. Sin embargo, es importante destacar que una glosa no constituye una traducción exacta de la seña a la lengua oral correspondiente. Las lenguas de señas poseen estructuras gramaticales y semánticas propias, por lo que la relación entre una seña y su glosa es generalmente una simplificación destinada a facilitar la anotación lingüística y el procesamiento computacional.

Además, no todas las señas pueden representarse de manera directa mediante una única palabra del español. En la práctica, los sistemas de glosado suelen incluir distintos tipos de etiquetas para capturar fenómenos propios de la lengua de señas.

Por ejemplo, algunas glosas representan herramientas gramaticales o partículas funcionales que no poseen una traducción directa en español. En otros casos se utilizan glosas para identificar secuencias de *fingerspelling*, es decir, cuando el intérprete deletrea una palabra utilizando el alfabeto manual. También existen glosas que describen explícitamente el movimiento realizado por el intérprete, utilizadas cuando una seña no posee una convención establecida o cuando se quiere destacar una acción particular. Finalmente, ciertos sistemas de anotación incluyen marcadores para identificar gestos o expresiones que forman parte de la comunicación visual pero que no corresponden estrictamente a una seña léxica.

Estas variantes reflejan la diversidad de elementos presentes en la comunicación en lengua de señas, pero al mismo tiempo introducen complejidad en la

anotación y en el procesamiento automático de los datos.

Si bien el uso de glosas permite transformar secuencias de señas en representaciones discretas comparables con las utilizadas en el procesamiento del lenguaje natural, este sistema presenta varias limitaciones (X. Zhang y Duh, 2021). En primer lugar, las glosas dependen fuertemente de las convenciones adoptadas por el equipo encargado del etiquetado manual, lo que puede generar inconsistencias en la notación. En segundo lugar, al reducir una seña a una única etiqueta textual, se pierde parte de la información lingüística presente en el gesto original, como matices de movimiento, intensidad o expresiones faciales asociadas.

A pesar de estas limitaciones, el uso de glosas continúa siendo la estrategia más extendida para la anotación de lenguas de señas en conjuntos de datos utilizados en investigación. En este trabajo, las glosas constituyen la representación principal utilizada para etiquetar las señas presentes en los videos del *dataset*, permitiendo así entrenar y evaluar los modelos de reconocimiento desarrollados.

2.2. Reconocimiento de señas

El reconocimiento automático de señas consiste en el desarrollo de sistemas capaces de identificar qué seña está realizando una persona a partir de un video. Estos sistemas suelen utilizar modelos de *Deep Learning*, particularmente del área de *Computer Vision*.

2.2.1. Tecnologías para el procesamiento automático de lenguas de señas

Visión por computadora

El área de visión por computadora (*Computer Vision*) es la rama de la inteligencia artificial que permite a las máquinas interpretar y analizar información visual proveniente de imágenes o videos.

Los avances en *Computer Vision* han impulsado significativamente el desarrollo del reconocimiento de señas. Entre las técnicas más utilizadas se encuentran extractores de *keypoints* (concepto que se explica en la sección siguiente) como OpenPose (Cao, Hidalgo, Simon, Wei, y Sheikh, 2021) o MediaPipe (Google, 2019), redes convolucionales (CNN), arquitecturas recurrentes como LSTM (Hochreiter y Schmidhuber, 1997) para modelar secuencias y, más recientemente, modelos basados en *Transformers* (Vaswani y cols., 2017) para capturar dependencias temporales de largo alcance.

Procesamiento de la entrada

Los métodos de *computer vision* que trabajan con imágenes de cuerpos humanos en movimiento pueden realizar un procesamiento a partir de imágenes completas, construyendo sus propias *features* a partir de un vector de valores

reales que representa la matriz completa de píxeles. Otra opción es trabajar con una representación estructurada de *features* precalculadas, como los *keypoints* de posición esquelética, extraídos por otro modelo.

Modelos basados en imágenes completas. Cuando el sistema recibe los *frames* crudos, las primeras capas de la red neuronal se encargan de aprender a detectar las *features* visuales relevantes: contornos, articulaciones, movimientos, texturas y cualquier otra característica útil para distinguir una señal de otra. Este enfoque es generalmente el más robusto, ya que conserva toda la información disponible del video. Sin embargo, también implica modelos más pesados, un costo computacional más alto tanto en entrenamiento como en inferencia, y una mayor necesidad de datos para aprender adecuadamente todos los patrones visuales involucrados.

Modelos basados en *keypoints*. Los *keypoints* (o puntos clave) son coordenadas que representan la ubicación de articulaciones y regiones relevantes del cuerpo, tales como manos, brazos, rostro y torso. Esta representación captura la postura y la dinámica del señante de forma compacta, eliminando la información visual que no resulta esencial para la interpretación del movimiento. La Figura 2.1 muestra un ejemplo típico de un conjunto de *keypoints* utilizado para reconocimiento de señas.

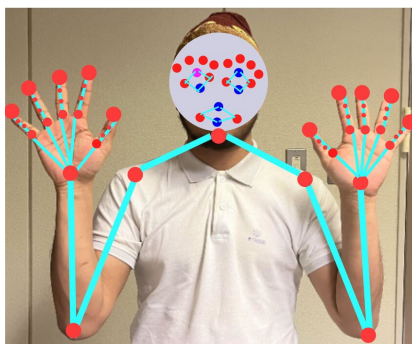


Figura 2.1: Ejemplo de extracción de *keypoints* para manos, rostro y brazos. Imagen obtenida de (Miah y cols., 2024).

Cuando el modelo trabaja únicamente con *keypoints*, la tarea se reduce a analizar patrones en las posiciones y trayectorias de manos, brazos y rostro. Esto permite utilizar arquitecturas mucho más livianas, ya que la extracción de *features* visuales queda delegada a un modelo externo. La principal ventaja de este enfoque es la disponibilidad de herramientas maduras y eficientes para la extracción de *keypoints*, como OpenPose (Cao y cols., 2021) o MediaPipe (Google, 2019), lo que facilita el desarrollo de modelos aún con *datasets* pequeños, una situación habitual en el reconocimiento de señas.

En cuanto a los extractores de *keypoints*, OpenPose utiliza una técnica denominada *Part Affinity Fields* (PAFs), que permite detectar simultáneamente las articulaciones de múltiples personas en una imagen mediante un enfoque *bottom-up*: primero identifica todos los puntos clave presentes y luego los asocia a individuos específicos mediante campos vectoriales que codifican la ubicación y orientación de las extremidades. MediaPipe, por su parte, emplea un paradigma de detector-seguidor (*detector-tracker*): ejecuta un modelo de detección más pesado en el primer cuadro y luego utiliza un seguidor ligero en los cuadros subsiguientes, reactivando la detección solo cuando la confianza del seguimiento desciende. Para la estimación de postura corporal utiliza el modelo BlazePose, y para las manos combina un detector de palma con un modelo de estimación de 21 *keypoints* tridimensionales por mano.

2.2.2. Tipos de reconocimiento automático de señas

Dentro del área de reconocimiento automático de lenguas de señas pueden distinguirse tres problemas principales: el reconocimiento aislado, el reconocimiento continuo y el *sign spotting*. Cada uno de ellos presenta diferentes niveles de complejidad y responde a distintos escenarios de aplicación (Shen, Yuan, Sheng, Du, y Yu, 2023).

Reconocimiento aislado. En este escenario se busca clasificar videos que contienen exactamente una única seña previamente segmentada. El objetivo del modelo es asignar a cada video una etiqueta correspondiente a la seña contenida.

Este problema consiste en una tarea de clasificación de secuencias de *frames* o *keypoints*, utilizando arquitecturas como redes neuronales convolucionales, modelos recurrentes o modelos basados en *transformers*. Debido a que los límites temporales de la seña ya están definidos, el modelo no necesita resolver el problema de segmentación temporal.

Además, muchos de los trabajos que se enfocan en este problema utilizan señas aisladas, que comienzan y terminan desde el reposo, y no señas extraídas de una secuencia.

Reconocimiento continuo. En el reconocimiento continuo el objetivo es interpretar una secuencia completa de señas producidas de forma natural en un video, identificando todas las señas presentes y su orden temporal. Este problema es conceptualmente similar al reconocimiento automático del habla o a la traducción automática de secuencias, ya que implica modelar dependencias lingüísticas entre señas consecutivas.

Sign spotting. El problema de *sign spotting* consiste en detectar la ocurrencia de una o varias señas dentro de un video más largo, que puede contener pausas, transiciones u otros movimientos no relevantes. En este caso, el sistema debe identificar tanto la etiqueta de la seña como su localización temporal dentro del video.

A diferencia del reconocimiento continuo, el objetivo del *sign spotting* no necesariamente es reconstruir toda la secuencia lingüística producida por un señante, sino detectar y delimitar instancias de señas específicas dentro del flujo visual.

El problema abordado en este trabajo corresponde al de *sign spotting*, ya que el objetivo es identificar y delimitar automáticamente las señas presentes en un video, asignando a cada segmento la etiqueta correspondiente.

2.2.3. Estrategias para utilizar un modelo de reconocimiento aislado en sistemas de detección de señas

En muchos casos, los sistemas de detección de señas dentro de videos se construyen utilizando modelos entrenados originalmente para reconocimiento aislado. Para ello, es necesario incorporar un mecanismo adicional que permita analizar el video completo y determinar qué porciones corresponden a cada seña.

A continuación se describen dos enfoques frecuentes para este problema.

Modelo segmentador

Esta estrategia consiste en utilizar primero un modelo que segmenta temporalmente el video, identificando los conjuntos de *frames* que contienen una seña (Amit Moryossef, 2023). Una vez detectados, cada segmento se clasifica utilizando el modelo de reconocimiento aislado.

Aunque este enfoque resulta intuitivo y modular, presenta la desventaja de la acumulación de errores: si el segmentador delimita incorrectamente una seña o mezcla dos consecutivas, el clasificador recibirá información incompleta o ruidosa.

Ventana deslizante

Otro enfoque consiste en aplicar el modelo de reconocimiento aislado sobre una ventana deslizante de tamaño fijo que recorre el video. De esta forma, el clasificador se ejecuta sobre numerosas subsecuencias solapadas (Zuo, Wei, y Mak, 2024).

Este método no requiere modelos adicionales, pero presenta varias limitaciones:

- Un tamaño de ventana fijo rara vez se ajusta correctamente a la duración de todas las señas: si es demasiado pequeño, la seña puede quedar incompleta; si es demasiado grande, incluirá *frames* irrelevantes, ruido o hasta *frames* de señas adyacentes.
- Es necesario incorporar explícitamente una clase *no-seña* para identificar momentos de reposo.

- Requiere un proceso de post-procesamiento para eliminar duplicados y determinar con mayor precisión los límites temporales de cada seña detectada.

2.2.4. Lenguas de señas como lenguas de bajos recursos

Uno de los principales desafíos del reconocimiento automático de lenguas de señas es la escasez de datos, una condición que las sitúa entre las denominadas *lenguas de bajos recursos*. Esta limitación tiene raíces tanto lingüísticas como prácticas.

En primer lugar, cada país o comunidad suele desarrollar su propia lengua de señas, con gramática, vocabulario y convenciones propias. Esta diversidad lingüística dificulta reutilizar directamente los conjuntos de datos o modelos construidos para una lengua en el contexto de otra, a diferencia de lo que ocurre con lenguas orales donde existen recursos compartidos entre idiomas emparentados.

En segundo lugar, la población usuaria de estas lenguas es reducida, lo que limita tanto la cantidad de señantes de LSU disponibles para la recolección de datos como el número de anotadores expertos capaces de realizar el etiquetado lingüístico. La anotación de lenguas de señas es un proceso particularmente costoso: requiere personal con conocimiento tanto de la lengua como de las convenciones de transcripción, y el trabajo debe realizarse manualmente a partir de material audiovisual, revisando cada seña cuadro a cuadro. A diferencia de la transcripción de audio, donde existen herramientas de reconocimiento automático del habla que aceleran el proceso, no se dispone aún de herramientas equivalentes para lenguas de señas.

Esta combinación de factores resulta en conjuntos de datos pequeños, con vocabularios limitados y distribuciones desbalanceadas, lo que impacta directamente en la capacidad de los modelos de aprendizaje automático para generalizar.

2.3. Estado del arte: Modelos y arquitecturas

En los últimos años, el reconocimiento automático de lenguas de señas ha experimentado avances significativos impulsados por el desarrollo de técnicas de *Deep Learning*, particularmente en las áreas de *Computer Vision* y *Sequence Modeling*. Este progreso ha permitido abordar tareas cada vez más complejas, pasando del reconocimiento de señas aisladas en entornos controlados a escenarios más realistas de reconocimiento continuo en videos no segmentados.

Tradicionalmente, los primeros enfoques para el reconocimiento de señas se basaban en modelos que operaban directamente sobre los píxeles de los videos, utilizando arquitecturas convolucionales 2D y 3D, como C3D (Tran, Bourdev, Fergus, Torresani, y Paluri, 2015) o I3D (Carreira y Zisserman, 2017), capaces de capturar patrones espaciales y temporales de forma conjunta. Si bien estos

modelos lograron buenos resultados, presentan un alto costo computacional y requieren grandes volúmenes de datos para generalizar correctamente.

Como alternativa, surgieron enfoques basados en representaciones estructuradas del movimiento humano, particularmente mediante la extracción de *keypoints* del cuerpo, manos y rostro. Estos métodos reducen significativamente la dimensionalidad del problema y eliminan información visual irrelevante, facilitando el aprendizaje en contextos con disponibilidad limitada de datos, una situación frecuente en lenguas de señas minoritarias.

En este contexto, las Redes Neuronales de Grafos (GCN) han demostrado ser especialmente efectivas para modelar la dinámica de los esqueletos humanos, al representar las articulaciones como nodos y sus relaciones como aristas. Modelos como ST-GCN (Yan, Xiong, y Lin, 2018) y sus extensiones han sido ampliamente utilizados en tareas de reconocimiento de acciones y posteriormente adaptados al reconocimiento de señas. En esta línea, arquitecturas más recientes como MSG3D (Ziyu Liu, Hongwen Zhang, Zhenghao Chen, Zhiyong Wang, Wanli Ouyang, 2020) introducen mecanismos para capturar dependencias espacio-temporales de múltiples escalas, logrando mejoras sustanciales en la modelización del movimiento.

En paralelo, el auge de los modelos basados en *Transformers* también ha impactado en el área. Estos modelos, originalmente propuestos para tareas de procesamiento de lenguaje natural, han demostrado una gran capacidad para capturar dependencias de largo alcance en secuencias. En el dominio de lenguas de señas, trabajos como (Camgoz, Koller, Hadfield, y Bowden, 2020) proponen arquitecturas *end-to-end* capaces de realizar reconocimiento continuo e incluso traducción a lenguaje natural, combinando información visual con modelado lingüístico.

Otra línea de investigación, relevante para los sistemas que adoptan la estrategia que integra un modelo segmentador, corresponde, precisamente, a la segmentación temporal de señas dentro de videos continuos. Esta tarea, estrechamente relacionada con el *sign spotting*, busca identificar automáticamente los intervalos donde ocurren las señas, constituyendo un paso clave para construir sistemas completos de reconocimiento continuo. Los enfoques actuales formulan este problema como una tarea de etiquetado a nivel de *frame*, utilizando modelos secuenciales como redes convolucionales temporales (TCN) o variantes híbridas.

En conjunto, estas líneas de trabajo reflejan una evolución desde modelos monolíticos altamente dependientes de grandes volúmenes de datos hacia enfoques más modulares y eficientes, capaces de adaptarse a contextos de bajos recursos. Esta última característica resulta especialmente relevante para lenguas de señas como la Lengua de Señas Uruguaya (LSU), donde la disponibilidad de datos anotados es limitada.

En este trabajo se adopta una combinación de estos enfoques, utilizando modelos basados en *keypoints* tanto para el reconocimiento de señas aisladas como para la segmentación temporal, priorizando arquitecturas que permitan un buen desempeño en escenarios con restricciones de datos y cómputo.

2.3.1. Modelos utilizados en este trabajo

A partir del análisis del estado del arte presentado anteriormente, se seleccionaron un conjunto de modelos representativos para abordar el problema de reconocimiento automático de señas en Lengua de Señas Uruguaya (LSU). La elección de estas arquitecturas estuvo guiada por dos criterios principales: su desempeño reportado en tareas similares y su adecuación a escenarios con disponibilidad limitada de datos.

En particular, se optó por trabajar con modelos basados en *keypoints*, en lugar de enfoques que operan directamente sobre los fotogramas de video. Esta decisión responde a que las representaciones esqueléticas permiten reducir significativamente la complejidad del problema, eliminando información visual irrelevante y facilitando el entrenamiento en contextos de bajos recursos, como es el caso de la LSU.

Dentro de este enfoque, se seleccionaron dos arquitecturas principales para el reconocimiento aislado de señas:

- **SPOTER** (Boháček y Hruz, 2022): un modelo basado en *Transformers* diseñado para procesar secuencias de *keypoints*, destacándose por su eficiencia computacional y su capacidad de generalización en *datasets* pequeños.
- **MSG3D** (Ziyu Liu, Hongwen Zhang, Zhenghao Chen, Zhiyong Wang, Wanli Ouyang, 2020): una arquitectura basada en Redes Convolucionales de Grafos, capaz de modelar de manera conjunta las relaciones espaciales y temporales entre articulaciones del cuerpo.

Ambos modelos fueron entrenados y evaluados en el contexto de reconocimiento aislado, permitiendo comparar dos paradigmas arquitectónicos distintos: modelos basados en atención y modelos basados en grafos.

Posteriormente, estos modelos fueron integrados en sistemas de *sign spotting* bajo dos estrategias diferentes: un enfoque basado en segmentación previa del video y un enfoque de ventana deslizante. Para la segmentación, se utilizó un modelo preentrenado propuesto en (Amit Moryossef, 2023), el cual permite identificar automáticamente los límites temporales entre señas.

Esta combinación de modelos y estrategias permitió analizar de forma sistemática las ventajas y limitaciones de cada enfoque en un escenario realista, caracterizado por la variabilidad de los datos y la ausencia de segmentaciones explícitas.

2.3.2. SPOTER

SPOTER (*Sign POse-based TransformER*) es un modelo basado en *keypoints* (Boháček y Hruz, 2022). Se distingue en el área de los modelos basados en poses por combinar una arquitectura basada en *Transformers* con un *pipeline* de preprocesamiento geométrico de *keypoints*, lo que le permitió obtener resultados competitivos con un costo computacional reducido.

El modelo fue evaluado en dos *datasets* de referencia: WLASL (*Word-Level American Sign Language*) y LSA64 (Lengua de Señas Argentina). En el subconjunto WLASL-100 alcanzó una precisión *top-1* del 63.18 %, mientras que en LSA64 obtuvo un 100 % de precisión, demostrando su capacidad de generalización incluso con pocos datos (Boháček y Hruz, 2022). Los detalles de estos *datasets* y los resultados completos se presentan en el Anexo A.1.1.

La arquitectura de SPOTER adapta el *Transformer* original de Vaswani et al. (Vaswani y cols., 2017) al procesamiento de secuencias temporales de poses humanas. Cada seña se representa como una secuencia de vectores con las coordenadas 2D de 54 *keypoints* (cuerpo, manos y rostro). La secuencia es procesada por un codificador y un decodificador que utiliza una consulta única (*Class Query*) para la clasificación final. La Figura A.1 (ver Anexo A.1) muestra la arquitectura completa.

El *pipeline* de preprocesamiento incluye técnicas de normalización que transforman las poses a proporciones corporales universales y normalizan las manos en coordenadas locales, así como aumentados espaciales (rotación, compresión lateral, transformación de perspectiva y rotación de articulaciones) que mejoran la robustez del modelo. En el Anexo A.1 se detallan estas técnicas y se presenta el estudio de ablación que cuantifica sus contribuciones.

SPOTER resulta especialmente relevante para LSU por su eficiencia computacional (5.92 millones de parámetros), su capacidad de generalizar con conjuntos de datos reducidos y su *pipeline* de normalización que mitiga posibles particularidades por parte de los señantes a la hora de realizar las señas y condiciones variables de grabación.

2.3.3. Arquitecturas basadas en Grafos: GCN y MSG3D

Las Redes Convolucionales de Grafos (GCN) constituyen una familia de modelos diseñados para procesar datos representados en forma de grafos. En el contexto de lenguas de señas, el cuerpo del señante se modela como un grafo no dirigido donde los nodos representan articulaciones (manos, codos, hombros, etc.) y las aristas representan las conexiones físicas entre ellas. Este enfoque permite capturar de manera natural las relaciones espaciales entre las distintas partes del cuerpo durante la ejecución de una seña. Los fundamentos matemáticos de las GCN se presentan en el Anexo A.2.

Sobre esta base, Liu et al. (Ziyu Liu, Hongwen Zhang, Zhenghao Chen, Zhiyong Wang, Wanli Ouyang, 2020) propusieron MSG3D (*Multi-Scale Graph Convolutional 3D Network*), una arquitectura que introduce dos innovaciones clave para el modelado de secuencias de esqueletos:

- **Módulo G3D (*Graph 3D*):** A diferencia de enfoques tradicionales que procesan espacio y tiempo de forma independiente, G3D construye un grafo unificado en el dominio espacio-temporal. Esto permite capturar relaciones simultáneas entre articulaciones en diferentes instantes de tiempo, algo fundamental para modelar movimientos coordinados como los que ocurren en las lenguas de señas.

- **Agregación multi-escala desentrelazada:** Para capturar relaciones a distintas distancias en el esqueleto (desde interacciones locales entre dedos hasta relaciones globales entre manos y cabeza), MSG3D introduce un mecanismo que aísla cada escala de vecindad, evitando el problema de ponderación sesgada presente en enfoques anteriores que utilizan potencias de la matriz de adyacencia.

La arquitectura de MSG3D se organiza en una pila de bloques STGC (*Spatio-Temporal Graph Convolution*), cada uno con dos vías complementarias: una vía G3D que modela relaciones espacio-temporales directas, y una vía factorizada que captura dependencias de largo alcance en espacio y tiempo por separado. Una descripción detallada de la arquitectura y sus componentes se presenta en el Anexo A.3.

Esta arquitectura resulta especialmente adecuada para lenguas de señas, donde el significado de una seña depende de la sincronización precisa entre movimientos manuales y postura corporal en una ventana de tiempo corta.

2.3.4. Trabajo relacionado: reconocimiento en Lengua de Signos Catalana

El trabajo de Garmendia et al. (Naiara Garmendia, Horacio Saggion y Euan McGill, 2025a) constituye una referencia fundamental para este proyecto, ya que aplica la arquitectura MSG3D al reconocimiento de señas aisladas en Lengua de Señas Catalana (LSC), un escenario con características similares a la LSU en términos de disponibilidad de datos y variabilidad.

El estudio se basa en el *LSC Corpus*, un recurso que contiene diálogos y monólogos espontáneos, a diferencia de la mayoría de los *datasets* existentes que utilizan señas aisladas y preparadas. A partir de este corpus, los autores realizaron un proceso de curación que incluyó segmentación temporal de las señas, agrupación de variaciones de una misma glosa bajo lemas principales, y filtrado por frecuencia, resultando en un conjunto final de 105 clases con más de 50 muestras cada una. Los detalles de este proceso se presentan en el Anexo A.4.

Para la clasificación, utilizaron MSG3D con *keypoints* extraídos mediante MediaPipe Holistic. Los autores evaluaron dos estrategias: entrenamiento desde cero y *transfer learning* utilizando un modelo pre-entrenado con el *dataset* SWL-LSE (Lengua de Señas Española). Los resultados mostraron que el pre-entrenamiento mejoró la precisión *top-1* en un 6.61 %, alcanzando un 49.24 % sobre el conjunto de 105 clases (Naiara Garmendia, Horacio Saggion y Euan McGill, 2025a).

Este trabajo resulta relevante para nuestro proyecto por varias razones: principalmente, al igual que en el contexto de este trabajo, los datos de entrada son videos completos, sin segmentar en señas individuales. Además, valida el uso de MSG3D en un contexto de lengua de señas con datos espontáneos, demuestra la viabilidad del *transfer learning* entre lenguas de señas, y proporciona una base experimental reproducible que hemos adaptado para el caso de LSU.

2.3.5. Modelos de Segmentación

En el contexto del reconocimiento automático de lenguas de señas, la segmentación temporal constituye una etapa fundamental del *pipeline*. A diferencia del reconocimiento de señas aisladas, donde cada instancia de entrada corresponde a un único signo previamente delimitado, los escenarios reales de uso involucran secuencias continuas de señas sin marcas explícitas de inicio y fin. La tarea de segmentación busca, por tanto, identificar automáticamente los límites temporales entre signos consecutivos dentro de una secuencia continua de video.

En este trabajo se toma como referencia el modelo de segmentación, propuesto por Moryossef et al. (Amit Moryossef, 2023), quienes abordan el problema desde una perspectiva de modelado secuencial aplicada a lenguas de señas continuas, con especial énfasis en la generalización a distintos dominios y señantes.

La segmentación se plantea como un problema de etiquetado temporal a nivel de *frame*, utilizando un esquema de anotación BIO (*Begin-Inside-Outside*) donde cada *frame* se clasifica como inicio de seña, interior de seña, o fuera de cualquier seña. Este enfoque evita la necesidad de detectar explícitamente cada seña como una unidad independiente y permite modelar directamente la continuidad temporal propia de la lengua de señas.

Los modelos operan sobre secuencias de *features* derivadas de estimación de postura corporal y manual (*keypoints*), lo cual reduce la variabilidad asociada al fondo, la iluminación y la identidad del señante. Esta decisión es consistente con los enfoques basados en esqueletos utilizados en tareas de reconocimiento aislado, y refuerza la idea de un *pipeline* unificado donde segmentación y clasificación comparten el mismo espacio de representación.

En particular, se emplean arquitecturas de tipo *Temporal Convolutional Networks* (TCN) con convoluciones causales y dilatadas, que permiten ampliar el campo receptivo temporal sin incrementar excesivamente la profundidad del modelo. Esto resulta especialmente relevante en lengua de señas, donde la duración de una seña puede variar considerablemente y donde los movimientos preparatorios y de transición aportan información clave para la detección de límites.

Uno de los principales desafíos en la segmentación de señas es la coarticulación: los movimientos de transición entre señas no son abruptos y, en muchos casos, comparten características tanto del signo precedente como del siguiente. Los modelos propuestos abordan este problema aprendiendo patrones temporales suaves, evitando decisiones binarias rígidas basadas en umbrales heurísticos. De este modo, la segmentación se aprende de manera implícita a partir de los datos, permitiendo una mayor robustez frente a variaciones individuales en estilo, velocidad y expresividad del señante.

Los resultados reportados por Moryossef et al. (Amit Moryossef, 2023) muestran que los modelos basados en convoluciones temporales superan enfoques más simples y generalizan adecuadamente a distintos conjuntos de datos, evidenciando su potencial como componente previo a sistemas de reconocimiento de señas aisladas o continuas.

La inclusión de modelos de segmentación automáticos resulta clave para

extender sistemas de reconocimiento de señas aisladas hacia soluciones de *sign spotting*, que constituye el objetivo final de este trabajo.

Capítulo 3

Pipeline de preprocesamiento

En este capítulo se describe el proceso de desarrollo del pipeline de preprocesamiento de los datos utilizados en el presente proyecto, así como las decisiones metodológicas adoptadas para su construcción. Dicho pipeline tiene como objetivo transformar los datos originales —material audiovisual anotado— en una representación adecuada para el entrenamiento y evaluación de los modelos de reconocimiento de lengua de señas.

La etapa de preprocesamiento presentó múltiples desafíos, tanto asociados a la naturaleza de los datos como a los aspectos técnicos de su manipulación. En particular, se identificaron problemas relacionados con la calidad variable de los videos, la falta de estandarización en la notación del glosado y la presencia de segmentos irrelevantes o tiempos muertos en las grabaciones. Asimismo, se enfrentaron dificultades técnicas vinculadas al procesamiento de video, como la correcta sincronización temporal entre las anotaciones de glosas y los fragmentos de video correspondientes luego de cada transformación.

A lo largo de este capítulo se detallan las estrategias empleadas para abordar estos desafíos, así como la estructura final del pipeline de preprocesamiento resultante, el cual constituye un componente fundamental para el correcto funcionamiento de los sistemas de reconocimiento desarrollados en este trabajo.

3.1. Dataset inicial

El dataset inicial utilizado (Fojo y cols., 2024) en este proyecto está compuesto por 27 archivos de video en formato `.mp4`, cada uno acompañado de sus respectivas anotaciones lingüísticas en archivos *ELAN*.

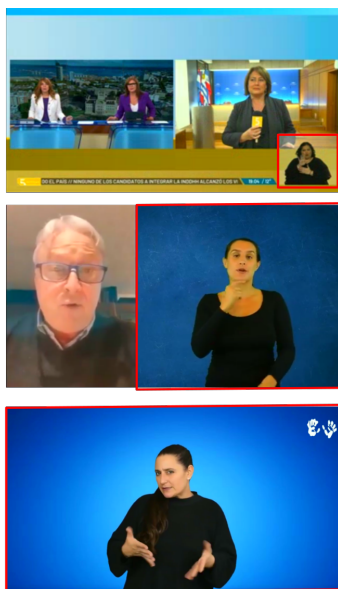


Figura 3.1: Ejemplos de posición de intérpretes en distintos videos.

3.1.1. Videos

Los videos corresponden a distintos tipos de contenido audiovisual, incluyendo programas de televisión y narraciones de cuentos clásicos, tales como *El Lazarillo de Tormes* y *La Bella Durmiente*.

La disposición visual del o la intérprete presenta una alta variabilidad entre los distintos videos. Como se puede apreciar en la Figura 3.1, en algunos casos, la persona se encuentra centrada y en primer plano, mientras que en otros aparece desplazada hacia uno de los lados del encuadre o ubicada dentro de un recuadro de menor tamaño en alguna de las esquinas del video. Esta heterogeneidad introduce desafíos adicionales para las etapas posteriores de procesamiento automático.

Adicionalmente, los videos presentan una amplia variabilidad en cuanto a su resolución espacial, incluyendo casos de resoluciones muy bajas, lo cual afecta directamente la calidad de la información visual disponible para la estimación de pose y la extracción de *features*. Todos los videos fueron grabados con una frecuencia de muestreo uniforme de 25 cuadros por segundo (fps).

Asimismo, especialmente en los videos correspondientes a programas televisivos, se identifican segmentos temporales prolongados —de varios segundos e incluso minutos— en los que el intérprete no es visible o no está realizando señas. Estos intervalos suelen corresponder a pausas publicitarias u otros contenidos no relevantes para el reconocimiento de lengua de señas, y deben ser considerados durante la etapa de preprocesamiento.

3.1.2. Archivos ELAN

Los datos brutos del proyecto consisten en archivos EAF (ELAN Annotation Format) (Wittenburg, Brugman, Russel, Klassmann, y Sloetjes, 2006), un formato XML estandarizado utilizado para la anotación de datos multimedia en estudios lingüísticos. Cada archivo EAF contiene una estructura jerárquica que incluye un encabezado (<HEADER>) con referencias al archivo de video asociado, un orden temporal (<TIME_ORDER>) que define los intervalos de tiempo mediante slots (<TIME_SLOT>), y capas de anotación (<TIER>) que almacenan las glosas en Lengua de Señas Uruguaya. Las anotaciones están organizadas en unidades alineables temporalmente (<ALIGNABLE_ANNOTATION>) que asocian cada glosa con intervalos específicos de tiempo, permitiendo una sincronización precisa entre el contenido visual del video y su transcripción glosada. Los archivos van desde cientos hasta miles de anotaciones.

El etiquetado manual de estos videos y consecuente generación de estos archivos *EAF* fue trabajo de Fojo et al. (Fojo y cols., 2024).

3.2. Procesamiento de videos

A partir de las características y limitaciones del dataset descritas en la sección anterior, resulta necesario aplicar una serie de transformaciones sobre los videos originales antes de su utilización en los modelos de reconocimiento. El objetivo principal de este procesamiento es reducir, en la medida de lo posible, la presencia de segmentos irrelevantes —como períodos prolongados sin actividad de señas— y eliminar regiones de la imagen que no aportan información útil para el análisis.

Asimismo, los videos deben ser convertidos a un formato de datos adecuado para su posterior procesamiento por los modelos de aprendizaje automático. Esto implica descomponer cada video en sus cuadros individuales y extraer, a partir de ellos, representaciones estructuradas del movimiento mediante la obtención de *keypoints* corporales.

Las subsecciones siguientes detallan cada una de las etapas que componen este proceso, incluyendo el recorte temporal y espacial de los videos, así como la extracción de características necesarias para el entrenamiento y evaluación de los modelos.

3.2.1. Recorte y seccionado

El objetivo de esta etapa fue eliminar la mayor cantidad posible de información irrelevante presente en los videos originales, utilizando para ello herramientas de procesamiento de video como *ffmpeg* (FFmpeg Developers, 2000) y OpenCV (Bradski y Kaehler, s.f.) en Python. Para lograrlo, se aplicaron tanto recortes espaciales de la imagen como recortes temporales de los videos, de modo que el encuadre se ajuste de la forma más precisa posible a la persona realizando las señas y se eviten intervalos prolongados sin actividad relevante.

En particular, se estableció como criterio que no existieran segmentos de video con más de 30 segundos consecutivos sin una señal anotada. Este umbral se definió con el fin de reducir la presencia de contenido no informativo y facilitar las etapas posteriores de entrenamiento y evaluación.

En una etapa inicial del proyecto, y con el objetivo de evaluar de forma rigurosa la capacidad de generalización de los modelos, se decidió realizar la división de los datos de entrenamiento, validación y prueba separando por video completo. En este contexto, los intervalos prolongados sin señales fueron interpretados como posibles cambios de contexto. Por esta razón, cuando se detectaba un intervalo sin señales superior a 30 segundos, el video original se segmentaba en dos partes independientes. El primer segmento finalizaba dos segundos después del inicio del intervalo, mientras que el segundo comenzaba dos segundos antes del final del mismo.

Adicionalmente, se aplicó un recorte temporal al inicio y al final de cada video, de modo que no existieran más de dos segundos sin señales antes de la primera glosa ni después de la última glosa anotada.

Cabe destacar que la aplicación de estos recortes temporales introdujo múltiples desafíos en cuanto a la correcta sincronización entre los archivos de anotaciones de glosas y los videos resultantes. La resolución de estos problemas requirió varias iteraciones de depuración y verificación, hasta garantizar la correspondencia temporal precisa entre ambos tipos de datos.

3.2.2. Extracción de *frames* y *keypoints*

La primera etapa del procesamiento consistió en la extracción de los fotogramas individuales a partir de los videos resultantes del recorte y seccionado. Para ello se utilizó la herramienta *ffmpeg*, configurada para obtener una frecuencia de muestreo uniforme de 25 cuadros por segundo (fps), consistente con la frecuencia original de los videos del dataset.

Posteriormente, se realizó la extracción de *keypoints*, también referidos como *features* visuales, a partir de los fotogramas obtenidos. Este proceso se llevó a cabo utilizando la biblioteca *MediaPipe* (Google, 2019). No obstante, la configuración de los *keypoints* extraídos difiere según el tipo de modelo considerado. En particular, se definieron conjuntos de *keypoints* específicos para cada una de las dos arquitecturas de modelos de reconocimiento aislado y para el modelo de segmentación.

3.3. Procesamiento de archivos EAF

Si bien los archivos *EAF* constituyen un formato adecuado para el almacenamiento de anotaciones lingüísticas de videos, no resultan prácticos ni directamente utilizables para el desarrollo de modelos de aprendizaje automático. Por este motivo, el primer paso en el procesamiento de las glosas consistió en convertir cada archivo *EAF* a un archivo en formato *CSV*.

Cada archivo *CSV* generado contiene tres columnas, correspondientes a la glosa y a sus tiempos de inicio y fin. Esta representación permite una manipulación más sencilla de las anotaciones, facilitando su lectura, parseo y procesamiento en las etapas posteriores del pipeline.

Una vez realizada la conversión a formato *CSV*, se aplicaron sobre estos archivos las mismas operaciones de segmentación temporal que se efectuaron sobre los videos. En particular, algunos archivos *CSV* fueron divididos en múltiples archivos independientes, y los tiempos de inicio y fin de las glosas fueron ajustados de manera consistente con los recortes temporales realizados sobre los videos correspondientes.

Cabe destacar que esta etapa también presentó desafíos significativos en términos de sincronización entre las anotaciones y los videos procesados, contribuyendo a parte de las dificultades de alineación temporal mencionadas previamente.

3.4. Procesado y limpieza de los datos

En las secciones anteriores se describieron transformaciones aplicadas principalmente al formato de los archivos de anotaciones y aquellas derivadas directamente de modificaciones realizadas sobre los videos, como los recortes y segmentaciones temporales. En esta sección se presentan las transformaciones asociadas al procesado de los datos propiamente dicho, orientadas a preparar el dataset para su utilización en los modelos de aprendizaje automático.

Esta etapa incluye procesos de filtrado, normalización y estandarización del dataset, con el objetivo de reducir la variabilidad no deseada y mejorar la estabilidad del entrenamiento de los modelos. Cabe destacar que la versión final del dataset procesado fue obtenida de manera iterativa, a partir del análisis de los resultados observados en las primeras etapas de entrenamiento y evaluación de los modelos, lo que permitió ajustar progresivamente las transformaciones aplicadas.

Adicionalmente, se llevó a cabo una instancia de intercambio con el equipo encargado de la anotación de los videos, con el objetivo de evacuar dudas relacionadas con la notación y el significado de un subconjunto de glosas anotadas. Esta reunión permitió comprender con mayor profundidad los criterios utilizados durante el proceso de glosado y resultó fundamental para definir transformaciones que no degradaran la calidad semántica de los datos, evitando modificaciones que pudieran introducir ambigüedades o pérdidas de información relevantes para el dominio lingüístico.

3.4.1. Dataset antes del procesado

Luego de las transformaciones de seccionado y recorte descritas anteriormente, el dataset quedó conformado por 34 videos junto con sus respectivos archivos de anotación. En total, estos archivos contenían 28.093 señas correspondientes a 4.612 glosas únicas. Al realizar un análisis exploratorio inicial sobre

este conjunto de datos, se identificaron una serie de problemáticas relevantes que afectaban directamente su idoneidad para el entrenamiento de modelos de aprendizaje automático. Dichos problemas se describen a continuación.

Problemas con el glosado

Al momento de inicio de este proyecto, los archivos de glosado se encontraban en una etapa temprana de desarrollo, conformando una versión inicial. Como consecuencia, la notación utilizada no estaba completamente estandarizada ni formalizada, lo que introduce inconsistencias que impactan negativamente en la estabilidad, eficiencia y capacidad de generalización de los modelos entrenados sobre estos datos.

Entre las principales inconsistencias y ambigüedades detectadas se encuentran las siguientes:

- **Inconsistencia en los géneros:** En términos generales, las señas no poseen género gramatical. Sin embargo, en el dataset una misma seña puede aparecer glosada utilizando su forma masculina, femenina o variantes que indican neutralidad mediante el uso de símbolos como @ o X.

Por ejemplo, las glosas {MAESTRA, MAESTRO, MAESTRX, MAESTR@} corresponden a una misma seña.

- **Espacios adyacentes a símbolos:** Existe una proporción considerable de glosas que contienen símbolos como '-' o ':'. Se observó que una misma seña puede presentarse, en distintas ocasiones, con o sin espacios adyacentes a estos símbolos, generando múltiples representaciones textuales para una única seña.

Por ejemplo, las glosas {PT:DET, PT :DET, PT: DET} refieren a la misma seña.

- **Uso de signos de interrogación para indicar incertidumbre:** Ante situaciones de duda durante el proceso de anotación, el equipo de glosado utilizó signos de interrogación. Estos podían acompañar a una glosa tentativa (por ejemplo, 'Gato?'), indicando falta de certeza, o bien constituir por sí solos el glosado de una seña cuando no era posible determinar su clasificación (por ejemplo, '?').

Algunos otros ejemplos de glosas que contienen signos '?' o '¿' son: {¿ENFRENTAR?, MIL-SEIS^CERO^CERO¿?, ¿?, ?}.

Distribución de las etiquetas

Otro aspecto crítico que vuelve al dataset poco adecuado para el entrenamiento de modelos de clasificación es la distribución de las etiquetas. La frecuencia de aparición de las glosas presenta un marcado desbalance, con una gran cantidad de clases con muy pocas instancias y un reducido número de señas extremadamente frecuentes.

De las 4.612 glosas únicas identificadas, únicamente 505 cuentan con al menos 10 apariciones en todo el dataset, mientras que solo 911 alcanzan un mínimo de 5 instancias. Adicionalmente, existen 2.519 señas que aparecen una única vez. Esta situación representa un problema significativo, ya que resulta prácticamente imposible que un modelo aprenda a reconocer una seña para la cual dispone de un solo ejemplo.

Por otro lado, las glosas más frecuentes también introducen un desbalance. Las 10 señas con mayor número de apariciones superan las 250 instancias cada una, siendo las tres más frecuentes **PT:DET** con 781 apariciones, **TAMBIÉN** con 491 y **NO** con 417. Considerando que el promedio de instancias por seña no supera las 6, estas diferencias generan un desbalance severo que afecta negativamente el proceso de entrenamiento y la capacidad de generalización de los modelos.

3.4.2. Transformaciones aplicadas

Con el propósito de mitigar las problemáticas identificadas en la sección anterior, se implementaron una serie de transformaciones automatizadas sobre el dataset de glosas. Estas transformaciones se llevaron a cabo mediante scripts de procesamiento desarrollados en Python, los cuales aplican reglas de normalización y filtrado sistemático para estandarizar la notación y reducir la variabilidad no deseada.

Normalización de la notación de género

Una de las primeras transformaciones aplicadas consistió en la estandarización de las formas de género en las glosas. Dado que las señas en Lengua de Señas Uruguaya no poseen género gramatical inherente, pero el proceso de anotación introdujo variaciones con formas masculinas, femeninas y neutrales, se implementó un mecanismo de unificación.

El proceso comienza con una primera pasada sobre todos los archivos CSV del dataset, durante la cual se construye un diccionario global que identifica las palabras neutrales marcadas con el símbolo @ o también con la letra X. Para cada palabra neutral identificada (ejemplo: **MAESTR@** o **MAESTRX**), se verifica la existencia de sus formas masculina (**MAESTRO**) y femenina (**MAESTRA**) en el vocabulario completo. Posteriormente, en una segunda pasada, se reemplazan sistemáticamente las formas específicas de género por su equivalente neutral, garantizando una representación unificada.

Estandarización de mayúsculas y eliminación de caracteres no deseados

Se aplicó una normalización consistente de mayúsculas, convirtiendo todas las glosas a formato completamente en mayúsculas para eliminar variaciones derivadas de diferencias en capitalización. Paralelamente, se eliminaron filas que no contenían glosas válidas, incluyendo aquellas que presentaban únicamente signos de interrogación o incertidumbre (**¿**, **?**), los cuales habían sido utilizados

durante el proceso de anotación para indicar duda o imposibilidad de clasificación.

Reemplazo de patrones específicos

Se implementaron reglas de reemplazo para patrones específicos identificados en el análisis exploratorio:

- **Deletreo:** Las secuencias de deletreo (finger spelling), identificadas por el patrón **FS:**, fueron reemplazadas por el token especial **<DELETREO>**, permitiendo agrupar todas las instancias de deletreo bajo una única representación. Esta decisión se fundamenta en la alta variabilidad de este tipo de señas, que no será abordada en este trabajo. Sin embargo, la utilización de un token específico permite detectar su ocurrencia dentro de las secuencias, facilitando su tratamiento en futuras investigaciones.
- **Espacios adyacentes a símbolos:** Se eliminaron los espacios innecesarios alrededor de símbolos como **:** y **-**, estandarizando representaciones como **PT:DET**, **PT :DET** y **PT: DET** a una forma única.

Filtrado basado en vocabulario

Finalmente, se aplicó un proceso de filtrado basado en la frecuencia de las glosas. A partir del listado de señas ordenadas por cantidad de apariciones, se definió un punto de corte con el objetivo de evitar concentrar el aprendizaje en aquellas con muy pocos ejemplos, reteniendo únicamente las glosas con suficiente representación en el *dataset*. El valor seleccionado para esta frecuencia mínima fue 10 apariciones. Esta transformación resulta crítica para reducir el desbalance de clases y mejorar la estabilidad del entrenamiento, eliminando glosas con muy baja frecuencia que no aportarían valor predictivo significativo a los modelos.

Eliminación de videos con baja calidad

Durante las primeras etapas de entrenamiento y evaluación de los modelos, se identificó que dos videos correspondientes a recortes del mismo programa de noticias original presentaban una calidad de video extremadamente deficiente. En particular, ambos videos tenían una resolución de 320×240 píxeles y, dado que el área donde aparece el intérprete constituye solo una pequeña porción del cuadro original, el recorte utilizado para el entrenamiento resultaba en imágenes de muy baja resolución. Esta degradación en la calidad visual impactaba negativamente en el rendimiento de los modelos de reconocimiento de señas.

Ante esta situación, se tomó la decisión de excluir dichos videos del dataset, lo que implicó una reducción en el número total de ejemplos disponibles. Sin embargo, esta medida resultó beneficiosa al mejorar significativamente la calidad general del conjunto de datos y, consecuentemente, las métricas de desempeño de los modelos entrenados.

Estas transformaciones se aplicaron de manera secuencial y reproducible, generando versiones intermedias del dataset que permitieron evaluar iterativamente el impacto de cada modificación sobre la calidad de los datos y el rendimiento de los modelos de aprendizaje automático.

3.4.3. Estadísticas del dataset LSU-446

Luego de la aplicación de las transformaciones descritas en la sección anterior, el dataset experimentó una reducción significativa en su tamaño, lo cual redundó en una mejora sustancial de su calidad. El conjunto final de datos quedó conformado por 17.793 señas clasificadas entre 446 glosas únicas. Esto representa una retención del 63,3 % de las instancias de señas (17.793 de 28.093), concentradas en el 9,7 % de las glosas originales (446 de 4.612).

Esta disminución drástica en la cardinalidad del vocabulario implica una simplificación considerable del problema de clasificación, permitiendo concentrar los esfuerzos de aprendizaje en aquellas señas para las cuales existe una cantidad suficiente de ejemplos de entrenamiento.

La Tabla 3.1 presenta una comparación detallada de las estadísticas principales del dataset antes y después de las transformaciones aplicadas. El dataset procesado se referirá en adelante como LSU-446. La denominación proviene del hecho de que el dataset cuenta con 446 clases resultantes luego del filtrado por mínimo de ocurrencias.

Tabla 3.1: Comparación de estadísticas del dataset antes y después de las transformaciones (LSU-446)

Métrica	Dataset original	Dataset procesado (LSU-446)
Número total de señas	28.093	17.793
Número de glosas únicas	4.612	446
Promedio de instancias por glosa	6.1	39.2
Desviación estándar de instancias por glosa	21.4	57.25
Número de glosas con al menos 10 apariciones	505	446
Porcentaje de glosas con al menos 10 instancias	10,9 %	100 %

Como se observa en la Tabla 3.1, la reducción del 90,3 % en el vocabulario no solo simplificó el problema de clasificación, sino que también mejoró significativamente la distribución de las instancias. El promedio de ejemplos por glosa aumentó de 6.1 a 39.2, lo que proporciona una base más sólida para el entrenamiento de modelos de aprendizaje automático. Adicionalmente, se observa que en el dataset procesado (LSU-446) todas las glosas cumplen la condición de al menos 10 instancias, ya que el filtrado por frecuencia mínima se aplicó explícitamente durante la construcción del vocabulario. La diferencia entre las 505 glosas frecuentes del *dataset* original y las 446 finales se explica por el proceso de normalización, que unificó variantes ortográficas y formas alternativas de una misma glosa.

En cuanto a la duración de las señas en el *dataset*, la duración promedio es de 403 milisegundos. Este valor es consistente con el hecho de que los videos corresponden a noticias o relatos narrados a velocidad natural, donde las transiciones

entre señas son rápidas y la coarticulación es frecuente. Desde el punto de vista del modelado, esto implica que cada seña queda representada por un número relativamente reducido de *frames*, lo que aumenta la sensibilidad a variaciones temporales y a imprecisiones en las anotaciones de inicio y fin. Como se discute más adelante, esta característica diferencia al dataset LSU de otros datasets de referencia como WLASL y LSA64, que fueron capturados bajo condiciones más controladas y con señas completamente aisladas.

Una observación importante es el aumento de la desviación estándar de instancias por glosa en el *dataset* LSU-446 respecto al original. Esto puede explicarse porque en el *dataset* crudo la gran mayoría de las glosas tiene muy pocas instancias (menos de 10). Al aplicar el filtrado, se eliminaron gran parte de esas clases minoritarias, lo que resultó en una distribución relativamente más dispersa entre las glosas restantes.

En este contexto y a modo ilustrativo, la Figura 3.2 presenta un ejemplo de una secuencia del *dataset* LSU-446, donde se muestran *frames* representativos junto con la alineación temporal de las glosas correspondientes. Esta visualización permite apreciar de manera concreta la naturaleza secuencial y continua de los datos. En particular, se observa cómo cada glosa ocupa un intervalo específico de tiempo y cómo la transición entre señas ocurre de forma fluida, sin pausas. Este tipo de estructura refuerza la necesidad de contar con una cantidad suficiente de instancias por glosa para que los modelos puedan capturar adecuadamente tanto la variabilidad *intra-clase* como las dinámicas temporales presentes en el dataset.

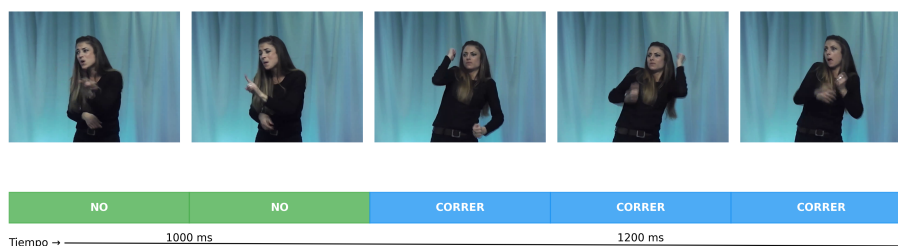


Figura 3.2: Ejemplo de una secuencia del dataset. En la parte superior se muestran *frames* representativos de un video, mientras que en la parte inferior se presenta la alineación temporal de las glosas correspondientes. Se observa la transición entre las señas “NO” y “CORRER”.

Capítulo 4

Reconocimiento aislado: entrenamiento y experimentación

En este capítulo se abordan los dos modelos de reconocimiento aislado de señas estudiados en este trabajo. Para cada uno de ellos se describen las pruebas y adaptaciones realizadas, así como los resultados obtenidos a lo largo de los distintos experimentos llevados a cabo.

4.1. SPOTER

En esta sección se describen las principales modificaciones implementadas, así como los experimentos realizados durante el entrenamiento y la evaluación del modelo SPOTER, utilizando el dataset de Lengua de Señas Uruguay (LSU).

4.1.1. Formato de entrada de SPOTER

El modelo SPOTER utiliza como entrada un archivo CSV en el que cada columna representa una coordenada de un *keypoint* específico a lo largo de una seña, junto con una columna final numérica que codifica el *ground truth* correspondiente a la glosa de la seña, representada como una clase. Cada fila del archivo corresponde a una instancia completa de seña, entendida como una secuencia temporal de *frames*. Para cada columna de coordenadas, el valor almacenado consiste en una lista que describe la evolución temporal de dicha coordenada a lo largo de la seña.

Dado que este formato de entrada difiere de manera significativa de la estructura original del dataset LSU-446, fue necesario implementar un script de transformación que permitiera adaptar los datos al formato requerido por SPOTER.

Como se describió en el Capítulo 3, el *dataset* LSU-446 se compone de archivos CSV que contienen las glosas, junto con archivos JSON que almacenan la codificación de *keypoints* obtenida mediante MediaPipe para cada *frame* de cada video. Para entrenar el modelo SPOTER con estos datos, fue necesario transformar dicha información al formato específico de entrada definido por el modelo.

Preprocesamiento implementado

En primer lugar, se realiza un mapeo entre los *keypoints* obtenidos a través de MediaPipe y las columnas definidas para los *keypoints* de entrada de SPOTER, dado que ambos enfoques utilizan convenciones de nomenclatura distintas. Asimismo, se omite la columna correspondiente a *handedness* extraída por MediaPipe, la cual provee, para cada mano detectada, una probabilidad asociada a su clasificación como izquierda o derecha. Esta información no es utilizada por el modelo SPOTER.

Por otra parte, dado que las glosas en los archivos CSV del dataset LSU están definidas temporalmente en milisegundos y no en número de *frames*, se utilizan los campos *TIEMPO_INICIAL* y *TIEMPO_FINAL* para extraer los *frames* exactos que componen cada señal. Adicionalmente, se emplea un archivo JSON de vocabulario, generado durante la etapa de preprocesamiento, que contiene la correspondencia entre glosas y sus respectivos identificadores de clase. Dichos identificadores son los que se utilizan para representar la glosa en la columna final del archivo CSV de entrada.

Finalmente, cada señal se transforma en una única fila del archivo resultante, iterando sobre los *frames* que la componen y organizando los datos en las listas de coordenadas requeridas por SPOTER.

4.1.2. Configuración experimental y criterios de evaluación

En esta sección se describe el protocolo experimental adoptado para el entrenamiento y la evaluación del modelo SPOTER sobre el dataset LSU, incluyendo la estrategia de partición del dataset, el entorno de ejecución y las métricas utilizadas para evaluar su desempeño.

Partición del *dataset*

Para entrenar y validar el rendimiento del modelo SPOTER sobre el conjunto de datos LSU, fue necesario dividir el dataset en subconjuntos de entrenamiento, validación y prueba (*train*, *val* y *test*, respectivamente). Para ello se adoptó una partición 80/10/10, asignando el 80 % de las señales al conjunto de entrenamiento, y el 10 % a cada uno de los conjuntos de validación y prueba. El conjunto de entrenamiento se utiliza para el ajuste de los parámetros del modelo, mientras que el conjunto de validación se emplea durante el proceso de entrenamiento, entre épocas, para monitorear el desempeño y, de ser necesario, aplicar técnicas

como *early stopping*. Por su parte, el conjunto de prueba se reserva exclusivamente para la evaluación final, permitiendo comparar de manera objetiva el rendimiento de los distintos experimentos realizados.

Como se mencionó en el Capítulo 3, inicialmente se optó por realizar la partición a nivel de secuencias contiguas de video, bajo la hipótesis de que la cantidad de datos y la distribución de las glosas permitirían una adecuada generalización. En este enfoque, se analizó el número total de señas contenidas en cada segmento contiguo de video y se realizó la división respetando la proporción 80/10/10.

Sin embargo, tras realizar varias pruebas utilizando este método y analizar los resultados obtenidos, se observó que el desempeño del modelo mejoraba de forma considerable al adoptar una partición convencional, en la que las señas se separan de manera independiente de su video de origen. Este enfoque permitió obtener una distribución más uniforme de las glosas entre los conjuntos de entrenamiento, validación y prueba, reduciendo el sesgo introducido por la concentración de ciertas clases en secuencias específicas de video, lo que se tradujo en una mejora consistente del desempeño del modelo.

Toda esta lógica de partición se encuentra integrada en el mismo código que realiza la transformación desde el dataset LSU, por lo que el resultado final utilizado para el entrenamiento y la evaluación del modelo consiste en tres archivos CSV independientes, correspondientes a los conjuntos de entrenamiento, validación y prueba.

Entorno y configuración de entrenamiento

Las distintas pruebas de entrenamiento del modelo fueron realizadas en el entorno de cómputo de alto rendimiento de la Facultad (ClusterUY). Una vez configurado el entorno de ejecución, incluyendo la instalación de Conda y las dependencias necesarias, se ejecutaron los entrenamientos utilizando las GPU disponibles del servidor.

A lo largo de todos los experimentos se utilizó un *learning rate* fijo de 0,001, en concordancia con el valor empleado en el trabajo original de SPOTER, con el objetivo de mantener la comparabilidad de los resultados.

Top-5 Accuracy y Balanced Accuracy

Dado el bajo desempeño inicial observado en términos de *accuracy* estándar, se utilizó también el cálculo de la métrica *top-5 accuracy*. Esta métrica considera una predicción como correcta si la clase verdadera se encuentra entre las cinco clases con mayor probabilidad asignada por el modelo, a diferencia de la *accuracy* convencional, que solo evalúa la predicción principal. Si bien la inclusión de esta métrica no resolvió por sí sola las limitaciones observadas en las métricas iniciales, permitió contar con un marco de referencia más informativo para la comparación entre experimentos. En particular, dado que las métricas más estrictas tendían a arrojar valores cercanos a cero, resultaba difícil distinguir diferencias de desempeño. La utilización de una métrica más flexible, como *top-*

5, permitió evidenciar variaciones entre los modelos y facilitar su comparación bajo un criterio más discriminativo.

Adicionalmente, se utilizó el cálculo de la *accuracy* ponderada por clase, también conocida como *balanced accuracy* o *macro-average*. Esta métrica permitió evaluar el rendimiento del modelo de forma menos sensible a las diferencias de frecuencia entre clases en los conjuntos de validación y prueba. Asimismo, se registró la versión *top-5* de esta métrica, que considera como acierto los casos en los que la clase correcta se encuentra entre las cinco predicciones más probables del modelo. En un contexto como el del dataset LSU, caracterizado por un marcado desbalance entre clases, estas métricas resultaron fundamentales para interpretar adecuadamente el desempeño del modelo.

Matriz de confusión

Con el fin de comprender en mayor profundidad las causas de las bajas métricas de precisión observadas en los primeros experimentos, se consideró necesario contar con una representación visual de los aciertos y errores del modelo entrenado. Para ello, se incorporó al ciclo de entrenamiento y validación la generación de matrices de confusión para cada *checkpoint*.

Este análisis permitió no solo visualizar la evolución del desempeño del modelo a lo largo de las épocas, sino también identificar problemas relacionados con el desbalance entre clases presentes en el dataset. Un ejemplo representativo fue el caso de la clase más frecuente del conjunto de datos, PT:DET, que en las primeras pruebas del modelo era predicha con una frecuencia significativamente mayor que el resto de las clases. Esta observación motivó la experimentación con técnicas de balanceo del dataset previas al entrenamiento del modelo.

4.1.3. Adaptaciones implementadas en el entrenamiento

En esta sección se describen las principales problemáticas identificadas, así como las modificaciones implementadas en el proceso de entrenamiento y evaluación, con el objetivo de mejorar el rendimiento del modelo sobre el conjunto de datos.

Como se mencionó anteriormente, las primeras pruebas realizadas con el modelo arrojaron métricas de precisión muy bajas, alcanzando aproximadamente un 3 % de *accuracy* en el conjunto de validación. A partir de estos resultados, se implementaron distintos mecanismos orientados a comprender y mitigar la problemática subyacente, lo que permitió incrementar de forma iterativa el desempeño del modelo. Cabe destacar que, una vez modificado el método de partición del dataset, se observó un aumento inmediato de aproximadamente un 10 % en la precisión con respecto a la mejor métrica obtenida previamente.

Variación en la cantidad de clases a clasificar

En los primeros experimentos, el modelo fue entrenado para clasificar la totalidad de las glosas que conformaban el vocabulario final, es decir, aquellas

que presentaban más de 10 ocurrencias en el *dataset*. Esto implicó entrenar un clasificador sobre 446 clases distintas.

Con el objetivo de obtener métricas más comparables con las reportadas en el trabajo original de SPOTER —donde el modelo es evaluado sobre los datasets WLASL100 y LSA64— se decidió experimentar reduciendo el número de clases a identificar. En particular, se entrenaron modelos considerando únicamente las 100 señas más frecuentes (con aproximadamente 10,000 muestras) y las 64 señas más frecuentes (con cerca de 8,800 muestras).

Early Stopping

A lo largo de los distintos experimentos se observó que, en la mayoría de los casos, el mejor valor de *accuracy* en el conjunto de validación no se alcanzaba en las últimas épocas de entrenamiento. Este comportamiento motivó la incorporación de la técnica de *Early Stopping*, que permite detener el entrenamiento cuando el desempeño de validación deja de mejorar.

La adopción de esta estrategia permitió reducir de forma significativa el tiempo de entrenamiento y el costo computacional asociado, evitando la ejecución de épocas adicionales que no contribuían a mejorar el rendimiento del modelo.

Augmentation y pruning

Si bien en el *pipeline* original de preprocesamiento se logró mejorar de forma significativa la distribución de las señas del dataset al conservar únicamente aquellas glosas con más de 10 ocurrencias, las matrices de confusión obtenidas en las primeras pruebas sugirieron que aún podían existir dificultades de generalización asociadas al desbalance entre clases. En particular, se observó una tendencia del modelo a favorecer las clases más frecuentes. Por este motivo, se implementaron técnicas de *augmentation* (aumentado de datos) (Shorten y Khoshgftaar, 2019) y *pruning* (eliminación de instancias) en el script encargado de la conversión de datos al formato de entrada del modelo y de la partición en conjuntos de entrenamiento y validación.

Augmentation Como puede observarse en la Figura 4.1, más de la mitad de las glosas presentes en el dataset cuentan con menos de 20 instancias, y menos de un 10 % posee más de 100 ejemplos. Con el objetivo de reducir este desbalance y favorecer un entrenamiento más estable, se desarrolló una implementación propia de aumentado de datos aplicado exclusivamente al conjunto de entrenamiento, previo a la construcción final del dataset utilizado por el modelo.

La estrategia adoptada consiste en generar nuevas instancias sintéticas a partir de ejemplos reales pertenecientes a las clases menos representadas, hasta alcanzar un umbral mínimo de ocurrencias por glosa. Para cada clase cuya cantidad de instancias es inferior a dicho umbral, se seleccionan aleatoriamente ejemplos originales y se generan copias aumentadas. Con el objetivo de reducir la dependencia del modelo de configuraciones puntuales de las instancias repli-

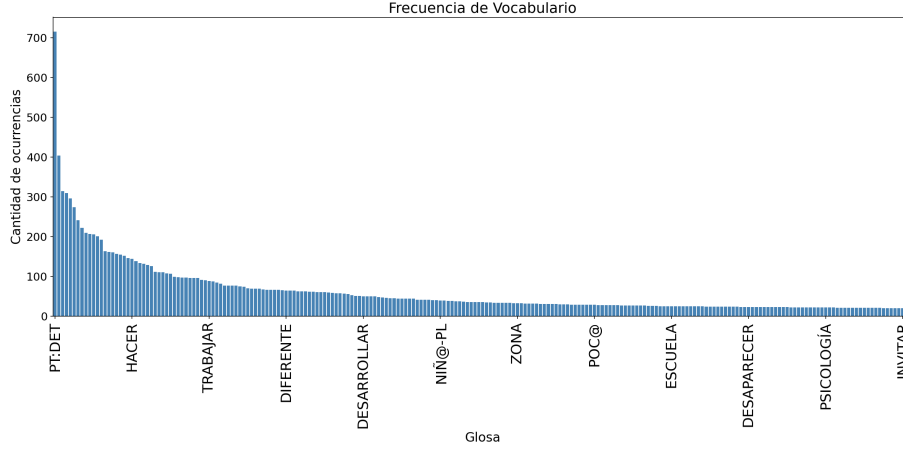


Figura 4.1: Distribución de ocurrencias de las glosas en el dataset final previo al proceso de *augmentation*. En el eje x se muestra una glosa cada 20.

cadass y favorecer una mayor robustez durante el entrenamiento, se añade ruido gaussiano a las coordenadas de cada *keypoint* de las réplicas generadas.

Cabe señalar que la perturbación introducida no busca modelar variaciones entre distintos realizadores de una seña, sino simular pequeñas imprecisiones propias del proceso de detección de *keypoints*. En este sentido, el ruido se aplica de forma independiente a cada *keypoint*, evitando introducir desplazamientos sistemáticos o transformaciones rígidas a lo largo de toda la seña, que podrían alterar artificialmente su dinámica temporal. Este enfoque permite incrementar la variabilidad local de los datos sin modificar la estructura temporal global ni la semántica del gesto.

El proceso de *augmentation* desarrollado se controla mediante tres parámetros configurables en el script:

- **Augment**: indica si se aplica o no el aumentado de instancias.
- **AugmentThreshold**: define el número mínimo de ocurrencias que debe tener cada glosa luego del proceso de aumentado.
- **AugmentNoise**: especifica la desviación estándar σ del ruido gaussiano aplicado a las coordenadas. Internamente, este parámetro es pasado como argumento a `numpy.random.normal(0,  $\sigma$ )` sobre los arrays de coordenadas de cada *keypoint*. En este trabajo se evaluaron los valores $\sigma \in \{0.001, 0.01\}$.

Sea $\mathbf{p}_t = (x_t, y_t)$ la coordenada bidimensional de un *keypoint* en el instante t . La transformación aplicada durante el proceso de *augmentation* puede expresarse como:

$$\tilde{\mathbf{p}}_t = (x_t + \varepsilon_x, y_t + \varepsilon_y), \quad \text{con } \varepsilon_x, \varepsilon_y \sim \mathcal{N}(0, \sigma^2),$$

donde σ corresponde a la desviación estándar configurada mediante **AugmentNoise**. La perturbación se aplica de forma independiente a cada coordenada y a cada *frame*. Las coordenadas de los *keypoints* están normalizadas en $[0, 1]$ por el *pipeline* de SPOTER; por lo tanto, $\sigma = 0,01$ implica una perturbación cuya desviación estándar es el 1 % del rango total de coordenadas, y $\sigma = 0,001$ implica el 0,1 %. Tras añadir el ruido, los valores resultantes se truncan al intervalo $[0, 1]$ mediante una operación de recorte (*clamping*): $\tilde{p} = \text{clip}(\tilde{p}, 0, 1)$, sin ninguna transformación lineal adicional.

Ambos valores de σ constituyen perturbaciones geométricas pequeñas: una desviación estándar del 1 % del rango de coordenadas es inferior a la variabilidad natural entre instancias de una misma seña y comparable a las imprecisiones propias del estimador de *keypoints*. Si bien esta técnica no preserva estrictamente las relaciones geométricas entre los *keypoints* de una misma parte del cuerpo, los valores utilizados se consideran lo suficientemente pequeños para no introducir deformaciones que alteren la semántica de las señas.

Cabe destacar que la *augmentation* se aplica únicamente sobre el conjunto de entrenamiento, preservando los conjuntos de validación y prueba como muestras representativas del dataset original, sin alteraciones sintéticas.

Pruning En paralelo al aumento de la cota inferior de ocurrencias por clase mediante *augmentation*, se incorporó un mecanismo de *pruning* con el objetivo de limitar la cantidad máxima de instancias de las glosas más frecuentes del dataset. Esta técnica consiste en acotar, de forma aleatoria y controlada, el número de ejemplos por clase a un valor máximo predefinido, aplicado también exclusivamente al conjunto de entrenamiento.

El objetivo principal del *pruning* es reducir el impacto desproporcionado de las clases dominantes durante el proceso de optimización, evitando que el modelo sesgue su aprendizaje hacia dichas clases y favoreciendo una distribución más equilibrada de gradientes durante el entrenamiento. En conjunto, las técnicas de *augmentation* y *pruning* permiten controlar tanto la cota inferior como la cota superior de ocurrencias por clase, contribuyendo a una mejor capacidad de generalización del modelo.

4.1.4. Resultados y análisis experimental

En esta sección se presentan y analizan los resultados obtenidos a partir de las distintas pruebas realizadas sobre el modelo SPOTER. El objetivo no es únicamente reportar las métricas finales, sino también documentar el proceso de experimentación que condujo a las decisiones adoptadas, identificando problemáticas, hipótesis evaluadas y sus respectivos impactos en el desempeño del modelo.

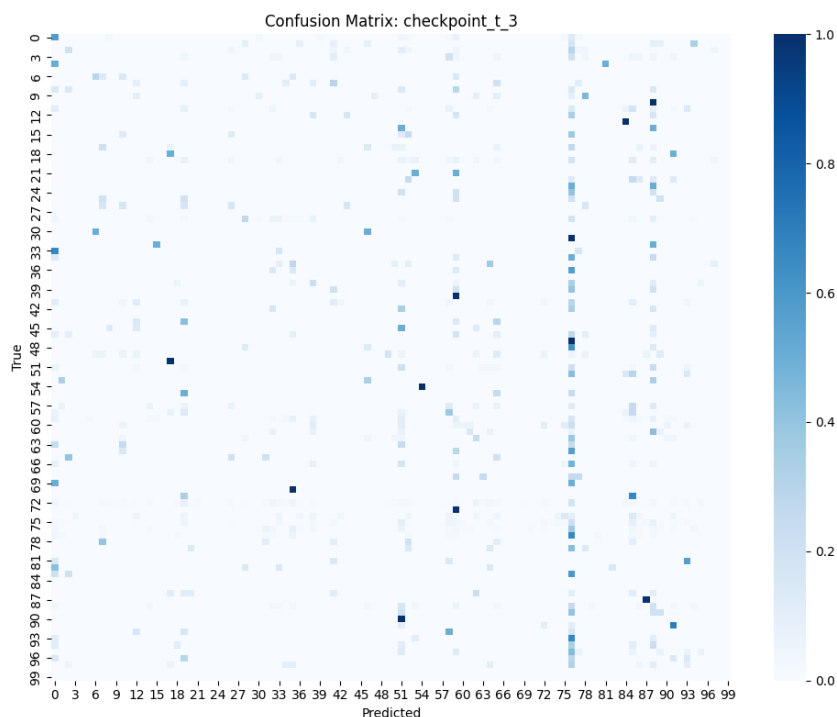


Figura 4.2: Matriz de confusión normalizada correspondiente a uno de los primeros entrenamientos realizados sobre un subconjunto de 100 clases. Se observa una tendencia marcada del modelo a concentrar las predicciones en la clase de ID 76 (PT:DET), evidenciando el impacto del desbalance entre clases en las etapas iniciales de experimentación.

Desempeño inicial y problemas detectados

El análisis de las matrices de confusión reveló una fuerte concentración de predicciones en un subconjunto reducido de clases, particularmente aquellas con mayor número de ocurrencias en el conjunto de entrenamiento.

La Figura 4.2 ilustra este fenómeno para uno de los primeros entrenamientos realizados, donde puede observarse una marcada tendencia del modelo a predecir la clase PT:DET, en detrimento del resto del vocabulario. Este comportamiento motivó la exploración sistemática de distintas estrategias orientadas a mitigar el impacto del desbalance entre clases y a mejorar la capacidad discriminativa del modelo.

Impacto del tamaño del vocabulario

Una de las primeras hipótesis evaluadas fue que la elevada cantidad de clases a clasificar constituía un factor limitante para el aprendizaje efectivo del modelo.

En consecuencia, se realizaron experimentos reduciendo el tamaño del vocabulario a subconjuntos más acotados, considerando únicamente las 100 y las 64 glosas más frecuentes del dataset. La elección de estos tamaños se fundamenta en la posibilidad de contrastar el desempeño del modelo con los benchmarks de referencia sobre los datasets WLASL100 y LSA64 (explicados en detalle en el Anexo 6.4.4).

Los resultados obtenidos se pueden visualizar en la Tabla 4.1, donde se observa una mejora consistente al reducir el número de clases respecto al escenario completo de 446 glosas. En particular, el modelo entrenado con 100 clases presenta un desempeño superior al entrenado sobre el vocabulario completo, tanto en términos de *accuracy top-1* como *top-5*. No obstante, al reducir el vocabulario de 100 a 64 clases no se observa una mejora adicional significativa en las métricas evaluadas, lo que podría indicar que, a partir de cierto punto, la simple reducción del espacio de clasificación deja de traducirse en ganancias de desempeño.

Este comportamiento sugiere que, si bien disminuir la cantidad de clases facilita el aprendizaje del modelo, otros factores comienzan a tener un peso comparable o dominante en el rendimiento alcanzable. En particular, es razonable conjeturar que la variabilidad intra-clase presente en los datos y la capacidad del modelo para modelarla adecuadamente influyen en este resultado. En este sentido, aunque el subconjunto de 64 glosas concentra un mayor número de ocurrencias, esta reducción adicional del vocabulario podría estar limitando la diversidad de patrones dinámicos representados en los *keypoints*, por lo que no se refleja en mejoras observables bajo las configuraciones experimentales consideradas.

Tabla 4.1: Comparación del desempeño del modelo SPOTER en función del tamaño del vocabulario. Se evalúan tres configuraciones del dataset LSU: LSU-446 (vocabulario completo filtrado), LSU-100 y LSU-64, correspondientes a reducciones progresivas a las 100 y 64 glosas más frecuentes, respectivamente.

<i>Dataset</i>	<i>Top-1</i>	<i>Top-5</i>	<i>Balanced Top-1</i>	<i>Balanced Top-5</i>
LSU-446	0.241	0.427	0.207	0.370
LSU-100	0.320	0.531	0.305	0.503
LSU-64	0.295	0.523	0.283	0.485

Augmentation y pruning

Con el objetivo de mitigar los efectos del desbalance entre clases, se evaluó el impacto de distintas configuraciones de *augmentation* (Shorten y Khoshgof-taar, 2019) y *pruning* aplicadas exclusivamente al conjunto de entrenamiento, las cuales son técnicas ampliamente utilizadas en aprendizaje profundo. Se consideraron tanto esquemas de aumentado agresivo como configuraciones más moderadas, así como escenarios sin aplicación de estas técnicas.

Asimismo, fue evaluado el efecto de incorporar ruido gaussiano a los *keypoints*

generados durante el aumentado de datos. Para cada régimen de cotas mínimas y máximas por clase se consideraron variantes sin ruido y con los dos niveles de perturbación descritos en la Sección 4.1.3 ($\sigma \in \{0.001, 0.01\}$), con el fin de analizar si pequeñas deformaciones geométricas contribuyen a mejorar la capacidad de generalización del modelo.

La Tabla 4.2 resume un subconjunto representativo de estos experimentos, todos realizados sobre LSU-446, seleccionado para cubrir distintos regímenes de intervención: desde ausencia de *augmentation* y *pruning* (E1), pasando por configuraciones suaves con ampliación moderada del número de muestras por clase (E2–E4), hasta esquemas más restrictivos y agresivos (E5–E7). De este modo, se busca mostrar el comportamiento del modelo a lo largo de un espectro de intensidades de balanceo de datos, en lugar de reportar únicamente un único punto de operación.

Los resultados obtenidos indican que el efecto de estas estrategias no es uniforme. En ciertos regímenes, particularmente en configuraciones intermedias, se observa una mejora en métricas balanceadas y una reducción de la dominancia de las clases más frecuentes. Sin embargo, en otros casos, el impacto sobre la *accuracy* global es limitado o incluso nulo.

No obstante, al analizar en detalle los resultados de la Tabla 4.2, se observa que las mejoras introducidas por las distintas configuraciones de *augmentation* son, en general, marginales. Si bien ciertos experimentos puntuales logran incrementos leves en métricas específicas (por ejemplo, Top-1 o Top-5), estos no son consistentes entre configuraciones ni se traducen en mejoras simultáneas en todas las métricas consideradas. En algunos casos, incluso, el uso de *augmentation* con mayor intensidad conduce a una degradación del rendimiento. Esto sugiere que el impacto del *augmentation* es acotado y altamente dependiente de su parametrización, sin evidenciar ganancias sustanciales o sistemáticas respecto al baseline (E1).

Estos hallazgos sugieren que, si bien las técnicas de *augmentation* y *pruning* pueden contribuir a un entrenamiento más equilibrado, su efectividad depende fuertemente de la configuración específica y del tamaño del vocabulario considerado, y no garantizan mejoras sistemáticas en todos los escenarios.

Tabla 4.2: Comparación de configuraciones de *augmentation* y *pruning* sobre la partición de prueba.

ID	Aug min	σ	Aug max	Top-1	Top-5	Bal Top-1	Bal Top-5
E1	–	–	–	0.241	0.427	0.207	0.370
E2	40	0.001	300	0.243	0.430	0.225	0.370
E3	40	0.01	300	0.241	0.431	0.224	0.379
E4	40	0	300	0.232	0.418	0.238	0.397
E5	100	0.001	100	0.218	0.374	0.231	0.354
E6	100	0.01	100	0.208	0.373	0.227	0.349
E7	100	0	100	0.202	0.355	0.229	0.340

Cambios arquitectónicos menores y clase *no-seña*

Como se mencionó en la Sección 3.4.3, la duración promedio de las señas del dataset LSU es de 403 milisegundos. En contraste, en el trabajo en el que se introduce el dataset WLASL se reporta una duración promedio de 2.41 segundos por seña (Li, Opazo, Yu, y Li, 2020). Ante esta diferencia significativa en la duración de las señas, y por lo tanto en la cantidad de *keypoints* que componen cada instancia, se decidió experimentar con el tamaño de la capa inicial de *embedding* del modelo, reduciendo su dimensión de 50 a 20. Estas variantes no mostraron diferencias consistentes en las métricas globales de clasificación aislada, por lo que no se adoptó una configuración alternativa sobre la base de evidencia cuantitativa concluyente.

En paralelo, se evaluó la incorporación explícita de una clase adicional correspondiente a *no-seña*. Esta extensión no se introduce con el objetivo de mejorar la clasificación aislada, sino como un requisito funcional que se utilizará en la siguiente etapa de reconocimiento continuo mediante ventana deslizante. En ese escenario, el modelo se aplica de forma densa sobre el video completo y debe discriminar explícitamente entre *frames* con actividad lingüística y *frames* de transición o reposo. La presencia de una clase de fondo permite modelar esta distinción dentro del propio espacio de etiquetas, evitando forzar asignaciones léxicas sobre segmentos sin seña.

La Tabla 4.3 muestra el impacto de esta decisión en modelos sin aumentado. Sobre LSU-446, la inclusión de *no-seña* produce una variación marginal en *accuracy* Top-1 y Top-5, junto con una leve caída en las métricas balanceadas. Sobre LSU-100, el efecto es más marcado, debido a que la nueva clase compite con un vocabulario significativamente más reducido y altera con mayor peso la distribución de probabilidad de salida.

El análisis por clase (Tabla 4.4) indica que la categoría *no-seña* presenta baja precisión y *recall* en evaluación aislada, lo que implica que el modelo tiende a confundir segmentos sin seña con glosas válidas. Este comportamiento es consistente con la alta variabilidad visual de los intervalos sin seña y con la ausencia de una estructura gestual estable comparable a la de las clases léxicas.

En consecuencia, aunque la incorporación de la clase *no-seña* no optimiza las métricas globales de reconocimiento aislado, se adopta deliberadamente en las configuraciones utilizadas para reconocimiento continuo, donde su rol como clase de fondo resulta necesario para una interpretación operativa correcta de las predicciones *frame-a-frame*.

Configuración seleccionada y resultados finales

A partir del análisis conjunto de los experimentos realizados, se seleccionaron dos configuraciones de referencia de SPOTER que representan los mejores compromisos entre desempeño, estabilidad y utilidad para las etapas posteriores del trabajo: (i) el modelo de LSU-100 sin aumentado, que maximiza el desempeño de clasificación en vocabulario reducido, y (ii) el modelo de LSU-446 con aumentado leve, que preserva cobertura léxica amplia con métricas balanceadas

Tabla 4.3: Impacto de la incorporación de la clase *no-seña* en SPOTER, sin aumentado de datos.

<i>Dataset</i>	No-seña	Top-1	Top-5	Bal. Top-1	Bal. Top-5
LSU-446	No	0.241	0.427	0.207	0.370
LSU-446	Sí	0.238	0.427	0.203	0.358
LSU-100	No	0.320	0.531	0.305	0.503
LSU-100	Sí	0.268	0.514	—	—

Tabla 4.4: Métricas por clase para *no-seña* (SPOTER sobre LSU-446 con clase *no-seña*, sin aumentado).

Clase	Precisión	Recall	F_1
no-seña	0.0769	0.0339	0.0471

superiores a la configuración base sin aumentado.

La Tabla 4.5 resume los resultados de ambas configuraciones. El modelo de LSU-100 sin aumentado alcanza los mejores valores absolutos de *accuracy* Top-1 y Top-5, así como de métricas balanceadas, lo que indica una mejora consistente tanto en desempeño global como en equidad entre clases. Este resultado es esperable dado el menor tamaño del vocabulario.

Por su parte, la mejor configuración sobre vocabulario completo corresponde al esquema con aumentado leve (E2 en la Tabla 4.2). Si bien las mejoras en *accuracy* global son moderadas respecto a la versión sin aumentado, se observa una ganancia clara en métricas balanceadas, en particular en Top-1 balanceado, lo que indica una distribución de errores más uniforme entre clases y una menor concentración del desempeño en las clases mayoritarias.

Desde el punto de vista metodológico, ambas configuraciones cumplen roles distintos en el resto del trabajo. El modelo de LSU-100 se utiliza como referencia de techo de desempeño en condiciones controladas, mientras que el modelo de LSU-446 con aumentado leve se adopta como base principal para la integración en la subsiguiente tarea de *sign spotting*, donde la cobertura léxica resulta prioritaria.

Si bien las métricas obtenidas se sitúan por debajo de valores reportados en la literatura sobre otros datasets, los resultados sobre LSU reflejan las particularidades del dominio: fuerte desbalance entre clases, variabilidad en ejecución gestual y diferencias de duración entre señas. En este contexto, las métricas balanceadas adquieren especial relevancia como indicador complementario al *accuracy* global.

Estos resultados constituyen la base sobre la cual se construyen las evaluaciones posteriores en escenarios de reconocimiento continuo presentadas en las secciones siguientes.

Tabla 4.5: Resultados finales de las dos configuraciones SPOTER seleccionadas.

Modelo	Dataset	Aumentado	Top-1	Top-5	Bal. Top-1	Bal. Top-5
SPOTER	LSU-100	Sin aumentado	0.320	0.531	0.305	0.503
SPOTER	LSU-446	E2	0.243	0.430	0.225	0.370

4.2. Implementación MSG3D

En esta sección se detallan los experimentos y las modificaciones implementadas durante el entrenamiento y la evaluación del modelo basado en la arquitectura MSG3D y tomando como punto de partida el trabajo realizado por la *Universitat Pompeu Fabra* sobre el reconocimiento de señas aisladas en LSC, utilizando el *dataset* de LSU.

4.2.1. Formato de dataset

Para la ejecución de los diversos experimentos, se tomó como base el marco de trabajo ya mencionado de LSC en el que además presentan un repositorio de código ([Naiara Garmendia, Horacio Saggion y Euan McGill, 2025b](#)). Dicho repositorio proporciona una solución integral que abarca desde la generación del *dataset* desde cero hasta el entrenamiento y *fine-tuning* de la arquitectura MSG3D.

Estructura de los datos

El formato requerido por la arquitectura para el entrenamiento se compone de dos archivos principales por cada partición (*split*):

1. Archivo de datos (.npz): Consiste en un arreglo de NumPy que almacena las coordenadas de los esqueletos. La estructura del tensor sigue el formato (N, C, T, V, M) , donde:
 - N : Número de muestras (instancias de señas en el split).
 - C : Número de canales, correspondiente a las coordenadas (x, y, c) , donde c representa el puntaje de confianza (*confidence score*).
 - T : Dimensión temporal (número de cuadros por seña).
 - V : Número de articulaciones o puntos clave (*keypoints*).
 - M : Número de personas presentes en la secuencia (fijado usualmente en 1)
2. Archivo de etiquetas (.pkl): Un objeto serializado mediante *Pickle* que contiene una tupla de dos listas de longitud N :
 - La primera lista almacena los nombres o glosas de las señas.

- La segunda lista contiene los identificadores numéricos (*labels*) correspondientes a cada clase.

Aunque se evaluó la posibilidad de desarrollar un *script* de conversión externo (similar al flujo de trabajo utilizado en SPOTER), se optó por integrar el código original de generación de datos del repositorio de referencia. Esta decisión se fundamentó en la simplicidad de implementación y en la robustez que ofrece el pipeline de los autores originales para asegurar la compatibilidad con la arquitectura MSG3D.

Adecuación y Consistencia Experimental

A pesar de utilizar el código base de la *Universitat Pompeu Fabra*, fue necesario realizar modificaciones críticas para garantizar la validez comparativa de nuestros resultados. El objetivo principal fue asegurar que tanto la codificación de las glosas como la partición de los datos (*splits*) fueran idénticas a las utilizadas previamente en los experimentos con SPOTER.

Esta consistencia es fundamental para realizar un análisis comparativo riguroso; de esta manera, las variaciones en el rendimiento pueden atribuirse a las capacidades de las arquitecturas y no a discrepancias en la distribución de los datos de entrenamiento. Por ejemplo, permite extraer conclusiones específicas sobre si un modelo es superior en términos generales o si destaca en escenarios de baja representatividad.

Las modificaciones implementadas se resumen en dos puntos clave:

- Mapeo de Glosas: Se incorporó un parámetro de entrada adicional en el proceso de generación para cargar un archivo JSON con el mapeo exacto de identificadores y glosas definido para nuestro dataset de LSU.
- Control de Particiones: Se integró un segundo archivo JSON que define explícitamente qué instancias pertenecen a cada split (entrenamiento, validación y *test*). Cabe destacar que esta división se realizó a nivel de señal individual y no por video completo, basándonos en hallazgos previos donde este enfoque demostró una mejora en la capacidad de generalización del modelo de SPOTER.

4.2.2. Adaptaciones implementadas en el entrenamiento

Para optimizar el rendimiento de la arquitectura MSG3D en el contexto de la LSU, se implementaron diversas estrategias que abordan las limitaciones de los datos y buscan una evaluación más profunda del modelo.

Variación en la cantidad de clases a clasificar

Dada la alta dimensionalidad del espacio de etiquetas (446 clases) y la inherente escasez de muestras por glosa, se adoptó un enfoque experimental dual. Se entrenaron modelos utilizando la totalidad del corpus y, en paralelo, se realizó

una reducción a las 100 glosas más frecuentes. Esta simplificación permite evaluar el comportamiento del modelo ante un escenario de *long-tail distribution* frente a uno con mayor densidad de datos por categoría, así como comparar estos resultados con los obtenidos en los experimentos realizados con la arquitectura SPOTER.

Adicionalmente, y también imitando lo realizado en la etapa de experimentación con el modelo SPOTER, se integró una categoría denominada “clase no-seña” (o clase de fondo).

Top-5 y Balanced Accuracy

La inclusión de estas métricas asegura la interoperabilidad y comparabilidad de los resultados obtenidos con los experimentos previos realizados en SPOTER, manteniendo un estándar de evaluación uniforme entre arquitecturas.

Fine-tuning vs From-scratch

Aprovechando las capacidades del repositorio de referencia, se exploraron dos metodologías de entrenamiento para determinar cuál capitaliza mejor la información espacial y temporal de los esqueletos:

1. Entrenamiento desde cero (*From-scratch*): Donde los pesos de la red se inicializan de forma aleatoria y se ajustan íntegramente con los datos de LSU.
2. Ajuste fino (*Fine-tuning*): Se utilizó un modelo pre-entrenado con el dataset SWL-LSE (SignaMed Word-Level Spanish Sign Language). Durante el proceso de ajuste fino, se actualizaron los pesos de todas las capas de la red, es decir, no se congeló ninguna capa.

Esta técnica de *Transfer Learning* resulta de particular interés, ya que permite evaluar si las características espacio-temporales aprendidas de la Lengua de Señas Española (LSE) son transferibles y beneficiosas para el reconocimiento del LSU. Este enfoque busca compensar la falta de datos mediante el aprovechamiento de un “conocimiento previo” extraído de otra lengua.

4.2.3. Resultados entrenando desde cero

En esta sección se presentan y analizan los resultados obtenidos al entrenar la arquitectura MSG3D partiendo de una inicialización aleatoria de sus pesos. Todos los experimentos descritos a continuación fueron ejecutados utilizando la infraestructura de cómputo de alto rendimiento de ClusterUY, empleando la configuración detallada en secciones anteriores.

Estrategia de preprocesamiento y experimentación

A diferencia de los experimentos con SPOTER, en esta fase de entrenamiento con MSG3D se tomó la decisión técnica de prescindir de técnicas de aumento de datos. Esta resolución se fundamenta en dos pilares:

1. Evidencia empírica previa: Los experimentos realizados con la arquitectura SPOTER demostraron que el aumento de datos no generaba mejoras estadísticas significativas en la precisión final para nuestro conjunto de datos específico.
2. Aislamiento de variables: El objetivo central de esta etapa es evaluar la capacidad intrínseca de la arquitectura de grafos (GCN) para modelar las dependencias espaciotemporales del LSU y determinar si el *transfer learning* (que se abordará más adelante) ofrece una ventaja competitiva real frente al entrenamiento convencional.

Evaluación por volumen de clases

Considerando la distribución del *dataset*, se evaluó el desempeño del modelo bajo dos escenarios de clasificación distintos, permitiendo contrastar la robustez de la arquitectura frente a la variabilidad de las glosas:

- *Dataset* Completo (446 clases): Representa el desafío máximo de clasificación, abarcando la totalidad del léxico disponible en el corpus, incluyendo clases con baja representatividad.
- Subconjunto Top 100: Basándonos en las observaciones de los experimentos con SPOTER (donde la reducción del espacio de búsqueda a las 100 clases más frecuentes derivó en un incremento sustancial de la *accuracy*) se replicó esta configuración para establecer una línea base comparativa sólida entre ambas arquitecturas.

Implementación de la clase “no-seña”

Siguiendo la línea metodológica establecida en las etapas previas del proyecto, se integró una categoría adicional denominada “no-seña” en los experimentos con MSG3D. Esta clase tiene como propósito fundamental actuar como un sustrato de fondo que facilite la segmentación temporal en tareas de reconocimiento de señas continuo. Su función es permitir que el modelo identifique periodos de inactividad, movimientos de transición o gestos no léxicos, evitando así falsos positivos durante la inferencia en flujos de video.

Para garantizar la consistencia en la evaluación y permitir una comparación directa entre todas las configuraciones posibles, la clase “no-seña” se incluyó de manera transversal en los siguientes escenarios:

- Volumen de clases: Tanto en el clasificador de 446 clases (totalidad del corpus) como en la versión optimizada de las 100 glosas más frecuentes.
- Metodología de aprendizaje: Se aplicó tanto en el entrenamiento (*from scratch*) como en los procesos de *fine-tuning* basados en el modelo pre-entrenado con el *dataset* SWL-LSE.

Tabla 4.6: Desempeño del modelo MSG3D entrenado *from scratch* en el conjunto de prueba, variando *dataset* y uso de *no-seña*.

<i>Dataset</i>	No-Seña	Top-1	Top-5	Balanced Top-1	Balanced Top-5
LSU-446	No	0.327	0.522	0.294	0.464
LSU-100	No	0.329	0.556	0.325	0.533
LSU-446	Sí	0.292	0.493	0.242	0.420
LSU-100	Sí	0.340	0.562	0.341	0.527

Análisis de resultados: Entrenamiento desde cero

Los resultados sistematizados en la Tabla 4.6 revelan un comportamiento sumamente interesante de la arquitectura MSG3D en relación con la escalabilidad del vocabulario. A diferencia de las tendencias observadas en los experimentos previos con SPOTER, donde la reducción del número de clases generaba un salto drástico en el rendimiento, MSG3D demuestra una notable capacidad de adaptación incluso al trabajar con el vocabulario completo de LSU-446.

Un hallazgo sorprendente es la reducida brecha de precisión entre LSU-100 y LSU-446. En la métrica Top-1, la diferencia es de apenas 0,002 puntos (0,329 frente a 0,327). Asimismo, la métrica Balanced Top-1 para el vocabulario completo (0,294) se mantiene relativamente cercana a la *accuracy* global, lo que indica que el modelo no solo se apoya en las clases mayoritarias, sino que logra una capacidad de discriminación aceptable a través de toda la distribución del *dataset*.

En cuanto a la integración de la clase *no-seña*, el modelo exhibe una estabilidad destacable:

- Sobre LSU-100, la inclusión de esta categoría incluso mejora el desempeño, alcanzando un Top-1 de 0.340 (el valor más alto de la comparativa). Esto sugiere que la clase de fondo podría estar ayudando al modelo a filtrar ruido gestual que, de otro modo, se clasificaría erróneamente en una de las 100 glosas.
- Sobre LSU-446 con clase *no-seña*, aunque se observa un ligero descenso en la precisión (de 0,327 a 0,292), el margen de caída es significativamente menor al esperado para una tarea de esta complejidad.

4.2.4. Resultados de *transfer learning*

Una de las hipótesis centrales de esta etapa fue determinar si el *Transfer Learning* podría mitigar la escasez de datos en el corpus de LSU. La premisa se basó en que las características espacio-temporales genéricas aprendidas por el modelo en una lengua de señas con mayor volumen de datos, como la Lengua de Señas Española, podrían servir como una base sólida para el reconocimiento del LSU debido a posibles similitudes en las estructuras gestuales.

Tabla 4.7: Desempeño del modelo MSG3D pre-entrenado con SWL-LSE y *fine-tuned* con LSU en función del *dataset*.

<i>Dataset</i>	Top-1	Top-5	Balanced Top-1	Balanced Top-5
LSU-446	0.323	0.504	0.280	0.457
LSU-100	0.335	0.545	0.303	0.501

Comparativa y Desempeño

A diferencia de los hallazgos reportados en el trabajo de referencia de la Universitat Pompeu Fabra (Naiara Garmendia, Horacio Saggion y Euan McGill, 2025a), donde el uso de un modelo pre-entrenado con el dataset SWL-LSE resultó en una mejora significativa del rendimiento, nuestros resultados (presentados en la Tabla 4.7) muestran un comportamiento distinto.

Al contrastar estos valores con los obtenidos mediante el entrenamiento desde cero, se observan los siguientes puntos clave:

- Margen de mejora nulo: La diferencia en las métricas entre ambas metodologías es marginal. Por ejemplo, en el vocabulario completo de 446 clases, el Top-1 del modelo ajustado (0,323) es incluso ligeramente inferior al del modelo entrenado desde cero (0,327).
- Estancamiento en el subconjunto Top 100: Aunque el ajuste fino en el subconjunto de 100 clases alcanzó un Top-1 de 0,335 (superando levemente al 0,329 del entrenamiento base), la ganancia no es estadísticamente significativa ni justifica la complejidad adicional del proceso de pre-entrenamiento.
- Métricas balanceadas: Se observa una tendencia similar en el Balanced Top-1, donde el modelo entrenado desde cero mantiene una ligera ventaja, sugiriendo que el pre-entrenamiento en LSE no necesariamente aporta una mejor discriminación para las clases con menos muestras en LSU.

Este fenómeno sugiere que la distancia lingüística o las diferencias en las condiciones de captura entre el dataset SWL-LSE y nuestro corpus de LSU podrían estar limitando la efectividad de la transferencia de conocimiento. En algunos casos, un modelo pre-entrenado en un dominio ligeramente distinto puede introducir sesgos que dificultan la convergencia óptima en el nuevo dominio, un efecto conocido como *negative transfer* (W. Zhang, Deng, y Wu, 2020).

Dado que el rendimiento superior se obtuvo de manera consistente mediante el entrenamiento desde cero, se tomó la decisión técnica de desestimar el uso de modelos fine-tuneados para las etapas posteriores de este proyecto. Los experimentos finales y la implementación de los sistemas de *sign spotting* se basarán exclusivamente en las versiones de modelos entrenadas íntegramente con datos de LSU.

Métricas para clase no-seña

Dada la relevancia de la categoría “no-seña” para la futura implementación de un sistema de *sign spotting* mediante la técnica de ventana deslizante, es imperativo analizar su comportamiento de forma aislada. A diferencia de las glosas léxicas (como “casa” o “persona”), que poseen una estructura gestual definida y repetible, la clase “no-seña” es una categoría residual y heterogénea. Esta comprende cualquier estado que no constituya una seña formal: desde el reposo absoluto y ajustes de postura hasta movimientos de transición entre señas o gestos involuntarios.

Tabla 4.8: Métricas detalladas para la clase de fondo (no-seña) en el conjunto de prueba, utilizando la arquitectura MSG3D inicializada aleatoriamente sobre el subconjunto de 100 glosas.

Clase	Precision	Recall	F1 Score
No seña	0.2	0.237	0.217

En la Tabla 4.8, se presentan las métricas de precisión, *recall* y *F1-score* para esta clase en la configuración que reportó los mejores resultados generales (100 clases + no-seña, entrenado *from scratch*).

Los valores obtenidos, aunque numéricamente inferiores al promedio de las clases léxicas, reflejan la dificultad intrínseca de clasificar la “nada” gestual:

- Equilibrio entre Precisión y Recall: Con una precisión de 0,200 y un recall de 0,237, el modelo demuestra una capacidad incipiente pero funcional para distinguir segmentos de no-seña. El hecho de que el recall sea ligeramente superior a la precisión indica que el modelo es relativamente “sensible” a detectar momentos de inactividad, aunque a costa de clasificar ocasionalmente señas reales como ruido (falsos positivos de no-seña).
- Análisis del F_1 -Score (0,217): Este valor confirma que, si bien la clase es desafiante debido a su alta varianza y falta de estructura semántica, el modelo MSG3D logra extraer características espacio-temporales que la diferencian del léxico formal.

4.3. Análisis comparativo

Tras completar las fases de experimentación para ambas arquitecturas, se realizó una comparativa integral para determinar cuál de los modelos se adapta mejor a las particularidades de la LSU. A continuación, se presenta la Tabla 4.9, que sintetiza el desempeño de ambos modelos en los dos escenarios de vocabulario más representativos.

Tabla 4.9: Comparativa de desempeño entre SPOTER y MSG3D sobre LSU-446 y LSU-100 en el conjunto de prueba. Sobre LSU-446 se reporta la configuración E2 de SPOTER (Tabla 4.2) y MSG3D entrenado desde cero (Tabla 4.6); sobre LSU-100 se reporta la configuración sin aumentado de ambas arquitecturas.

Modelo	Dataset	Top-1	Top-5	Bal. Top-1	Bal. Top-5
SPOTER	LSU-446	0.243	0.430	0.225	0.370
MSG3D	LSU-446	0.327	0.522	0.294	0.464
SPOTER	LSU-100	0.320	0.531	0.305	0.503
MSG3D	LSU-100	0.329	0.556	0.325	0.533

Robustez ante el crecimiento del vocabulario

La observación más destacada de esta comparativa es la estabilidad de MSG3D frente al incremento de clases. Mientras que SPOTER sufre una degradación significativa en su *accuracy* Top-1 al pasar de LSU-100 a LSU-446 (cayendo de 0,320 a 0,243), MSG3D logra mantener un rendimiento prácticamente constante (0,329 frente a 0,327). Esta diferencia sugiere que la arquitectura de grafos de MSG3D es intrínsecamente más robusta para modelar la complejidad de LSU-446. La capacidad de MSG3D para capturar dependencias espaciales y temporales de forma simultánea a través de múltiples escalas le permite discriminar mejor entre glosas similares, incluso cuando el espacio de búsqueda se cuadruplica.

Otro punto de contraste relevante es la dependencia de técnicas de balanceo. Para que SPOTER alcanzara sus mejores métricas, fue necesaria una exploración exhaustiva de estrategias de *data augmentation*. En contrapartida, MSG3D alcanzó resultados superiores entrenando directamente desde cero (*from scratch*) y sin necesidad de aumentar artificialmente el *dataset*. Este hecho refuerza la superioridad de las redes de convolución de grafos (GCN) para este dominio específico:

- Modelado Geométrico: MSG3D aprovecha la estructura jerárquica del esqueleto humano, lo que facilita el aprendizaje de patrones de movimiento naturales.
- Métricas Balanceadas: Como se observa en la métrica *Balanced Top-1*, MSG3D supera a SPOTER en todos los escenarios, lo que indica una mejor capacidad para reconocer señas con pocas instancias, mitigando de forma más efectiva el sesgo producido por el desbalance del corpus LSU.

Si bien ambos modelos muestran que el Top-5 *Accuracy* es considerablemente mayor al Top-1 (confirmando que ambas arquitecturas logran extraer información semántica relevante incluso en sus errores), MSG3D se posiciona como la arquitectura más eficaz.

Capítulo 5

Sign Spotting: experimentación

Una vez concluida la etapa de experimentación orientada al reconocimiento de señas aisladas, se dio inicio al proceso de adaptación de los modelos obtenidos hacia sistemas de *sign spotting*, que constituye el objetivo principal de este proyecto.

Los experimentos realizados en esta fase se organizan en torno a dos enfoques metodológicos distintos:

- Segmentación automática del video utilizando un modelo preentrenado.
- Reconocimiento continuo mediante ventana deslizante.

Cada una de estas estrategias será descrita y analizada en profundidad en las secciones siguientes. Cabe destacar que los sistemas desarrollados para la tarea de *sign spotting* se basan principalmente en los modelos de reconocimiento aislado discutidos en el Capítulo 4. Para ello, se reutilizan los modelos entrenados previamente, seleccionando en cada caso la mejor configuración obtenida para cada arquitectura, y evaluando su desempeño al ser integrados en un escenario donde el dato de entrada puede contener potencialmente una secuencia de señas a reconocer.

5.1. Metodología de evaluación

La evaluación de un sistema de *sign spotting* de lengua de señas presenta desafíos adicionales respecto a la evaluación de modelos de reconocimiento aislado, ya que involucra no solo la correcta clasificación de las señas, sino también la detección precisa de sus límites temporales dentro de una secuencia continua. En este contexto, resulta fundamental definir una metodología de evaluación que permita analizar de manera integral el desempeño del sistema en escenarios realistas de uso.

En esta sección se describe la metodología adoptada para evaluar los sistemas de reconocimiento continuo desarrollados en este trabajo y se detallan las características del video de prueba utilizado. Asimismo, se explican las decisiones metodológicas tomadas para garantizar una evaluación rigurosa, evitando sesgos derivados del uso de datos previamente utilizados durante el entrenamiento de los modelos.

5.1.1. Métricas principales

A diferencia del análisis llevado a cabo en el Capítulo 4, en este enfoque, la evaluación del sistema se estructura en dos dimensiones: la capacidad de detección temporal y la precisión en el reconocimiento lingüístico. Para ello, las métricas se definen en un sistema jerárquico, en el que cada nivel representa una condición más exigente que el nivel anterior. De este modo, se puede aislar el error de segmentación (nivel 1) del error de clasificación (niveles 2 y 3) y analizar el desempeño de cada etapa. Para cada nivel, se reporta el éxito de detección con la métrica F_1 , así como la fidelidad del ajuste temporal a través del índice IoU , que será detallado a continuación.

Nivel 1: Segmentación Temporal

En el primer nivel de análisis, el objetivo es validar si el sistema es capaz de identificar la existencia de una señal. Para esto, se distinguen dos conceptos: la **detección por solapamiento** y la **fidelidad de alineación**.

Criterio de Solapamiento Mínimo (Métrica Laxa) Se define una métrica de detección permisiva donde un segmento predicho S_p se considera un Verdadero Positivo (TP) si existe una intersección no vacía con un segmento de referencia S_g ($|S_p \cap S_g| > 0$). Basta con que un solo *frame* coincida para validar la detección. Los resultados de este nivel se reportan mediante el **Valor-F1** (F_1), que balancea la precisión y la exhaustividad de la detección.

Fidelidad Temporal mediante IoU La métrica F_1 por sí sola no captura la calidad del ajuste temporal: un sistema puede detectar correctamente la presencia de todas las señales y aun así producir segmentos con fronteras significativamente desalineadas respecto al *ground truth*. Complementariamente, se utiliza la *Intersection over Union* (IoU) para medir la precisión de los límites temporales detectados:

$$IoU(S_p, S_g) = \frac{|S_p \cap S_g|}{|S_p \cup S_g|} \quad (5.1)$$

Mientras que el F_1 del Nivel 1 mide si el sistema encontró la señal, el IoU mide la calidad del recorte de sus fronteras.

IoU Promedio Como el *IoU* por segmento es una medida local, resulta necesario agregarlo en un valor representativo del desempeño global. Se define el IoU Promedio como el promedio del *IoU* sobre un conjunto de pares (S_p, S_g) , que admite dos variantes naturales según la dirección de la asociación: (1) la variante *centrada en la referencia*, que para cada segmento de referencia toma su mejor predicción asociada y promedia los *IoU* resultantes, y (2) la variante *centrada en las predicciones*, que para cada segmento predicho toma su mejor referencia asociada y promedia los *IoU* resultantes.

IoU General Dado que cada una de las dos variantes del IoU Promedio refleja solo una de las dos formas posibles de error temporal (señas no detectadas en un caso, predicciones espurias en el otro), se define el **IoU General** como el promedio de ambas. Esta formulación bidireccional penaliza tanto las señas no detectadas (que reducen el promedio desde la referencia) como las predicciones incorrectas (que reducen el promedio desde las predicciones), y es la métrica utilizada para reportar la fidelidad temporal en las evaluaciones jerárquicas del Capítulo.

Nivel 2 y Nivel 3: Reconocimiento Integrado

En los niveles 2 y 3, la evaluación integra la clasificación lingüística. Un segmento se valida únicamente si presenta solapamiento temporal con un segmento de referencia (la misma condición de detección del Nivel 1) y la clase predicha coincide con la condición de etiqueta correspondiente al nivel. Para mantener la coherencia de la métrica de fidelidad temporal en estos niveles integrados, se asigna un valor de $IoU = 0$ a aquellos segmentos donde la clasificación fue errónea, de modo que el error lingüístico se refleje también como una falla de calidad temporal en el promedio global.

Nivel 2: Reconocimiento Top-5 Este nivel evalúa si la seña correcta está presente en las primeras cinco sugerencias del modelo. Sea $f(\cdot)$ la función de clasificación del sistema, un segmento se considera correcto si existe solapamiento temporal y la glosa de referencia $G_g \in \text{Top-5}(f(S_p))$.

Nivel 3: Reconocimiento Top-1 Representa la clasificación total y es el nivel más riguroso. Sea $f(\cdot)$ la función de clasificación del sistema, un segmento es correcto si existe solapamiento temporal y $G_g = \text{Top-1}(f(S_p))$.

5.1.2. Características del conjunto de prueba

Para la evaluación de los sistemas de reconocimiento continuo se utilizó un video nuevo que no formaba parte del dataset empleado para el entrenamiento de los modelos de reconocimiento de señas aisladas. Adicionalmente, el intérprete de LSU del video tampoco figuraba en otros videos del dataset, y el contenido del video corresponde a un dominio distinto al del conjunto de entrenamiento (lo que constituye un desafío adicional para los modelos). Esta elección permite analizar

el desempeño del sistema en un escenario completamente no visto durante el entrenamiento, evitando posibles sesgos y garantizando una evaluación realista.

El video seleccionado corresponde a una sesión de la Cámara de Representantes de Uruguay y contiene un total de 730 señas distribuidas a lo largo de 10 minutos y 31 segundos de grabación. Se trata de un contexto discursivo formal y continuo, representativo de un escenario real de uso del lenguaje.

Como consecuencia de esta diferencia de dominio, el video presenta señas que no se encuentran en el conjunto de entrenamiento, tales como *COOPERATIVA*, *ARTÍCULO* o *SENADXR*. Estas señas se denominan *out-of-vocabulary* (OOV) y representan aproximadamente el 33,8% de las señas del video de prueba. La presencia de OOV es una propiedad del conjunto de prueba y afecta por igual a todos los enfoques metodológicos considerados. Asimismo, algunas señas sí presentes en el dataset original aparecen con una frecuencia considerablemente menor, como es el caso de la seña *VOTAR*. Esta diferencia en la distribución léxica introduce un desafío adicional para el sistema.

Para asegurar una evaluación comparable entre métodos, las señas OOV no se excluyen del *ground truth*. En las métricas de detección temporal, las señas OOV se consideran instancias válidas de actividad lingüística. En las métricas de clasificación lingüística, en cambio, constituyen fallos inevitables, ya que ninguna arquitectura puede predecir etiquetas que no forman parte de su vocabulario.

5.2. Modelos de reconocimiento aislado utilizados

A lo largo de este capítulo se reutilizan cuatro configuraciones específicas de los modelos de reconocimiento aislado entrenadas y reportadas en el Capítulo 4, todas sobre el vocabulario completo de LSU-446.

El enfoque que opera sobre segmentos ya recortados (Sección 5.3) utiliza las variantes sin clase *no-seña*: SPOTER en la configuración E2 sobre LSU-446 (fila correspondiente de la Tabla 4.2) y MSG3D entrenado desde cero sobre LSU-446 (fila correspondiente de la Tabla 4.6).

El enfoque de ventana deslizante (Sección 5.4), en el que el propio modelo cumple el doble rol de detección y clasificación, utiliza las contrapartes entrenadas con la clase *no-seña*: SPOTER sobre LSU-446 con clase *no-seña*, sin aumentado (fila correspondiente de la Tabla 4.3) y MSG3D entrenado desde cero sobre LSU-446 con clase *no-seña* (fila correspondiente de la Tabla 4.6).

En lo que sigue del capítulo, las menciones a SPOTER y MSG3D deben interpretarse según el enfoque que se esté presentando: en la Sección 5.3 se refieren a sus respectivas configuraciones sin clase *no-seña*, y en la Sección 5.4 a sus contrapartes con dicha clase.

5.3. Modelo de segmentación para escenarios reales

En esta sección se presenta el uso de un modelo de segmentación diseñado para operar en escenarios reales, donde las secuencias de entrada pueden incluir transiciones suaves, coarticulación entre signos y variaciones significativas entre distintos intérpretes. Este modelo actúa como un preprocesamiento automático del video continuo, generando segmentos candidatos que posteriormente pueden ser clasificados mediante modelos de reconocimiento de señas aisladas.

Para esta tarea se adoptó como base el trabajo de (Amit Moryossef, 2023), en el cual se proponen y evalúan distintos modelos de segmentación entrenados sobre representaciones basadas en la estimación de posturas.

A continuación, se describen las características principales del modelo de segmentación empleado, así como su integración dentro del pipeline de reconocimiento continuo propuesto en este trabajo, reportando métricas finales sobre el video de prueba.

5.3.1. Selección del modelo de segmentación

Entre los modelos presentados en el trabajo de Moryossef et al. (Amit Moryossef, 2023), se seleccionó como *baseline* el modelo **E1s**, dado que es el que reporta los mejores resultados globales en términos de precisión de segmentación. E1s es una red convolucional temporal multi-etapa (MS-TCN) que recibe como entrada secuencias de *keypoints* de postura y produce una clasificación a nivel de *frame*, determinando si cada instante pertenece a una seña o a una transición entre señas. A partir de estas predicciones binarias se derivan los límites temporales de los segmentos candidatos. Este modelo se utiliza inicialmente sin modificaciones, empleando los pesos preentrenados provistos en el repositorio original de los autores.

Si bien la evaluación final de este enfoque se realizó sobre el nuevo video que se mencionó en la sección anterior, en la etapa inicial de selección del modelo de segmentación no se contaba aún con dicho video. Por este motivo, se optó por emplear un video del dataset como insumo preliminar para evaluar estos modelos. Cabe aclarar que el uso de un video perteneciente al dataset original no representa una limitación metodológica en esta etapa, porque no intervienen los modelos de reconocimiento aislado, y el objetivo se restringe exclusivamente a comparar el comportamiento relativo de los distintos modelos segmentadores.

De este modo, se evaluó inicialmente el rendimiento del modelo preentrenado utilizando dos métricas específicas para el análisis de la segmentación temporal, ambas ya introducidas en la Sección 5.1.1: el F_1 bajo el Criterio de Solapamiento Mínimo (métrica laxa) y el IoU Promedio (en su variante centrada en la referencia). Los valores resultantes, reportados en la Tabla 5.1, establecen una línea base del desempeño del segmentador en un escenario no visto durante su entrenamiento original.

Fine-tuning del modelo

A partir de la línea base obtenida, se realizó un proceso de ajuste fino del modelo E1s utilizando el resto de los videos disponibles en el dataset. El objetivo de esta etapa fue adaptar el segmentador a las características particulares de los datos utilizados en este trabajo. Una vez finalizado el proceso de *fine-tuning*, el modelo ajustado fue evaluado nuevamente sobre el mismo video utilizado en la evaluación inicial, empleando exactamente las mismas métricas.

La Tabla 5.1 resume los valores obtenidos para ambos modelos.

Tabla 5.1: Comparación del rendimiento en segmentación temporal entre el modelo base y el modelo ajustado.

Métrica	Modelo Base	Modelo Fine-Tuned
F1 (Criterio Laxo)	88,90 %	88,16 %
IoU Promedio	32,47 %	31,73 %

Los resultados obtenidos muestran que el modelo ajustado presenta un desempeño levemente inferior al del modelo preentrenado original. Dado que la diferencia observada es marginal y no representa una mejora respecto al modelo base, se decidió continuar el desarrollo utilizando el modelo E1s preentrenado provisto por los autores.

Más allá de la comparación entre variantes, los resultados absolutos del segmentador revelan un comportamiento asimétrico. El F_1 cercano al 89 % indica que el modelo es eficaz para detectar las regiones del video donde ocurren señas. Sin embargo, el IoU promedio de aproximadamente 32 % muestra que los segmentos detectados cubren, en promedio, menos de un tercio de la duración real de cada seña. Esto sugiere que el segmentador tiende a generar recortes temporales más cortos que las señas reales, o que considera como señas segmentos de video que no corresponden. Estas características se propagarán al clasificador y es conveniente tener presente al interpretar los resultados de reconocimiento en secciones posteriores.

5.3.2. Integración con el reconocimiento de señas aisladas

Una vez seleccionado el modelo de segmentación, se procedió a evaluar el sistema completo de reconocimiento continuo. Para ello, se tomó el video nuevo mencionado anteriormente y se lo segmentó automáticamente utilizando el modelo E1s, obteniendo una serie de segmentos temporales entendidos por el modelo como señas individuales.

Cada uno de estos segmentos fue posteriormente clasificado utilizando las configuraciones de SPOTER y MSG3D definidas en la Sección 5.2 para el enfoque con segmentador (Tabla 4.9). De esta forma, se pudo analizar el desempeño del sistema completo, desde la segmentación automática hasta la predicción final de la seña, utilizando las métricas introducidas en la sección 5.1.1.

Es importante destacar que el modelo de segmentación y los modelos de reconocimiento aislado no utilizan exactamente la misma representación de entrada. En particular, el segmentador emplea como entrada *keypoints* obtenidos mediante *MediaPipe Holistic*, que incluye información de manos, rostro y postura corporal completa. Esta diferencia implica una etapa adicional de pre-procesamiento en el pipeline, pero permite aprovechar modelos especializados para cada tarea, manteniendo la segmentación y la clasificación como módulos conceptualmente independientes.

5.3.3. Resultados

En esta sección se analiza el desempeño de los modelos **MSG3D** y **SPO-TER** sobre el video de prueba procesado por el segmentador. La Tabla 5.2 presenta los resultados bajo el marco de evaluación jerárquico definido en la sección 5.1.1.

Tabla 5.2: Comparación de rendimiento estándar entre modelos MSG3D y SPOTER.

Nivel de Evaluación	Modelo MSG3D		Modelo SPOTER	
	F1 Score	IoU G.	F1 Score	IoU G.
Nivel 1: Segmentación	95,86 %	43,44 %	95,86 %	43,44 %
Nivel 2: Reconocimiento Top-5	13,53 %	6,87 %	15,61 %	7,42 %
Nivel 3: Reconocimiento Top-1	7,07 %	3,89 %	6,84 %	3,18 %

El desempeño de ambos modelos permite identificar fortalezas diferenciadas en la cadena de procesamiento.

Segmentación Temporal (Nivel 1)

Ambos modelos presentan el mismo desempeño en la detección de eventos temporales (F_1 de 95,86 % e IoU General de 43,44 %). Esta coincidencia es esperable: la segmentación está enteramente determinada por el modelo E1s, que es el mismo para ambas configuraciones, y el video se procesa una única vez para producir el conjunto de segmentos que luego se reutiliza para ambos clasificadores. El F_1 cercano al 96 % y el **IoU General** de aproximadamente 43,4 % indican que, si bien el sistema identifica la presencia de casi la totalidad de las señas, existe una imprecisión sistemática en el ajuste fino de las fronteras inicial y final de cada segmento. Este resultado es consistente con lo observado en la evaluación del segmentador aislado, donde el IoU promedio se ubicaba en torno al 32 %.

Reconocimiento Top-5 (Nivel 2)

Al incorporar la exigencia de clasificación, el F_1 cae drásticamente a 15,61 % para SPOTER y 13,53 % para MSG3D. Esta caída respecto al Nivel 1 se ex-

plica por dos factores. Por un lado, los propios modelos ISLR presentan un rendimiento limitado en reconocimiento aislado (Top-5 de 52,2 % para MSG3D y 43,0 % para SPOTER sobre el conjunto de prueba, según la Tabla 4.9), por lo que es esperable que no acierten consistentemente al operar sobre los segmentos extraídos en esta nueva configuración. Por otro lado, los segmentos producidos por el segmentador no coinciden exactamente con señas individuales (como lo evidencia el IoU de ~ 43 % en Nivel 1), lo que implica que los clasificadores reciben entradas con fronteras imprecisas, contenido parcial de señas adyacentes o transiciones, degradando aún más su capacidad de predicción respecto al escenario aislado. El IoU General también desciende significativamente (7,42 % para SPOTER y 6,87 % para MSG3D), combinando ambos efectos.

Reconocimiento Top-1 (Nivel 3)

En el nivel más exigente, ambos modelos convergen a un desempeño similar: F_1 de 7,07 % para MSG3D y 6,84 % para SPOTER. Es notable que el orden relativo se invierte respecto al Nivel 2: MSG3D, que era inferior en Top-5, resulta levemente superior en Top-1. Esto sugiere que su predicción más probable es más precisa, aunque su conjunto de candidatos sea menos confiable que el de SPOTER. El IoU General se ubica en 3,89 % para MSG3D y 3,18 % para SPOTER.

Comparación entre arquitecturas

Los resultados revelan un compromiso entre cobertura y precisión: **SPOTER** lidera en el Reconocimiento Top-5, mientras que **MSG3D** es superior en el Reconocimiento Top-1, logrando mayor certeza en su predicción individual. Es notable que este ordenamiento se invierte respecto al reconocimiento aislado, donde MSG3D supera a SPOTER en ambas métricas (Tabla 4.9). Esto indica que la ventaja de SPOTER en Top-5 emerge específicamente al operar sobre segmentos imperfectos, sugiriendo que su ranking de candidatos se degrada de forma más gradual ante entradas con fronteras temporales imprecisas. Este fenómeno también será observado en el apartado siguiente con el enfoque de ventana deslizante.

En conclusión, los datos sugieren que para aplicaciones de asistencia donde un humano valida opciones, el modelo SPOTER es el más adecuado. Por el contrario, para escenarios donde se requiere la predicción más probable, el modelo MSG3D ofrece resultados más precisos. No obstante, cabe destacar que en cualquiera de los dos casos los resultados obtenidos se encuentran lejos de permitir un uso autónomo del sistema sin intervención humana. Los niveles de reconocimiento alcanzados posicionan a estos sistemas como herramientas de asistencia más que como soluciones independientes en su estado actual.

Un análisis complementario mediante métricas adicionales – incluyendo IoU Condicional, variantes de robustez del solapamiento y métricas Best-Match – se presenta en el Anexo B.1. Dicho análisis permite desacoplar los errores de segmentación de los de clasificación, y revela que el IoU Condicional de MSG3D

alcanza un 54,24 % en Nivel 3. Es decir, en los casos en que el modelo clasifica correctamente una seña, los segmentos sobre los que se realizó la predicción presentan un buen ajuste temporal. Esta observación permite hipotetizar que, si se contara con un segmentador de mayor calidad capaz de producir fronteras temporales más precisas de forma sistemática, el desempeño del sistema completo podría mejorar significativamente.

5.4. Enfoque de ventana deslizante

En esta sección se presenta el segundo enfoque explorado para la tarea de *sign spotting*: la ventana deslizante. A diferencia del sistema basado en un segmentador, donde un modelo dedicado detecta los límites temporales de cada seña y luego un modelo ISLR clasifica los segmentos resultantes, en este enfoque el propio modelo ISLR cumple ambos roles simultáneamente. El modelo se aplica de forma repetida sobre ventanas temporales superpuestas que recorren el video, y las predicciones se agregan mediante un esquema de votación por *frame*.

Este paradigma requiere que los modelos sean capaces de distinguir la ausencia de señas además de clasificarlas, por lo que se utilizan los modelos entrenados con la clase NO-SEÑA descritos en la sección 4.1.4. Las métricas de reconocimiento de dicha clase resultan un indicador relevante de la capacidad del método para delimitar temporalmente las señas en un video continuo.

Al igual que en el enfoque anterior, se evaluaron las arquitecturas SPOTER y MSG3D, en este caso bajo las configuraciones con clase NO-SEÑA definidas en la Sección 5.2. En ambos casos, la reproducción exacta del *pipeline* de preprocesamiento de los *keypoints* aplicado durante el entrenamiento resultó esencial para obtener resultados representativos. La evaluación se realizó sobre el mismo video de prueba descrito en la sección 5.1.2.

5.4.1. Descripción del método

Dado un video continuo representado como una secuencia de N *frames*, se define una ventana de longitud fija W (en *frames*) que se desplaza con un paso constante (*stride*) S . En cada posición $i \in \{0, S, 2S, \dots\}$, se extrae la subsecuencia de *frames* $\{f_i, f_{i+1}, \dots, f_{i+W-1}\}$ y se la presenta al modelo ISLR, obteniéndose una predicción Top-1 y un ranking Top-5.

Votación por *frame* En lugar de asignar las predicciones directamente a la ventana completa, se adopta un esquema de **votación por *frame***: cada ventana contribuye con un voto Top-1 a la clase predicha, y 5 votos Top-5 al ranking obtenido, para todos los *frames* que la componen. De esta forma, cada *frame* f_j acumula votos de todas las ventanas que lo cubren, construyendo un histograma de clases. Como las ventanas avanzan en pasos de S *frames*, los *frames* dentro de un mismo bloque de S posiciones alineado al *stride* quedan cubiertos por exactamente el mismo conjunto de ventanas. Por ejemplo, los

frames $0, 1, \dots, S - 1$ son cubiertos por las mismas ventanas; lo mismo ocurre con los *frames* $S, S + 1, \dots, 2S - 1$, y así sucesivamente. Cada uno de estos bloques comparte, en consecuencia, el mismo histograma de votos. Para *frames* interiores (alejados de los bordes del video), la cantidad de ventanas que cubren un *frame* es aproximadamente $\lfloor W/S \rfloor$; cerca de los extremos, esta cobertura es menor.

A partir del histograma se obtienen dos representaciones por *frame*:

- **Clase ganadora (Top-1 por *frame*):** la clase con mayor cantidad de votos acumulados.
- **Conjunto de clases presentes (Top-N por *frame*):** el conjunto de todas las clases que recibieron al menos un voto, proveniente de haber figurado en el Top-5 de alguna ventana que cubre el *frame*. Este conjunto se utiliza únicamente para el cálculo de las métricas de Nivel 2 (Reconocimiento Top-5), donde la seña de referencia se considera detectada si pertenece al conjunto. La derivación de segmentos descrita a continuación opera exclusivamente sobre la clase ganadora.

Consenso de predicciones y derivación de segmentos. Si bien la representación por *frame* permite un análisis de precisión a nivel de cada instante temporal, es necesario traducir las predicciones a segmentos temporales para aplicar las métricas definidas en la sección 5.1.1 y poder comparar directamente con el enfoque de segmentación. Para ello, toda sucesión de *frames* contiguos clasificados con la misma clase se interpreta como un segmento, y toda sucesión de *frames* contiguos clasificados como NO-SEÑA se interpreta como un espacio entre segmentos. De este modo, la secuencia de predicciones por *frame* genera una partición del video en segmentos candidatos.

Filtro de mediana para post-procesado. Con el objetivo de obtener una partición más robusta, se implementó un filtro de mediana sobre las predicciones por *frame* previo a la derivación de segmentos. El filtro asume la relación $W = 3S$ y recorre la secuencia evaluando bloques de tres *strides* consecutivos. Si la clase predicha en el primer y tercer bloque coincide (y es distinta de NO-SEÑA) pero la del bloque central difiere, la predicción del bloque central se sobrescribe con la clase mayoritaria de los extremos. De esta manera se busca eliminar predicciones erróneas en un *stride* que generen una interrupción en un segmento contiguo.

5.4.2. Configuración de la ventana

Para que el filtro de mediana descrito anteriormente pueda aplicarse de manera directa sobre la secuencia de *strides*, los parámetros W y S deben satisfacer la relación $W = 3S$. Esta restricción no es una propiedad intrínseca del enfoque de ventana deslizante, sino una consecuencia del diseño particular del filtro

adoptado en este trabajo, que opera sobre bloques de tres *strides* consecutivos. Se eligió este filtro por su simplicidad de implementación y porque permite acotar el espacio de parámetros a explorar dentro del alcance del proyecto.

Bajo esta restricción, la elección concreta de W se motivó a partir de la duración promedio de las señas en el video de prueba, aproximadamente 14,7 *frames* a 25 fps. Se evaluaron dos configuraciones que comparten la misma densidad de votación ($\lfloor W/S \rfloor = 3$) y rodean dicha duración promedio:

- ($W = 18, S = 6$): Apunta a capturar una seña completa junto con parte de sus transiciones, proveyendo mayor contexto al clasificador.
- ($W = 12, S = 4$): Explora la posibilidad de que una ventana más corta produzca histogramas de votación más puros al reducir la proporción de *frames* pertenecientes a señas adyacentes.

Los resultados experimentales favorecieron consistentemente la configuración $W = 12, S = 4$, lo cual resulta coherente con dicha duración promedio: una ventana de 12 *frames* se ajusta mejor al núcleo temporal de una seña típica, reduciendo la contaminación por *frames* de transición o señas adyacentes. En consecuencia, todos los resultados reportados en esta sección corresponden a la configuración $W = 12, S = 4$.

El número total de ventanas procesadas depende de la longitud del video. Para el video de prueba (10 minutos y 31 segundos, 730 señas anotadas), se procesan 15 782 *frames* a 25 fps, resultando en 3 943 ventanas para la configuración seleccionada.

5.4.3. Resultados en video de prueba

En esta sección se presentan los resultados del enfoque de ventana deslizante sobre el video de prueba, utilizando la configuración de ventana $W = 12$ con paso $S = 4$ y el filtro de suavizado por mayoría activado. Al igual que en el enfoque con segmentador, las métricas se reportan sobre la totalidad de las 730 señas del *ground truth*.

La Tabla 5.3 presenta los resultados principales para ambos modelos.

Tabla 5.3: Rendimiento de MSG3D y SPOTER con ventana deslizante ($W = 12, S = 4$, con filtro).

Nivel de Evaluación	MSG3D		SPOTER	
	F ₁	IoU G.	F ₁	IoU G.
Nivel 1: Segmentación	99,79 %	10,05 %	99,66 %	8,72 %
Nivel 2: Reconocimiento Top-5	6,44 %	2,29 %	12,06 %	2,20 %
Nivel 3: Reconocimiento Top-1	3,11 %	1,40 %	3,12 %	0,32 %

Segmentación Temporal (Nivel 1)

Ambos modelos alcanzan un F_1 cercano al 100 % en la detección temporal (99,79 % para MSG3D y 99,66 % para SPOTER). Este comportamiento es esperable bajo este enfoque: al recorrer el video con ventanas solapadas y agrupar predicciones consecutivas de la misma clase, el sistema genera segmentos que cubren la secuencia temporal de forma casi continua, por lo que prácticamente cualquier señal del *ground truth* queda contenida en al menos un segmento predicho. Por este motivo, el F_1 del Nivel 1 no es en sí mismo el indicador más informativo del desempeño del método en este enfoque, y conviene leerlo junto con el IoU General que se discute a continuación.

El **IoU General**, en cambio, se ubica en valores notablemente bajos: 10,05 % para MSG3D y 8,72 % para SPOTER. Esta divergencia entre un F_1 casi perfecto y un IoU reducido refleja una característica estructural del enfoque. El sistema produce un número limitado de segmentos predichos (38 para MSG3D y 31 para SPOTER) frente a las 730 señales del *ground truth*, por lo que cada segmento predicho abarca temporalmente múltiples señales reales: esto garantiza la detección (F_1 alto) pero resulta en un ajuste temporal pobre entre las fronteras predichas y las reales (IoU bajo). Este comportamiento es consistente con el bajo rendimiento de ambos modelos en el reconocimiento de la clase *no-señal* en evaluación aislada (Sección 4.1.4): dado que la clase *no-señal* es la que delimita los intervalos entre señales consecutivas, su escasa predicción impide que el sistema separe segmentos adyacentes, produciendo en su lugar segmentos extensos que engloban múltiples señales. El tamaño de ventana elegido también podría incidir en este efecto: con ventanas más cortas, las regiones de transición entre señales tienen mayor probabilidad de quedar contenidas íntegramente en una ventana, lo que induciría al modelo a predecir *no-señal* con mayor frecuencia y, por tanto, a generar más fronteras de segmento; explorar configuraciones alternativas de ventana queda planteado como línea de trabajo futuro (Sección 6.4.2).

Reconocimiento Top-5 (Nivel 2)

En el Nivel 2 se observa la mayor diferencia entre arquitecturas. SPOTER alcanza un F_1 de 12,06 %, casi el doble que MSG3D (6,44 %). Al descomponer esta métrica, se observa que SPOTER logra una precisión del 72,73 % contra el 53,19 % de MSG3D, indicando que cuando SPOTER incluye la clase correcta dentro de su Top-5, lo hace con mayor confiabilidad. No obstante, ambos modelos presentan una exhaustividad muy baja: 6,58 % para SPOTER y 3,42 % para MSG3D, lo que refleja que la gran mayoría de las señales no son recuperadas correctamente en este nivel.

Este bajo rendimiento es esperado considerando que los modelos ISLR ya presentan un rendimiento moderado en reconocimiento aislado (Top-5 de 52,2 % para MSG3D y 43,0 % para SPOTER, según la Tabla 4.9). A diferencia del escenario aislado donde los límites temporales de cada señal están dados, en este sistema los segmentos predichos engloban múltiples señales reales, ofreciendo al clasificador entradas sustancialmente distintas a aquellas sobre las que fue

entrenado. El IoU General en este nivel es bajo para ambos modelos (2,20 % y 2,29 %), reflejando esta combinación de errores de clasificación y delimitación temporal.

Reconocimiento Top-1 (Nivel 3)

En el nivel más exigente, ambas arquitecturas convergen a un desempeño similar: F_1 de 3,12 % para SPOTER y 3,11 % para MSG3D, con 12 verdaderos positivos cada uno sobre las 730 señas del *ground truth*. Las precisiones son del 30,77 % y 28,57 % respectivamente. Sin embargo, el sistema solo logra reconocer el 1,64 % del total de señas.

La convergencia de ambos modelos en este nivel sugiere que las diferencias arquitectónicas se diluyen frente a la dificultad compartida: tanto el rendimiento limitado de los clasificadores como la naturaleza de los segmentos (que engloban múltiples señas simultáneamente) imponen un tope de desempeño que afecta a ambas arquitecturas.

Comparación entre arquitecturas

Los resultados permiten identificar perfiles diferenciados para cada modelo:

- **SPOTER** presenta una ventaja notable en la precisión de sus predicciones Top-5, lo que sugiere que su mecanismo de atención genera conjuntos de candidatos más confiables. Esto se refleja en un F_1 de Nivel 2 casi el doble que MSG3D.
- **MSG3D** genera más segmentos predichos (38 vs. 31) y obtiene un IoU General levemente superior en todos los niveles, lo que indica una mayor sensibilidad en la detección de cambios temporales. Sin embargo, esta mayor sensibilidad no se traduce en mejor reconocimiento.

En el Nivel 3, la convergencia indica que las diferencias arquitectónicas impactan principalmente en la fase de selección de candidatos (Top-5), pero no en la clasificación final (Top-1).

Cabe señalar que esta ventaja de SPOTER en Top-5 contrasta con su rendimiento inferior en reconocimiento aislado, donde MSG3D lo supera tanto en Top-1 como en Top-5 (Tabla 4.9). Este mismo patrón se observa en el enfoque basado en segmentación, donde SPOTER también supera a MSG3D en Top-5 a pesar de su desventaja en reconocimiento aislado. La consistencia de este comportamiento en ambos enfoques continuos sugiere que la capacidad de SPOTER para ubicar la clase correcta entre sus primeros candidatos se degrada de forma más gradual que la de MSG3D al operar sobre segmentos imperfectos. En Top-1, donde se requiere acertar la clase exacta, ambos modelos se ven igualmente limitados y la ventaja aislada de MSG3D no logra manifestarse.

Un análisis complementario mediante métricas adicionales – incluyendo ratios por *frame*, impacto del filtro de suavizado y evaluación restringida a señas del vocabulario – se presenta en el Anexo B.2. Entre los hallazgos principales, se

destaca que el *Ratio de Clase Condicional* en configuración Top-5 supera el 88 % para ambos modelos, indicando que cuando el sistema acierta temporalmente en un *frame*, la clase correcta suele figurar entre los cinco primeros candidatos. Asimismo, el filtro de suavizado tiene un impacto despreciable en las métricas finales a pesar de realizar aproximadamente 47–48 correcciones por video.

5.4.4. Comparación con el enfoque basado en segmentación

Para contextualizar los resultados obtenidos, se los compara con los del enfoque basado en el modelo de segmentación presentado en la sección anterior. Ambas tablas reportan métricas sobre la totalidad de las 730 señas del *ground truth*.

Tabla 5.4: Comparación entre el enfoque de segmentación y el de ventana deslizante.

Nivel / Modelo	Segmentador		Ventana Deslizante	
	F ₁	IoU G.	F ₁	IoU G.
Nivel 1 – MSG3D	95,86 %	43,44 %	99,79 %	10,05 %
Nivel 1 – SPOTER	95,86 %	43,44 %	99,66 %	8,72 %
Nivel 2 – MSG3D	13,53 %	6,87 %	6,44 %	2,29 %
Nivel 2 – SPOTER	15,61 %	7,42 %	12,06 %	2,20 %
Nivel 3 – MSG3D	7,07 %	3,89 %	3,11 %	1,40 %
Nivel 3 – SPOTER	6,84 %	3,18 %	3,12 %	0,32 %

La comparación revela diferencias sustanciales entre ambos enfoques, tanto en sus fortalezas como en sus limitaciones.

Detección temporal

En el Nivel 1, ambos enfoques exhiben perfiles muy distintos. La ventana deslizante alcanza un F_1 superior (99,7 % vs. 95,9 %), aunque este valor era esperable dado el diseño del enfoque (como se discutió en la sección anterior): al agrupar predicciones consecutivas sobre todo el video, prácticamente cualquier seña queda cubierta por algún segmento predicho. En cuanto al IoU General, la diferencia se invierte de forma marcada (9–10 % para la ventana deslizante vs. 43–44 % para el segmentador). Esta divergencia refleja la diferencia conceptual entre los enfoques: el segmentador identifica los límites de cada seña individual, produciendo segmentos acotados con buen ajuste temporal; la ventana deslizante barre el video completo y agrupa predicciones, generando pocos segmentos amplios que cubren la totalidad de la secuencia pero con fronteras poco alineadas a las señas individuales.

Reconocimiento

En los niveles de reconocimiento, el segmentador presenta ventaja en la mayoría de las configuraciones. En el Nivel 2, el segmentador obtiene mejores resultados para MSG3D (13,53 % vs. 6,44 %) y también para SPOTER (15,61 % vs. 12,06 %), aunque la brecha es menor en este último caso. En el Nivel 3, la ventaja del segmentador es clara para ambos modelos.

En cuanto al IoU General, el segmentador supera ampliamente a la ventana deslizante en todos los niveles, lo que refleja la mayor calidad de sus segmentos individuales.

Síntesis

Cada enfoque presenta fortalezas en aspectos distintos del problema:

- El **segmentador** produce segmentos temporales mejor delimitados (IoU superior) y obtiene resultados de clasificación más consistentes, particularmente en el Nivel 3. Su fortaleza radica en la calidad de los segmentos individuales, que al corresponder más fielmente a señas aisladas, facilitan la tarea del clasificador.
- La **ventana deslizante** garantiza una cobertura temporal casi total y no requiere un modelo de segmentación adicional, lo que simplifica el *pipeline*. Sin embargo, como se discutió en la Sección 5.4.3, su capacidad de delimitar señas individuales se ve limitada por el bajo rendimiento de los modelos en la predicción de la clase *no-seña*, lo que se traduce en un IoU bajo y restringe la calidad de la clasificación.

Estos resultados sugieren que la segmentación explícita del video aporta información valiosa para el reconocimiento, ya que alimenta al clasificador con segmentos que corresponden más fielmente a señas individuales. No obstante, el enfoque de ventana deslizante demuestra que es posible obtener resultados no triviales sin depender de un modelo de segmentación externo, lo cual resulta relevante en escenarios donde no se dispone de un segmentador entrenado o donde se busca reducir la complejidad del sistema.

Capítulo 6

Conclusiones y Trabajo Futuro

6.1. Resumen de objetivos cumplidos

El proyecto concluye con el cumplimiento de los principales objetivos planteados. A partir de un conjunto de datos crudos, se logró la conformación de un *dataset* curado y adecuado para el entrenamiento de modelos de aprendizaje automático. Este conjunto de datos permitió la experimentación con distintos enfoques y arquitecturas, posibilitando la comparación de fortalezas y debilidades entre los diferentes sistemas desarrollados.

Por un lado, se trabajó con dos arquitecturas distintas de modelos de aprendizaje profundo para el reconocimiento aislado de Señas: el modelo SPOTER, basado en *transformers*, y el modelo MSG3D, una red neuronal convolucional.

Una vez obtenidos los mejores modelos para cada arquitectura, estos fueron utilizados como base para los sistemas de *sign spotting* evaluados en la etapa posterior. Se implementaron dos enfoques: un *pipeline* que primero segmenta el video mediante un modelo preentrenado y luego predice la señal correspondiente a cada segmento, y un enfoque de ventana deslizante, en el cual se realizan predicciones sobre intervalos de largo fijo del video, seguidas de una etapa de postprocesamiento para obtener el resultado final.

Si bien el rendimiento alcanzado por los modelos para la tarea de reconocimiento de señas de LSU aún no es suficiente para su aplicación en un entorno real, la experimentación con distintos enfoques, arquitecturas y parámetros permitió obtener *insights* valiosos y sentar bases sólidas para futuras investigaciones en el área del Reconocimiento Automático de Lengua de Señas Uruguaya.

6.2. Descubrimientos

Como principal descubrimiento, se destaca un comportamiento observado durante la experimentación con el sistema que utiliza un modelo segmentador: se evidenció un mejor rendimiento del modelo basado en la arquitectura MSG3D en términos de predicción exacta de señas, en contraste con la fortaleza del modelo SPOTER en cuanto a cobertura *Top-5*. Este patrón se mantuvo de forma consistente en todas las variantes evaluadas.

Otro hallazgo relevante fue el bajo rendimiento de las pruebas de *transfer learning* utilizando pesos preentrenados sobre otras lenguas de señas. Tanto el experimento de entrenamiento del modelo MSG3D a partir de pesos obtenidos con un *dataset* de Lengua de Señas Española (SWL-LSE), como la prueba de *fine-tuning* del modelo segmentador, no lograron mejorar los resultados respecto al modelo entrenado desde cero y al modelo base, respectivamente. Esto podría estar indicando la existencia de diferencias lingüísticas significativas entre la Lengua de Señas Uruguaya y las lenguas utilizadas para el preentrenamiento, así como posibles discordancias en el formato de los videos o en la estructura de las señas.

Por último, los resultados de la experimentación confirmaron que el problema de *sign spotting* es sustancialmente más complejo que el de reconocimiento aislado, tanto desde el punto de vista del modelado, de la evaluación y los resultados obtenidos.

6.3. Dificultades encontradas

Las principales dificultades encontradas durante el desarrollo del proyecto estuvieron relacionadas, principalmente, con la calidad y características de los datos de entrada. Por un lado, en algunos casos la calidad de imagen de los videos no era óptima, lo que implicó la necesidad de descartar un video con una cantidad considerable de glosas. Por otro lado, al momento de iniciar el proyecto, el proceso de glosado se encontraba en una etapa inicial, por lo que las anotaciones disponibles constituían una versión tentativa, presentando posibles errores y una falta de homogeneidad en la notación utilizada.

Adicionalmente, las distintas disposiciones de *keypoints* requeridas por los modelos empleados implicaron la necesidad de mantener múltiples versiones del *dataset*, lo que complejizó significativamente la etapa de procesamiento y gestión de los datos.

6.4. Trabajo futuro

En esta sección se detallan los principales aspectos del proyecto en los cuales se identificaron oportunidades de mejora y posibles líneas de trabajo a desarrollar en el futuro.

6.4.1. Mejoras en el dataset

El primer punto que se desprende directamente de lo expuesto anteriormente es la posibilidad de realizar un nuevo entrenamiento de los modelos una vez que los videos glosados alcancen su versión definitiva. De este modo, al contar con un *dataset* más grande, curado y estandarizado, es esperable obtener una mejora significativa en el rendimiento de los sistemas desarrollados.

6.4.2. Optimización de hiperparámetros

A lo largo de los experimentos de reconocimiento aislado, se utilizó un *learning rate* fijo de 0,001 para ambas arquitecturas, siguiendo los valores reportados en los trabajos originales. Una línea de mejora directa consistiría en explorar estrategias de ajuste del *learning rate*, como pueden ser *schedulers* o la búsqueda sistemática de hiperparámetros mediante técnicas como *grid search*. Dado que el *dataset* LSU presenta características distintas a los conjuntos de datos utilizados en los trabajos de referencia (señas más cortas, mayor desbalance, datos espontáneos), es esperable que los valores óptimos de hiperparámetros difieran de los reportados originalmente.

Más allá del *learning rate*, se identificaron en el transcurso del proyecto otros parámetros de configuración que en este trabajo se fijaron a partir de criterios heurísticos o de simplicidad operativa, pero que admiten ser tratados como hiperparámetros sujetos a optimización. En particular, en el enfoque de ventana deslizante (Capítulo 5) se identificaron tres líneas concretas que se dejan como trabajo futuro:

- **Filtro de post-procesado.** El filtro de mediana adoptado para suavizar las predicciones por *frame* fue implementado con un comportamiento fijo (vinculado a la relación $W = 3S$). Explorar variantes con tamaño configurable o esquemas de suavizado alternativos podría mejorar la calidad de la partición resultante.
- **Configuración de la ventana (W y S).** En este trabajo se evaluaron únicamente dos configuraciones que cumplen $W = 3S$. Eliminando esta restricción, podría experimentarse con un espacio mayor de pares (W, S) , lo que permitiría una caracterización más completa del enfoque.
- **Umbral de predicción para la clase NO-SEÑA.** El bajo rendimiento del sistema en la predicción de la clase *no-seña* (Sección 4.1.4) limita la capacidad de la ventana deslizante para generar intervalos entre señas. Un ajuste del umbral o la definición de un criterio que favorezca su selección como Top-1 en casos ambiguos podría obtener mejores resultados sin necesidad de reentrenar los modelos.

6.4.3. Experimentación con nuevos modelos

Por otro lado, se abre la posibilidad de incursionar en experimentos con diferentes arquitecturas de aprendizaje profundo. Algunos ejemplos incluyen

variantes híbridas CNN-Transformer aplicadas al reconocimiento de lenguas de señas en escenarios dinámicos (con señantes en movimiento, transiciones suaves entre señas y variabilidad significativa entre intérpretes) (Hashi y cols., 2025), así como enfoques unificados de detección y clasificación de acciones en video, como YOLO-Act (Alzahrani, Bchir, y Ben Ismail, 2025).

6.4.4. Aplicaciones al mundo real

En caso de alcanzarse niveles de *performance* adecuados, un paso lógico sería el desarrollo de un sistema integral que permita a un usuario procesar un video con un señante y obtener como salida el glosado automático de las señas realizadas, constituyendo una herramienta de apoyo con potencial impacto social. No obstante, para que una herramienta de este tipo resulte efectivamente útil, sería imprescindible complementar la evaluación técnica con pruebas de usabilidad que involucren a usuarios finales, incluyendo intérpretes de LSU y miembros de la comunidad sorda e hipoacúsica del Uruguay. Este tipo de evaluación permitiría validar no solo la precisión del sistema, sino también su nivel de adopción en contextos reales de uso.

Referencias

- Alzahrani, N., Behir, O., y Ben Ismail, M. M. (2025). Yolo-act: Unified spatio-temporal detection of human actions across multi-frame sequences. *Sensors*, 25(10), 3013. doi: 10.3390/s25103013
- Amit Moryossef, M. M. S. E. Y. G., Zifan Jiang. (2023). Linguistically motivated sign language segmentation. En *Findings of the association for computational linguistics: Emnlp 2023* (p. 12703–12724).
- Boháček, M., y Hruží, M. (2022, January). Sign pose-based transformer for word-level sign language recognition. En *Proceedings of the ieee/cvf winter conference on applications of computer vision (wacv) workshops* (p. 182–191).
- Bradski, G., y Kaehler, A. (s.f.). Opencv: Open source computer vision library. *GitHub repository*. <https://github.com/opencv/opencv>.
- Camgoz, N. C., Koller, O., Hadfield, S., y Bowden, R. (2020). Sign language transformers: Joint end-to-end sign language recognition and translation. En *Proceedings of the ieee/cvf conference on computer vision and pattern recognition*.
- Cao, Z., Hidalgo, G., Simon, T., Wei, S.-E., y 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. doi: 10.1109/TPAMI.2019.2929257
- Carreira, J., y Zisserman, A. (2017). Quo vadis, action recognition? a new model and the kinetics dataset. En *Proceedings of the ieee conference on computer vision and pattern recognition (cvpr)*.
- FFmpeg Developers. (2000). *Ffmpeg*. <https://ffmpeg.org/>. (Software library for multimedia processing)
- Fojo, A., Tancredi, M., Etcheverry, F., Pintos, A., Pérez, B., Bianchi, J., ... Chiruzzo, L. (2024). Corpus de lengua de señas uruguaya y desarrollo de metadatos. En *Congreso internacional de pesquisas em língua de sinais*.
- Google. (2019). *Guía de tareas de detección de puntos de referencia holísticos*. https://ai.google.dev/edge/mediapipe/solutions/vision/holistic_landmarker?hl=es-419.
- Hashi, A. O., Mohd Hashim, S. Z., Mirjalili, S., Kebande, V. R., Al-Dhaqm, A., Nasser, M., y Samah, A. B. A. (2025). A hybrid cnn-transformer

- framework optimized by grey wolf algorithm for accurate sign language recognition. *Scientific Reports*, 15. doi: 10.1038/s41598-025-27390-2
- Hochreiter, S., y Schmidhuber, J. (1997). Long short-term memory. *Neural Computation*, 9(8), 1735–1780. doi: 10.1162/neco.1997.9.8.1735
- Li, D., Opazo, C. R., Yu, X., y Li, H. (2020). *Word-level deep sign language recognition from video: A new large-scale dataset and methods comparison*. Descargado de <https://arxiv.org/abs/1910.11006>
- Miah, A. S. M., Hasan, M. A., Okuyama, Y., Tomioka, Y., y Shin, J. (2024, 04). Spatial-temporal attention with graph and general neural network-based sign language recognition. *Pattern Analysis and Applications*, 27. doi: 10.1007/s10044-024-01229-4
- Ministerio de Salud Pública de Uruguay. (2023). *Personas sordas e hipoacusia por primera vez en el ministerio de salud pública*. Descargado de <https://www.gub.uy/ministerio-salud-publica/comunicacion/noticias/personas-sordas-hipoacusia-primera-vez-ministerio-salud-publica> (Accedido: 9 mayo 2026)
- Moryossef, A., Yin, K., Neubig, G., y Goldberg, Y. (2021). Data augmentation for sign language gloss translation. En *Proceedings of the 1st international workshop on automatic translation for signed and spoken languages (at4ssl)* (pp. 1–11).
- Naiara Garmendia, Horacio Saggion y Euan McGill. (2025a). Spontaneous catalan sign language recognition: Data acquisition and classification. *aclanthology*, 1(1).
- Naiara Garmendia, Horacio Saggion y Euan McGill. (2025b). *Spontaneous-lsc-recognition*. <https://github.com/LaSTUS-TALN-UPF/Spontaneous-LSC-Recognition>. GitHub.
- Ronchetti, F., Quiroga, F. M., Estrebou, C., Lanzarini, L., y Rosete, A. (2023). *Lsa64: An argentinian sign language dataset*. Descargado de <https://arxiv.org/abs/2310.17429>
- Shen, X., Yuan, S., Sheng, H., Du, H., y Yu, X. (2023). Auslan-daily: australian sign language translation for daily communication and news. En *Proceedings of the 37th international conference on neural information processing systems*. Red Hook, NY, USA: Curran Associates Inc.
- Shorten, C., y Khoshgoftaar, T. M. (2019). A survey on image data augmentation for deep learning. *Journal of Big Data*, 6(1), 60. doi: 10.1186/s40537-019-0197-0
- Tran, D., Bourdev, L., Fergus, R., Torresani, L., y Paluri, M. (2015). Learning spatiotemporal features with 3d convolutional networks. En *Proceedings of the ieee international conference on computer vision (iccv)*.
- Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., ... Polosukhin, I. (2017). Attention is all you need. En *Proceedings of the 31st international conference on neural information processing systems* (p. 6000–6010). Red Hook, NY, USA: Curran Associates Inc.
- Wittenburg, P., Brugman, H., Russel, A., Klassmann, A., y Sloetjes, H. (2006). Elan: A professional framework for multimodality research. En *5th inter-*

- national conference on language resources and evaluation (lrec 2006)* (pp. 1556–1559).
- Yan, S., Xiong, Y., y Lin, D. (2018). Spatial temporal graph convolutional networks for skeleton-based action recognition. En *Proceedings of the aaai conference on artificial intelligence*.
- Zhang, W., Deng, L., y Wu, D. (2020). A survey on negative transfer. *arXiv preprint arXiv:2009.00909*. doi: 10.48550/arXiv.2009.00909
- Zhang, X., y Duh, K. (2021). Approaching sign language gloss translation as a low-resource machine translation task. En *Proceedings of the 1st international workshop on automatic translation for signed and spoken languages (at4ssl)* (pp. 60–70).
- Ziyu Liu, Hongwen Zhang, Zhenghao Chen, Zhiyong Wang, Wanli Ouyang. (2020). Disentangling and unifying graph convolutions for skeleton-based action recognition. *arxiv*, 1(1).
- Zuo, R., Wei, F., y Mak, B. (2024, noviembre). Towards online continuous sign language recognition and translation. En Y. Al-Onaizan, M. Bansal, y Y.-N. Chen (Eds.), *Proceedings of the 2024 conference on empirical methods in natural language processing* (pp. 11050–11067). Miami, Florida, USA: Association for Computational Linguistics. Descargado de <https://aclanthology.org/2024.emnlp-main.619/> doi: 10.18653/v1/2024.emnlp-main.619

Anexo A

Detalles complementarios de arquitecturas y modelos

A.1. Detalle de SPOTER

A.1.1. Datasets utilizados

El rendimiento del modelo SPOTER fue medido sobre dos *datasets* mundialmente reconocidos.

Dataset WLASL El dataset WLASL (*Word-Level American Sign Language*) (Li y cols., 2020) es una colección de videos de Lengua de Señas Americana (ASL) recopilados de recursos públicos educativos en línea. Cada video contiene exactamente una seña aislada, lo que lo convierte en un *dataset* orientado al reconocimiento aislado (*isolated sign language recognition*). Contiene 21.083 videos distribuidos en 2.000 glosas (clases de señas), con un promedio de 10,5 instancias por glosa y 119 señantes en total. Los videos presentan variabilidad en estilos de señas, dialectos, fondos y condiciones de grabación, ya que fueron interpretados por señantes nativos o intérpretes profesionales. Para evaluación, los autores crearon subconjuntos como WLASL100 (100 glosas, 2.038 videos, promedio de 20,4 instancias por glosa, 97 señantes), WLASL300 (300 glosas, 5.117 videos, promedio de 17,1, 109 señantes), WLASL1000 y WLASL2000, permitiendo benchmarks escalables.

Dataset LSA64 El dataset LSA64 (Ronchetti, Quiroga, Estrebou, Lanzarini, y Rosete, 2023), publicado en 2016, es un conjunto de datos para la Lengua de Señas Argentina (LSA) que incluye 3,200 videos de 64 glosas seleccionadas entre las más comunes del léxico LSA, abarcando verbos y sustantivos. Cada glosa cuenta con 50 instancias, interpretadas por 10 sujetos no expertos (5 hombres y 5 mujeres), lo que introduce variabilidad en la ejecución. Los videos fueron grabados en RGB con guantes de colores para facilitar la segmentación

de manos. Este dataset es más pequeño y controlado que WLASL, enfocado en señas aisladas, y se utiliza comúnmente para benchmarks en lenguas de señas latinoamericanas, logrando altas precisiones en modelos eficientes.

A.1.2. Arquitectura detallada

La arquitectura de SPOTER está basada en el Transformer propuesto por Vaswani et al. (Vaswani y cols., 2017), adaptado al procesamiento de secuencias temporales de poses humanas. Cada instancia de una seña se representa como una secuencia de vectores, donde cada vector corresponde a un *frame* y contiene las coordenadas 2D concatenadas de 54 *keypoints* (cuerpo, manos y rostro) extraídos utilizando Vision API.

Estas representaciones se proyectan mediante una capa lineal a un espacio de dimensión fija 108 donde se agregan codificaciones posicionales temporales, lo que permite al modelo distinguir el orden y la posición relativa de cada *frame* dentro de la seña. Posteriormente, la secuencia es procesada por una pila de capas Transformer que constan de 6 capas de codificador y 6 capas de decodificador, con 9 cabezas de atención en cada módulo de atención multi-cabeza.

Como puede visualizarse en la Figura A.1, el decodificador utiliza una consulta única, denominada *Class Query*, que se proyecta a través de un módulo de proyección multi-cabeza antes de integrarse con la salida del codificador. Finalmente, esta consulta decodificada se pasa por una capa lineal con activación softmax para predecir la clase de la seña. Esta configuración resulta en un modelo liviano con aproximadamente 5.92 millones de parámetros, optimizado para secuencias de poses y enfocado en eficiencia computacional.

A.1.3. Preprocesamiento de *keypoints*

Además de la arquitectura, uno de los pilares fundamentales del rendimiento de SPOTER es su pipeline de preprocesamiento geométrico, diseñado específicamente para mejorar la robustez del modelo ante cambios de escala, posición o perspectiva de la imagen, o del intérprete realizando las señas.

Técnicas de normalización Una de las contribuciones más significativas de SPOTER es su esquema de normalización robusto, el cual se basa en transformar la anatomía de cada intérprete a proporciones corporales universales. Para esto, se utiliza la altura de la cabeza (estimada como la mitad de la distancia entre los hombros) como unidad de referencia, y se escalan las poses de manera que el espacio de señas se delimite en 6 unidades de cabeza de ancho (centradas en la nariz) y 7 unidades de alto (desde 0.5 unidades por encima del ojo izquierdo hasta 6 unidades por debajo). De esta forma, se logra que el modelo se centre en el contenido semántico de la seña independientemente de la distancia a la cámara o las proporciones corporales del individuo.

Adicionalmente, las manos se normalizan en sus propios sistemas de coordenadas locales, permitiendo al modelo aprender la forma de las manos separada-

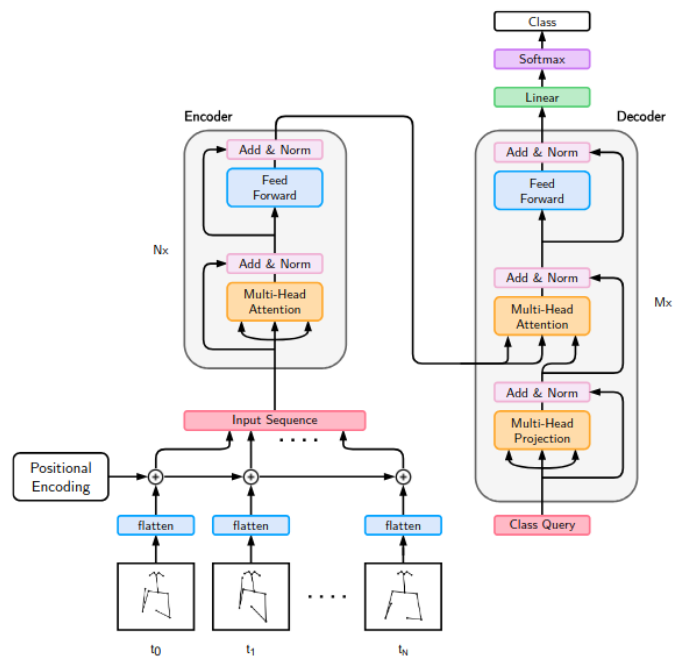


Figura A.1: La arquitectura del modelo SPOTER basada en Transformers. Obtenido de (Boh  cek y Hr  z, 2022).

mente de su ubicación espacial. Estas coordenadas locales se obtienen mediante cajas delimitadoras mínimas que engloban los 21 *keypoints* de cada mano, con un margen de al menos el 10 % del tamaño de la caja para evitar distorsiones. Finalmente, todas las coordenadas normalizadas se desplazan por $[-0.5, -0.5]$ para centrar la media en 0 y lograr una dispersión de 1, lo que optimiza el aprendizaje del modelo.

En el *dataset* WLASL-100, estas técnicas de normalización supusieron una mejora sustancial del 14.01 % en la precisión del modelo, como puede observarse en la Tabla A.1.

Técnicas de aumentado Para prevenir el sobreajuste y mejorar la capacidad de generalización del modelo, SPOTER implementa un conjunto de aumentados espaciales aplicados sobre los datos esqueléticos durante el entrenamiento. Los parámetros de los aumentados se seleccionan aleatoriamente pero se mantienen consistentes para todos los *frames* dentro de una misma instancia de señal, preservando la coherencia temporal del gesto.

Las técnicas de aumentado implementadas son:

- Rotación en el plano: Rota todas las articulaciones hasta 13 grados, simulando variaciones en la orientación del intérprete.
- Compresión lateral: Reduce el ancho del *frame* hasta 15 % desde ambos lados simulando diferentes encuadres.
- Transformación de perspectiva: Proyecta las coordenadas sobre un plano inclinado replicando grabaciones con cámara no perpendicular.
- Rotación secuencial de articulaciones: Cada articulación del brazo se rota ligeramente (hasta ± 4 grados con probabilidad 3:10) respecto a la anterior, simulando las micro-diferencias naturales en cada ejecución de una señal sin alterar su significado semántico.

Las mismas pueden apreciarse en un ejemplo de un *frame* en la Figura A.2.

El estudio de ablación sobre el dataset WLASL-100 reveló que cada aumentado aporta mejoras individuales de 1-2 % en precisión, y que su combinación junto con ruido gaussiano mínimo significó una mejora de 4.21 % sobre los datos normalizados sin aumentar, como puede visualizarse en la Tabla A.1.

A.1.4. Resultados y performance

SPOTER estableció en 2022 el estado del arte en reconocimiento basado en poses para los subconjuntos WLASL-100 (63.18 % de precisión top-1) y WLASL-300 (43.78 %), como puede observarse en la Figura A.3. También obtuvo la mejor precisión en el dataset LSA64 (100 %). En comparación con enfoques basados en *features*, como I3D (que alcanza 66.18 % en WLASL-100), SPOTER cierra la brecha con un costo computacional mucho menor: 5.92 millones de parámetros frente a 12.35 millones de I3D, y un tiempo de inferencia promedio de 0.05

segundos por video (incluyendo estimación de poses) contra 0.55 segundos para I3D. Además, requiere solo 1.42 GFLOPs por inferencia, en contraste con los 5.22 GFLOPs de I3D.

El estudio de ablación confirma la robustez del modelo, mientras que experimentos con *datasets* reducidos demuestran su capacidad para generalizar con pocos datos: en LSA64, SPOTER alcanza 88.68 % de precisión con solo el 10 % del conjunto de entrenamiento, superando ampliamente a I3D (45.47 %). Estas características lo hacen ideal para dispositivos móviles y escenarios con recursos limitados.

A.2. Fundamentos de Redes Convolucionales de Grafos

A.2.1. Representación del cuerpo como grafo

En el contexto de lenguas de señas, el cuerpo del intérprete se modela como un grafo no dirigido $G = (V, E)$. Los nodos V representan las articulaciones o puntos clave (muñecas, codos, hombros, dedos) extraídos por herramientas como MediaPipe. Las aristas E representan las conexiones físicas (huesos) entre dichas articulaciones.

A.2.2. Convolución en grafos

A diferencia de las Redes Convolucionales (CNN) tradicionales que operan sobre estructuras regulares (como una cuadrícula de píxeles), en un grafo la vecindad es irregular y se define por las conexiones de cada nodo.

El funcionamiento de una GCN se basa en el mecanismo de paso de mensajes (*Message Passing*), con tres pasos iterativos:

1. **Agregación:** Cada nodo recolecta las características de sus nodos vecinos.
2. **Actualización:** La información recolectada se combina con la propia información del nodo mediante una transformación lineal (pesos aprendibles).
3. **Activación:** Se aplica una función no lineal (como ReLU) para obtener la nueva representación del nodo.

A.2.3. Representación matemática

Para procesar el grafo de manera eficiente, la estructura se representa mediante una Matriz de Adyacencia A , que indica qué nodos están conectados, y una Matriz de Características H , que contiene los datos de cada nodo.

La operación de una capa de GCN se expresa como:

$$H^{(l+1)} = \sigma \left(\tilde{D}^{-\frac{1}{2}} \tilde{A} \tilde{D}^{-\frac{1}{2}} H^{(l)} W^{(l)} \right)$$

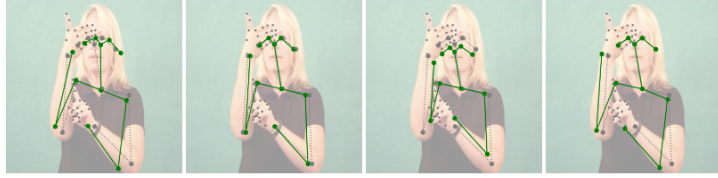


Figura A.2: Un ejemplo de los aumentados de SPOTER en un *frame*. De izquierda a derecha: rotación en el plano, compresión lateral, transformación de perspectiva, rotación de articulaciones. Obtenido de (Boh  cek y Hr  z, 2022).

Tabla A.1: Resultados del estudio de ablaci  n en el subconjunto WLASL-100. Obtenido de (Boh  cek y Hr  z, 2022).

Model	Norm.	Aug.	Accuracy
A	�	�	44.96
B	�	�	58.97
C	�	Rotate	61.24
D	�	Squeeze	60.85
E	�	Perspective t.	60.47
F	�	Arm joint rotate	61.24
G	�	All	62.79
H	�	All + Gaussian noise	63.18

Model	App.	Pose	Backbone	WLASL100	WLASL300
I3D (baseline) [21]	�	�	�	65.89	56.14
TK-3D ConvNet [22]	�	�	�	77.55	68.75
Fusion-3 [13]	�	�	�	75.67	68.30
GCN-BERT [36]	�	�	�	60.15	42.18
Pose-TGCN [21]	�	�	�	55.43	38.32
Pose-GRU [21]	�	�	�	46.51	33.68
SPOTER (Ours)	�	�	�	63.18	43.78

Figura A.3: Resultados de Top-1 Macro precisi  n de SPOTER comparado a otros modelos. Obtenido de (Boh  cek y Hr  z, 2022).

Donde:

- $\tilde{A} = A + I$ es la matriz de adyacencia con auto-conexiones (para que el nodo no olvide su propia información)
- $\tilde{D}$ es la matriz de grado, utilizada para normalizar los datos (evitando que los nodos con muchas conexiones tengan valores desproporcionadamente altos)
- $W^{(l)}$ es la matriz de pesos que la red aprende durante el entrenamiento
- σ es la función de activación (ej. ReLU)

A.3. Arquitectura MSG3D

A.3.1. El problema de las redes previas

Los enfoques tradicionales tratan los datos de esqueletos como una secuencia de grafos espaciales independientes, alternando entre capas de convolución espacial (GCN) y capas temporales (TCN). Esto crea un “cuello de botella” en el flujo de información, ya que las relaciones complejas que ocurren simultáneamente en el espacio y el tiempo (ej. mover la mano derecha mientras se avanza el pie izquierdo) no se capturan directamente hasta capas muy profundas.

A.3.2. El módulo G3D: unificación espacio-temporal

La innovación central de MSG3D es el operador G3D (*Graph 3D*). En lugar de ver el video como “fotogramas separados”, G3D construye un grafo unificado en el dominio espacio-temporal (Ver Figura A.4).

- **Grafo Espacio-Temporal Denso:** El modelo crea aristas no solo entre articulaciones dentro del mismo *frame* (espacial) y la misma articulación en *frames* consecutivos (temporal), sino también conexiones “cruzadas” directas a través del espacio-tiempo.
- **Ventana Deslizante Temporal:** Se aplica sobre una ventana de *frames*, permitiendo que la información fluya instantáneamente entre, por ejemplo, la mano en el tiempo t y la cabeza en el tiempo $t + 1$.

A.3.3. Agregación multi-escala desentrelazada

En el reconocimiento de señas, el modelo debe capturar relaciones a diferentes distancias: desde escalas locales (relación entre dedos y muñeca) hasta escalas globales (relación entre manos y cabeza).

El problema: ponderación sesgada (*Biased weighting problem*) En una GCN estándar, para obtener información de un vecino a distancia 2, se eleva la matriz de adyacencia al cuadrado (A^2). Esto causa redundancia: A^2 incluye no solo vecinos a distancia 2, sino también vecinos de 1 paso y la identidad del propio nodo. Al sumar estas potencias, la información de nodos cercanos se agrega múltiples veces, “ahogando” la señal de nodos lejanos.

La solución: matrices de adyacencia por distancia $A_{(k)}$ MSG3D propone desentrelazar las escalas usando matrices que aíslan cada vecindario:

$$\tilde{A}_{(k)}[i, j] = 1 \quad \text{si y solo si} \quad d(v_i, v_j) = k$$

Donde $d(v_i, v_j)$ es la distancia de camino más corto entre nodos.

De esta forma se generan canales separados para cada escala:

1. Escala 1: Solo articulaciones adyacentes (ej. muñeca-codo)
2. Escala 2: Solo información a 2 pasos (ej. muñeca-hombro)
3. Escala k : Solo información a k pasos

Las características se extraen independientemente para cada escala, se concatenan (no se suman), y luego se fusionan mediante una capa adicional.

A.3.4. Bloque STGC y arquitectura general

La arquitectura final de MSG3D se organiza como una pila de r bloques STGC (*Spatio-Temporal Graph Convolution*), seguidos de una capa de *Global Average Pooling* y un clasificador *Softmax*.

Cada bloque STGC tiene dos vías complementarias:

- **Vía G3D:** Construye ventanas espacio-temporales, aplica convoluciones multiescala y utiliza una capa totalmente conectada para extraer características.
- **Vía Factorizada:** Incluye un módulo espacial de largo alcance (GCN multiescala) y un módulo temporal de largo alcance (MS-TCN) para capturar dependencias extendidas.

A.4. Detalle del trabajo en Lengua de Signos Catalana

A.4.1. Adquisición y naturaleza de los datos

A diferencia de la mayoría de los conjuntos de datos existentes que utilizan signos aislados y preparados, este trabajo se fundamenta en un corpus de

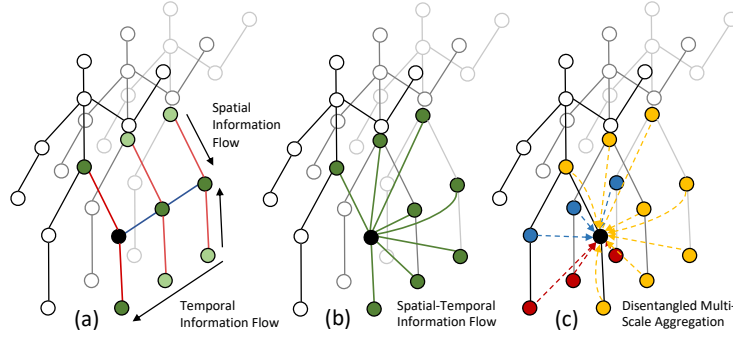


Figura A.4: Comparación entre el enfoque factorizado tradicional (a) y el módulo G3D (c), que introduce aristas directas espacio-temporales. Imagen extraída de (Ziyu Liu, Hongwen Zhang, Zhenghao Chen, Zhiyong Wang, Wanli Ouyang, 2020).

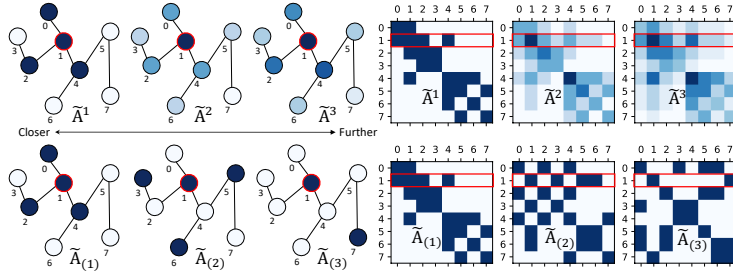


Figura A.5: Ilustración del problema de ponderación sesgada y del esquema de agregación desentrelazada. Imagen extraída de (Ziyu Liu, Hongwen Zhang, Zhenghao Chen, Zhiyong Wang, Wanli Ouyang, 2020).

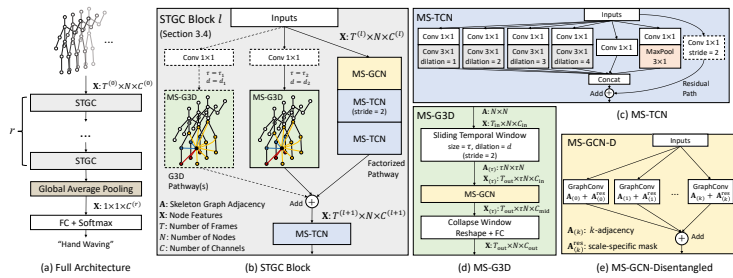


Figura A.6: Arquitectura general de MSG3D con bloques STGC. Imagen extraída de (Ziyu Liu, Hongwen Zhang, Zhenghao Chen, Zhiyong Wang, Wanli Ouyang, 2020).

diálogos y monólogos espontáneos. Se utilizó el *LSC Corpus*, un recurso de referencia que incluye grabaciones de tareas de elicitación y conversaciones libres, capturando la diversidad lingüística, geográfica y generacional de los señantes. Se recolectaron un total de 45,587 videos correspondientes a 6,527 glosas diferentes.

Al provenir de conversaciones continuas, los datos presentan desafíos únicos: los señantes no siempre miran de frente a la cámara y la producción del signo está influenciada por la coarticulación (efecto de los signos precedentes y siguientes).

A.4.2. Preprocesamiento y curación del dataset

Para entrenar un modelo robusto, los autores aplicaron un proceso riguroso de extracción y filtrado:

- **Segmentación Temporal:** Utilizando anotaciones manuales en software ELAN, se marcaron los puntos exactos de inicio y fin de cada signo dentro del flujo continuo de la conversación para extraerlos como instancias aisladas.
- **Agrupación de Clases (Normalización):** Una misma glosa (ej. “AGAFAR” - agarrar) puede tener múltiples variaciones visuales según conjugación, posición de la mano o contexto gramatical. El estudio agrupó estas variaciones bajo lemas principales, reduciendo las 6,527 glosas iniciales a 1,885 clases principales.
- **Filtrado por Frecuencia:** Observaron que la distribución de datos era desbalanceada (muchas glosas con pocas muestras). Para garantizar un aprendizaje efectivo, seleccionaron únicamente las glosas con más de 50 muestras, resultando en 105 clases para entrenamiento.
- **Extracción de Features:** Se procesaron los videos utilizando MediaPipe Holistic (Google, 2019), convirtiendo la información visual en *keypoints* del cuerpo, manos y rostro.

A.4.3. Experimentos y resultados completos

El estudio evaluó dos estrategias principales: entrenamiento desde cero (*from scratch*) y *transfer learning* utilizando un modelo pre-entrenado con el *dataset* SWL-LSE (Lengua de Señas Española). Los resultados completos se muestran en la Tabla A.2.

A.4.4. Contribución general

Este trabajo representa la primera investigación en reconocimiento de signos aislados espontáneos para la LSC. Más allá de los resultados numéricos, el aporte principal radica en la validación de un pipeline que va desde la extracción

de datos de un corpus lingüístico real y continuo hasta la clasificación mediante arquitecturas avanzadas de grafos (MSG3D), poniendo a disposición de la comunidad tanto el dataset procesado como el código completo.

Tabla A.2: Resultados cuantitativos del reconocimiento de señas aisladas en LSC ([Naiara Garmendia, Horacio Saggion y Euan McGill, 2025a](#)).

Configuración	Top-1 Accuracy	Top-5 Accuracy
Entrenamiento desde cero	42.63 %	67.65 %
Pre-entrenado (SWL-LSE)	49.24 %	72.27 %

Anexo B

Métricas adicionales para enfoques continuos

B.1. Enfoque con segmentador

B.1.1. IoU Condicional

Con la finalidad de aislar la capacidad puramente técnica del segmentador cuando el modelo identifica con éxito una señal, se define una métrica condicional. **IoU Condicional** (Métrica de Recorte): Se promedia el *IoU* únicamente de aquellos segmentos que han sido clasificados correctamente.

La comparación entre el **IoU Condicional de Nivel 3 y 2** y el **IoU del Nivel 1** es de interés, ya que permite determinar si existe una correlación entre el nivel de clasificación del modelo de una señal y una mayor precisión en la delimitación de las fronteras temporales de la misma.

Tabla B.1: Comparación de IoU entre modelos MSG3D y SPOTER.

Nivel de Evaluación	Modelo MSG3D		Modelo SPOTER	
	IoU G.	IoU C.	IoU G.	IoU C.
Nivel 2: Reconocimiento Top-5	6,87 %	50,26 %	7,42 %	46,90 %
Nivel 3: Reconocimiento Top-1	3,89 %	54,24 %	3,18 %	45,64 %

El contraste entre ambas métricas es marcado. El IoU General se ubica entre 3 % y 7 %, mientras que el IoU Condicional alcanza valores entre 46 % y 54 %. Esta diferencia indica que el bajo IoU General no se debe principalmente a una mala delimitación temporal de los segmentos, sino a la gran proporción de segmentos incorrectamente clasificados que reciben *IoU* = 0 y disminuyen el promedio. Cuando el clasificador acierta, la calidad temporal del recorte producido por el segmentador es razonable.

Entre arquitecturas, MSG3D obtiene un IoU Condicional consistentemente superior al de SPOTER (54,24 % frente a 45,64 % en Nivel 3), lo que indica que sus aciertos tienden a concentrarse en segmentos mejor delimitados. Además, el IoU Condicional de MSG3D aumenta del Nivel 2 al Nivel 3 (de 50,26 % a 54,24 %), sugiriendo una correlación entre la confianza de clasificación y la calidad temporal del segmento: las señas que MSG3D clasifica correctamente en Top-1 presentan un mejor ajuste de fronteras que aquellas que solo aparecen en su Top-5. SPOTER, en cambio, muestra una leve disminución (de 46,90 % a 45,64 %), sin evidenciar esta correlación.

B.1.2. Variantes de Robustez para el Solapamiento

Con el fin de obtener un análisis más granular que permita desacoplar los errores de segmentación de los de clasificación, se proponen dos variantes adicionales de las métricas de solapamiento. Estas variantes se aplican de forma sistemática tanto al análisis de **Reconocimiento Top-5** como al de **Top-1**.

Métrica de Solapamiento Filtrada Esta variante evalúa la precisión del clasificador lingüístico únicamente sobre aquellos segmentos que el segmentador ha identificado como válidos. Bajo este criterio, se eliminan del denominador de la Precisión aquellos segmentos predichos S_p que no poseen ningún solapamiento con el ground truth ($|S_p \cap S_g| = 0$ para todo S_g). Esta métrica permite determinar para casos en los que el sistema detectó correctamente la presencia de una seña, con qué exactitud identificó su glosa, aislando así el ruido producido por los falsos positivos de la segmentación pura.

Métrica Best-Match Esta variante surge de la tendencia a sobre-segmentar del modelo segmentador e introduce una restricción de unicidad para mitigar este problema. Mientras que en la métrica estándar cualquier S_p que solape con un S_g puede ser contabilizado, en la métrica **Best-Match** solo es elegible para la evaluación el par (S_p, S_g) que maximice la intersección temporal:

$$S_p^* = \arg \max_{S_p} |S_p \cap S_g| \quad (\text{B.1})$$

Para un segmento de referencia dado S_g , solo su “mejor candidato” predicho S_p^* puede ser validado como un acierto. Si dicho candidato posee una glosa incorrecta, el caso se registra como un fallo de clasificación, independientemente de la existencia de otros segmentos menores que sí coincidan en etiqueta. Esta métrica resulta más rigurosa y proporciona una medida conservadora de la capacidad de reconocimiento del sistema, penalizando los casos en los que la sobre-segmentación genera múltiples candidatos para una misma seña de referencia.

Los resultados se resumen en la Tabla [B.2](#).

El análisis de estas métricas permite extraer los siguientes hallazgos:

Tabla B.2: Evaluación de robustez del reconocimiento (F1 Score) para Niveles 2 y 3.

Métrica de Evaluación	Top-5		Top-1	
	MSG3D	SPOTER	MSG3D	SPOTER
Estándar (Solapamiento)	13,53 %	15,61 %	7,07 %	6,84 %
Filtrada (Sin falsos positivos)	13,89 %	16,03 %	7,26 %	7,03 %
Best-Match (Mejor candidato)	12,72 %	14,63 %	6,68 %	6,62 %

1. **Impacto de la Sobre-segmentación:** Al aplicar la **Métrica Filtrada**, se observa un incremento en el F_1 en ambos modelos. Esto confirma que el clasificador es ligeramente más preciso de lo que sugieren las métricas estándar, ya que una fracción de sus errores se deben a predicciones realizadas sobre “espacios vacíos” donde no existe ninguna señal en el ground truth.
2. **Efecto de la Fragmentación (Best-Match):** El descenso del rendimiento bajo el criterio de **Best-Match** (por ejemplo, de 15,61 % a 14,63 % en SPOTER Top-5) revela que una parte de la exhaustividad (*Recall*) del sistema proviene de predicciones múltiples o desalineadas que solapan con la señal de referencia pero no son la coincidencia principal.
3. **Fidelidad temporal en aciertos:** El modelo **MSG3D** mantiene un IoU Condicional superior al de SPOTER en todos los niveles de clasificación. En particular, su IoU Condicional de Nivel 3 alcanza un **54,24 %**, lo que indica que cuando MSG3D clasifica correctamente una señal, el solapamiento temporal promedio entre el segmento predicho y el real supera la mitad de la unión de ambos intervalos. Este resultado sugiere que la calidad del recorte temporal del segmentador no se degrada significativamente en los segmentos correctamente clasificados.

B.2. Enfoque de ventana deslizante

B.2.1. Ratios por *frame*

Para complementar el análisis a nivel de segmentos, se evaluaron métricas de clasificación por *frame* sobre las señales del vocabulario. El *Ratio de Clase* mide, para cada señal del *ground truth*, la fracción de *frames* donde el sistema asigna la clase correcta; la variante generativa promedia sobre todas las señales, mientras que la condicional se restringe a aquellas donde al menos un *frame* fue clasificado correctamente.

Adicionalmente, se reportan dos métricas de precisión directa. La *Precisión Winner* mide la fracción de señales donde la clase ganadora del histograma de votos coincide con la clase del *ground truth*. La *Precisión Any* mide la fracción

de señas donde la clase correcta recibió al menos un voto, es decir, figuró como predicción de alguna ventana. La diferencia entre ambas refleja cuántas veces la clase correcta estuvo presente entre las predicciones pero no logró imponerse como ganadora en la votación.

La Tabla B.3 presenta los resultados.

Tabla B.3: Ratios por *frame* ($W=12$, $S=4$, con filtro, sobre señas del vocabulario).

Métrica	Top-1		Top-5	
	MSG3D	SPOTER	MSG3D	SPOTER
Precisión Winner	26,92 %	17,81 %	27,12 %	18,22 %
Precisión Any	30,85 %	22,36 %	54,87 %	41,20 %
Ratio Clase Gen.	16,59 %	12,02 %	48,86 %	36,29 %
Ratio Clase Cond.	61,64 %	67,53 %	89,06 %	88,08 %

La brecha entre *Precisión Winner* y *Precisión Any* revela con qué frecuencia la clase correcta estuvo presente entre las predicciones de las ventanas pero no logró imponerse en la votación. Para MSG3D en Top-5, esta diferencia es notable (27,12 % *vs.* 54,87 %): en más de la mitad de las señas del vocabulario, alguna ventana predijo la clase correcta dentro de su Top-5, pero esta no resultó ganadora del histograma.

El *Ratio de Clase Condicional* complementa esta observación al mostrar que, entre las señas donde el modelo acierta al menos un *frame*, la clasificación correcta se sostiene de forma consistente a lo largo de la seña: 88–89 % en Top-5 y 62–68 % en Top-1. Esto indica que los aciertos no son fluctuaciones aisladas sino predicciones sostenidas.

B.2.2. Impacto del filtro de suavizado

El filtro de suavizado por mayoría opera sobre bloques del tamaño del paso (*stride*), reemplazando la predicción de un bloque cuando sus bloques adyacentes coinciden en una clase diferente. Para evaluar su utilidad, se compararon los resultados con y sin filtro. La Tabla B.4 presenta esta comparación.

A pesar de que el filtro realiza aproximadamente 47–48 correcciones en las predicciones por *frame*, su impacto en las métricas finales es despreciable. Para MSG3D, las métricas a nivel de segmento son idénticas con y sin filtro, observándose únicamente una variación de 0,62 puntos porcentuales en la precisión por *frame*. Para SPOTER, la diferencia es igualmente marginal: un verdadero positivo adicional en Nivel 2 sin filtro y un segmento predicho más.

Este resultado indica que, para el video evaluado, las predicciones del sistema son suficientemente consistentes entre ventanas adyacentes como para que el suavizado temporal no aporte mejoras significativas.

Tabla B.4: Impacto del filtro de suavizado ($W=12$, $S=4$).

Métrica	MSG3D		SPOTER	
	Con filtro	Sin filtro	Con filtro	Sin filtro
F_1 Nivel 2 (Top-5)	9,43 %	9,43 %	17,49 %	17,82 %
F_1 Nivel 3 (Top-1)	4,57 %	4,57 %	4,60 %	4,59 %
Segmentos predichos	38	38	31	32
Prec. Winner Top-1	26,92 %	27,54 %	17,81 %	17,81 %
Correcciones aplicadas	47	0	48	0

B.2.3. Evaluación sobre señas del vocabulario

Las métricas principales se calculan sobre la totalidad de las 730 señas del *ground truth*, incluyendo las 247 señas OOV que constituyen fallos inevitables. Para aislar el desempeño del sistema sobre señas que efectivamente puede reconocer, se presenta en la Tabla B.5 la evaluación restringida a las 483 señas pertenecientes al vocabulario.

Tabla B.5: Rendimiento sobre señas del vocabulario (483 señas, excluyendo OOV).

Nivel de Evaluación	MSG3D		SPOTER	
	F_1	IoU G.	F_1	IoU G.
Nivel 2: Reconocimiento Top-5	9,43 %	2,36 %	17,49 %	2,25 %
Nivel 3: Reconocimiento Top-1	4,57 %	1,44 %	4,60 %	0,33 %

Al excluir las señas OOV, los valores de F_1 aumentan de forma notable – por ejemplo, SPOTER pasa de 12,06 % a 17,49 % en Nivel 2 – debido a que las señas fuera de vocabulario, que representan un 33,8 % del total, constituyen falsos negativos inevitables que reducen la exhaustividad. Esta perspectiva complementaria confirma que el impacto de las señas OOV es significativo en las métricas globales, y permite interpretar los resultados principales con mayor contexto.