**Instituto Tecnológico**

**y de Estudios Superiores de Occidente**

Reconocimiento de validez oficial de estudios de nivel superior según acuerdo secretarial 15018, publicado en el Diario Oficial de la Federación del 29 de noviembre de 1976.

Departamento de Electrónica, Sistemas e Informática

Maestría en Sistemas Computacionales



**título del trabajo.**

Trabajo recepcional que para obtener el grado de

Maestro en Sistemas Computacionales

Presenta: Nombre de quien presenta

Elija un elemento. Nombre de del asesor

Tlaquepaque, Jalisco. Haga clic aquí para escribir una fecha..

[La portada se deberá actualizar con la versión oficial disponible en: <https://www.iteso.mx/titulacionposgrados>]

[SI EL DOCUMENTO ES ESCRITO EN INGLES, se debe colocar la portada principal en inglés, y una segunda portada en español.]

AGRADECIMIENTOS

El autor desea dar las gracias a [Escriba los Agradecimientos].

[Para quienes hayan recibido apoyo CONACYT, se debe incluir aquí el agradecimiento por la Beca recibida, así como el número de beca – no confundir con número de CVU.]

[Se sugiere incluir agradecimiento al ITESO si se les apoyó con descuento o beca. Lo mismo para la empresa donde trabajan. También podrían incluir aquí agradecimiento a sus profesores, asesores o directores del trabajo.]

[SI EL DOCUMENTO ES ESCRITO EN INGLES, la primera página de agradecimientos es en inglés, y la segunda en español.]

DEDICATORIA

El autor dedica esta tesis a [Escriba la Dedicatoria].

[SI EL DOCUMENTO ES ESCRITO EN INGLES, la primera página de dedicatoria va en inglés, y se incluye una seguna en español]

RESUMEN

[Hablar brevemente sobre las partes de este trabajo, y lo que se presenta en cada uno] Se presenta una breve introducción a [al problema principal que resuelve este trabajo], el cual tiene como objetivo principal [mencionar objetivo principal], resolviendo de manera particular [a los objetivos específicos. [Se presenta el desarrollo del trabajo y sus principales resultados.] Finalmente, [se presentan las conclusiones del trabajo]. [Este resumen cuenta con 250 palabras máx.]

[SI EL DOCUMENTO SE ESCRIBE EN INGLÉS, la primera página de resumen va en inglés, y se deberá incluir una segunda página con el resumen en español.]

TABLA DE CONTENIDO

[Dar clic en el menú de Referencias/ Dar clic en Tabla de Contenido/ Seleccionar Tabla de Contenido Personalizada/ Seleccionar Formato Formal, 4 Niveles, Dar Clic en Opciones. En la ventana de Opciones, asignar Nivel de TDC 1 a Título Intro. Dar clic en Aceptar, y de nuevo en Aceptar].

[Maestría en Sistemas Computacionales 1](#_Toc200978041)

[AGRADECIMIENTOS 3](#_Toc200978042)

[DEDICATORIA 5](#_Toc200978043)

[RESUMEN 7](#_Toc200978044)

[TABLA DE CONTENIDO 9](#_Toc200978045)

[LISTA DE FIGURAS 11](#_Toc200978046)

[LISTA DE TABLAS 12](#_Toc200978047)

[LISTA DE ACRÓNIMOS Y ABREVIATURAS 13](#_Toc200978048)

[1. INTRODUCCIÓN 14](#_Toc200978049)

[1.1. Antecedentes 15](#_Toc200978050)

[1.2. Justificación 15](#_Toc200978051)

[1.3. Problema 15](#_Toc200978052)

[1.4. Hipótesis 15](#_Toc200978053)

[1.5. Objetivos 15](#_Toc200978054)

[1.5.1. Objetivo General: 15](#_Toc200978055)

[1.5.2. Objetivos Específicos: 15](#_Toc200978056)

[1.6. Novedad científica, tecnológica o aportación 15](#_Toc200978057)

[2. ESTADO DEL ARTE o de la TÉCNICA 16](#_Toc200978058)

[2.1. Tema relacionado 1 17](#_Toc200978059)

[2.2. Tema relacionado 2 17](#_Toc200978060)

[3. MARCO TEÓRICO/CONCEPTUAL 19](#_Toc200978061)

[3.1. Concepto básico 1 20](#_Toc200978062)

[3.2. Esquema básico 2 20](#_Toc200978063)

[3.2.1. Si se requiere subtema 20](#_Toc200978064)

[3.2.1.1. Subsubtema 20](#_Toc200978065)

[4. DESARROLLO METODOLÓGICO 21](#_Toc200978066)

[5. RESULTADOS Y DISCUSIÓN 26](#_Toc200978067)

[5.1. Resultados 27](#_Toc200978068)

[5.2. Discusión 27](#_Toc200978069)

[6. CONCLUSIONES 28](#_Toc200978070)

[6.1. Conclusiones 29](#_Toc200978071)

[6.2. Trabajo Futuro 29](#_Toc200978072)

[BIBLIOGRAFÍA 30](#_Toc200978073)

LISTA DE FIGURAS

[Figure 1: Methodology overview 22](#_Toc200982460)

[En el menú de Referencias, seleccionar Insertar Tabla de Ilustraciones. Si se desea insertar para Figuras, solo se debe seleccionar el rótulo de Figura.]

LISTA DE TABLAS

[Table 1: Raw dataset samples 22](#_Toc200977930)

[En el menú de Referencias, seleccionar Insertar Tabla de Ilustraciones. Si se desea insertar para Tablas, solo se debe seleccionar el rótulo de Tabla.]

LISTA DE ACRÓNIMOS Y ABREVIATURAS

[Del lado izquierdo va el acrónimo o abreviatura y del lado derecho su significado. La columna de en medio se deja en blanco. Al final, se deben quitar los bordes de la tabla]

|  |  |  |
| --- | --- | --- |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

[5 SALTOS DE LÍNEA CON ESTILO NORMAL ANTES DE PONER EL TITULO]

# INTRODUCCIÓN

**Resumen:** En este capítulo se presenta brevemente los antecedentes del objeto de estudio, justificación del objeto de estudio, justificación y la definición del problema. [hipótesis si la investigación lo justifica].

## Antecedentes

[incluir desarrollos, trabajos, resultados, publicaciones, etc., similares al objeto de este trabajo. Mencionar brevemente, sin entrar en el detalle de los mismos. El detalle se realizará en la sección de Estado del Arte o de la Técnica]

## Justificación

[Razones científicas, económicas, y/o sociales que muestren la necesidad de este trabajo]

## Problema

[Describir problema práctico, definir problema científico]

## Hipótesis

[Si la investigación lo justifica]

## Objetivos

#### Objetivo General:

[Definir de ser necesario. No incluir métodos o procedimientos]

#### Objetivos Específicos:

[Definir, no incluir métodos o procedimientos, numerados]

## Novedad científica, tecnológica o aportación

[Especificar lo novedoso de este trabajo o la aportación que brinda a la empresa o usuario]

# ESTADO DEL ARTE o de la TÉCNICA

**Resumen:** En este capítulo se presenta un resumen de los trabajos relacionados con [el objeto de estudio].

## Tema relacionado 1

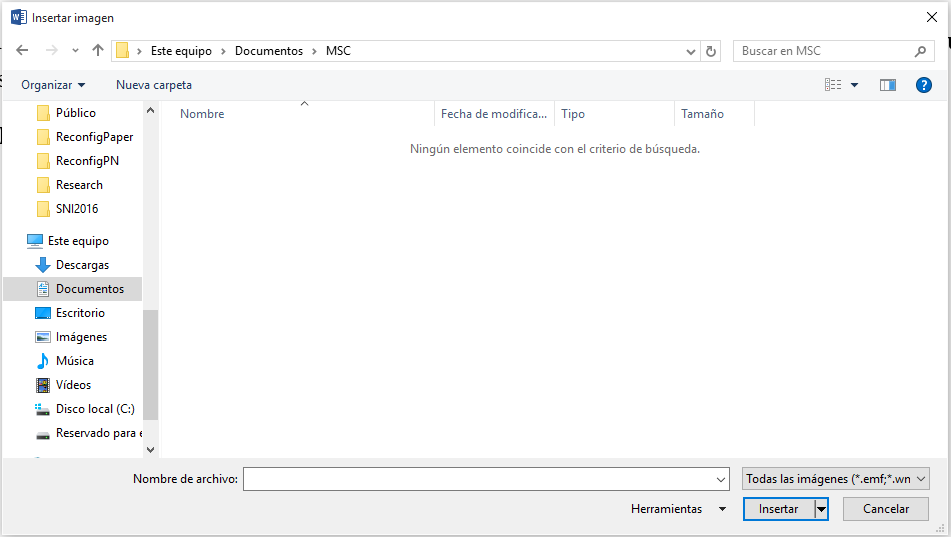
## Tema relacionado 2

[Detalle de desarrollos, investigaciones, publicaciones, trabajos relacionados a este proyecto]

[Ayuda a contextualizar el trabajo junto con la introducción, colocándolo en un marco más amplio. También servirá para prevenir la repetición de trabajo y/o errores de otros, para ayudar a ubicar información relevante para tu propio trabajo, aumentar el conocimiento alrededor del objeto de estudio, y para convencer a los evaluadores de este trabajo y a compañeros de la necesidad, pertinencia y la importancia de su investigación y la idoneidad de la metodología que se ha adoptado.

La presentación de lo que se ha investigado y escrito sobre un tema, es una manera de mostrar lo que hay que hacer. Se puede hacer esto mediante la indicación de las deficiencias de los estudios/trabajos anteriores, basándose en los resultados de estudios/trabajos anteriores, llevándolos un paso más allá, poniendo de relieve un área de investigación o desarrollo aún sin resolver o no reconocida, o simplemente mediante la adopción de un enfoque completamente diferente a un sujeto o problema. Al hacerlo, se muestra la importancia y el valor de su propia investigación o desarrollo. La identificación de temas clave o relevantes en su área de estudio. Identificar los métodos, enfoques y técnicas que podrían ser relevantes para su propia investigación/desarrollo. A familiarizarse con diferentes puntos de vista y/o opuestos, y para demostrar su capacidad de crítica y evaluar el trabajo de otros.

Para insertar una imagen, se recomienda ir a Imágenes en el menú Insertar. Se abrirá una ventana para seleccionar la imagen a insertar, tal como se muestra en la Figura 1. Para centrar la imagen, solo seleccionar la imagen y teclear Ctrl + T. Para darle un título a la imagen, seleccionar la imagen, dar clic en Insertar título en el menú Referencias. En la ventana de título, seleccionar rótulo Figura y en posición debajo de.]



No table of figures entries found.Figura 1. Ventana que se abre para insertar una imagen.

[Para darle título a una tabla, se realiza un proceso similar al de la imagen, con rótulo Tabla, y posición encima de. La forma de referenciar las tablas es de la misma manera que las referencias a las Figuras. Por ejemplo… En la Tabla 1 se muestran el costo por distancia recorrida.]

Tabla 1. Presentación de información en tablas.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Distancia (km)** | 2 | 3 | 5 | 7 | 8 | 10 | 12 | 13 |
| **Precio ($)** | 441,5 | 661,5 | 1.101,5 | 1.541,5 | 1.761,5 | 2.201,5 | 2.641,5 | 2.861,5 |

[DEBIDAMENTE REFERENCIADOS]

# MARCO TEÓRICO/CONCEPTUAL

**Resumen:** En este capítulo se presentan las bases teóricas y conceptuales sobre [el objeto de estudio].

## Concepto básico 1

## Esquema básico 2

## Si se requiere subtema

## Subsubtema

[La extensión de esta sección dependerá del tema y opinión del tutor.]

[conceptos, definiciones claves, teorías, descripción de desarrollos previos]

[DEBIDAMENTE REFERENCIADOS]

# DESARROLLO METODOLÓGICO

**Resumen:** [En este capítulo se presenta en detalle el desarrollo metodológico que incluye [pasos o proceso a seguir] un resumen de los trabajos relacionados con [el objeto de estudio].]

##### 4.1 Methodology Overview

We present an end-to-end pipeline for live-inference Mexican Sign Language (MSL) recognition, from raw video acquisition through final model deployment. Our ultimate goal is a working system that balances recognition accuracy with real-time performance on standard hardware. We first describe the raw dataset and compare it to available alternatives. Next, we detail our multi-phase data preparation, including landmark extraction, optional transformations, and dimensionality reduction. We then outline a two-stage pruning strategy—dataset reduction via lightweight proxies and encoder selection via focused fine-tuning—before presenting our best model training and tuning procedures. At each stage, we use objective metrics to guide decisions, ensuring a balance between accuracy and real-time performance.

|  |
| --- |
| Figure 1: Methodology overview |

##### 4.2 Raw Dataset

|  |  |
| --- | --- |
|  |  |
|  |  |

We obtained a publicly shared MSL dataset of 11 signers, each performing the 28 letters (those in the english alphabet as well as “LL” and “RR”), and 11 numerals (0–10) approximately ten times—five repetitions per hand—for a total of 4,156 short video clips [9]. The recordings isolate single static and dynamic signs under controlled conditions, providing balanced coverage across signers, classes, and handedness. Table [?] shows 4 frames from video samples of our dataset for the letters A, B, C and D by four different people, two with each hand.

Table 1: Raw dataset samples

The dataset is exceptionally well labeled and organized, following a clear directory and filename convention. At the top level, each signer’s data sits in a folder named pXX, where XX is a number identifying the signer. Within that, subfolders use the pattern Ciclo\_Y\_5\_Z, where Y denotes the cycle number (which goes from 1 to 5) and Z is “Derecha”  or “Izquierda” for left- or right-hand, indicating the hand which is actively performing the sign. Inside each subfolder, video files are named Ciclo\_Y\_5\_Z\_S.mp4, Y and Z keeping the same meaning, and S indicating the specific letter or numeral. This hierarchy encodes signer ID, cycle, handedness, and class at a glance, simplifying downstream filtering and preparation, analysis and training.

It is also important to note that although the sign label is the “natural” or intended tag, the other annotations can in practice be used as labels. For example, we leverage the handedness annotation to solve a secondary classification problem — active hand detection — since all signs in this dataset are single-handed. While continuous MSL often involves two-handed expressions, this controlled handedness information proves invaluable for our project’s preprocessing (see Section 4.3.3).

We load each clip into a pandas DataFrame, annotating signer ID, sign class, hand used, and frame index to establish a structured foundation for all preprocessing phases.

###### 4.2.1 Data Organization and Filtering

|  |
| --- |
| Figure 2: Data Organization and Filtering |

We leverage the dataset’s existing directory hierarchy parse signer ID, cycle number, handedness, and sign class into DataFrame columns. We then apply three filters to ensure data quality:

* We drop clips with fewer than twelve frames to guarantee sufficient temporal context.
* We enforce class balance by requiring at least 100 videos per class; any under-represented classes are excluded.
* We remove the “7” class entirely because its handshape is visually indistinguishable from “G” in this dialect, which would otherwise confuse isolated-sign models.

These steps yield a clean, balanced set of isolated-sign videos.

##### 4.3 Data Preparation

###### 4.3.1 Preparation Overview

Data preparation proceeds in three steps. First, we extract frame-level landmarks to capture hand shape, pose, and detection context (we call this phase 1, or PH1). Next, we optionally transform these landmarks to isolate the principal components of signs—hand configuration, hand orientation, hand position relative to the body, and hand movement (we call this phase 2, or PH2). Finally, we optionally apply dimensionality reduction (PCA or UMAP with varying component counts) to condense high-dimensional features (phase 3, PH3). Each step generates dataset forks that feed into our evaluation and pruning pipeline.

|  |
| --- |
| Figure 3: Data Preparation |

###### 4.3.2 (PH1) Video to Landmarks

**4.3.2.1 Purpose of Using Landmarks Instead of Images**  
Signs in MSL decompose into four principal components [1]:

* hand configuration
* hand orientation
* hand movement
* hand position relative to the body

These are sometimes supplemented by facial or torso cues for semantic meaning [1].

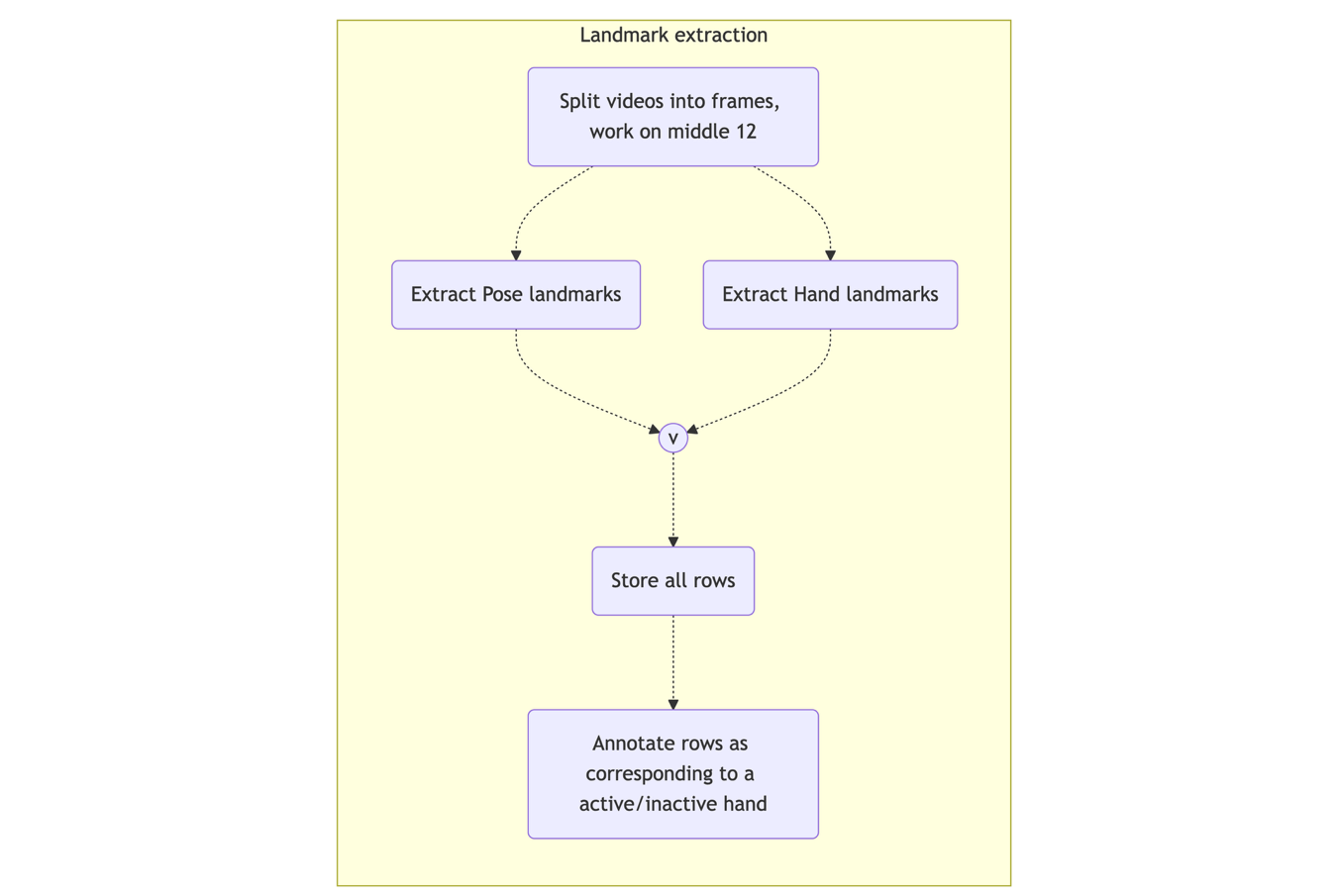
Extracting landmark coordinates offers a direct, low-dimensional representation of these components. Landmarks reduce data size by orders of magnitude compared to raw pixels, eliminate background clutter, and allow us to inherit robustness from MediaPipe’s training on diverse synthetic and real-world hand datasets.

**4.3.2.2 Basic Use of MediaPipe**  
MediaPipe Hands and Pose provide real-time hand and body landmark detection from a single RGB frame without the need for specialized hardware [5,6]. MediaPipe Hands returns 21 three-dimensional hand landmarks, which can be seen in Figure [?], as well as handedness labels, and confidence scores; MediaPipe Pose returns a small set of pose landmarks, which can be seen in Figure [?].

Integrating MediaPipe at the beginning of our system's pipeline allows us to inherit from it's robust capabilities. MediaPipe Hands was trained on a wide variety of data, with varying lighting, quality of image, camera angles, etc. Additionally, the model was partially trained on computer generated 3D images of hands, which allow it to infer depth coordinates even when using non-specialized cameras like MicroSoft Kinect. Lastly, the lightweight, optimized pipeline ensures frame-level feature extraction runs in under 10 ms per frame on CPU—critical for live inference.

|  |  |
| --- | --- |
| Figure 4: MediaPipe Hand Landmarks | Figure 5: MediaPipe Pose Landmarks |

**4.3.2.3 Use of MediaPipe for This Project**



|  |
| --- |
| Figure 6: Phase 1: Landmark Extraction |

A diagram summarizing this section can be seen in Figure [?].

We read each video using python's OpenCV package, and sample the 12 central frames. This strikes a balance between dataset size -- allowing us to obtain over 1000 frames per class -- and temporal coverage of the sign, particularly necessary in the case of dynamic signs. Additionally, it helps address a leakage issue with some of the videos in the dataset where the start (or end, respectively) of the video shows part of the previous (or following, respectively) sign which the signer performed. Through random exploration of the raw dataset, we found that 12 frames per video generally covered the totality of the sign's action. It is worth mentioning that the author of the paper which presented the dataset [9] used for this project, M. E. Rodriguez, used a strategy in a similar project where 30 frames per sign were obtained, and in cases where the video was too short, the last frame of the video was duplicated until the 30 mark was attained.

For each frame, MediaPipe Hands first detects potential hands. Then for each of these hands, it extracts 21 landmarks, the detected handedness, and a confidence score, as well as other information we don’t use here.

There is often more than one hand per frame. As mentioned previously, the signs in our dataset are all single-handed. Therefore, of the detected hands, one is often "inactive" and provides no information on the sign being performed, and we decided it would be better to train a second simpler model to for "active hand detection" which we use prior to the sign recognition model in our live-inference system. To train this model, we need a label for each frame that tells us whether the had is active or not. The frames or videos are not tagged in this manner directly, but the handedness tags on the video combined with the detected handedness tag and detected handedness confidence score provided by MediaPipe offer a proxy which we will describe shortly.

Similarly to the Hands framework, MediaPipe Pose extracts body landmarks, of which we use three (indices 0, 11, 12 in Figure [?]). Because there is only one person per video in our dataset, we never obtain more than one detection per frame.

In total, for each frame we obtain a varying number of rows[[1]](#footnote-1) in a pandas dataframe: one for each hand detected. For each of these rows we apply the following logic:

* if it is the only row corresponding to it's frame in it's video, then we label that row as corresponding to the active hand
* if it is one row of multiple (there are never more than 2) corresponding to it's frame in it's video, but it is the only one where the detected handedness matches the tagged handedness for the video, then we label that row as corresponding to the active hand; we tag the other rows corresponding to that frame of that video as inactive
* if it is one row of multiple corresponding to it's frame in it's video, and it is not the only one for which the detected handedness corresponds to the tagged handedness of the video, then we sort the rows in that group by handedness confidence score, and label the one with the highest score as active and the rest as inactive

In this way we obtain a secondary label for each row indicating the activity of the detected hand.

It is important to note that because the data for inactive hands is not relevant for sign recognition, we will drop the rows labeled as inactive right before training sign recognition models. Also, at this point the unit of data in our dataset is a frame. We intend on taking advantage of the sequential aspect of videos for sign recognition, which should be especially useful for dynamic signs. For that, we will also reformat our data right before training so that the rows corresponding to the 12 frames of a single video are concatenated into a single row per video. We do these two dataset transformations right before transformations right before training instead of earlier to avoid repeated storage of data and application of data preparation processes.

**4.3.2.4 PH1 Output Description**  
The result of this first data preparation process is a pandas dataframe with 69,572 rows and 80 columns, namely:

* 4 video tag columns
  + fileid
  + person\_id
  + cycle\_num
  + handedness
* 3 label columns
  + class\_name and class\_numeric, which indicate the sign
  + active\_hand
* One column to later concatenate rows coming from a single video in the correct order if needed
  + current\_frame
* 21 hand landmark triplet columns
  + h{i}x, h{i}y, h{i}z for i in [0, ..., 20]
* 3 pose landmark triplet columns
  + p{i}x, p{i}y, p{i}z for i in [0, 11, 12]

###### 4.3.3 (PH2) Landmark Transformations

|  |
| --- |
| Figure 7: Phase 2: Landmark Transformations |

###### As mentioned previously, MSL signs consist of four core components—hand configuration, hand orientation, hand position relative to the body, and hand movement [1]—all of which we already encode via raw landmark coordinates. However, these components intermix in the original camera frame. In PH2, we isolate seek to isolate each component. The diagram in Figure [?] summarizes the following descriptions.

###### To obtain the hand orientation, we approximate the palm of the hand to a plane, obtain a pair of vectors which describe it, compute a vector normal to them which points the way the palm faces and keep the coordinates for these three vectors.

###### To obtain the hand configuration, we use these three vectors and the coordinates of the wrist to define a new frame of reference in which we obtain the coordinates for each of the 21 hand landmarks.

###### To obtain the hand's position relative to the body, we apply a process similar to the previous two, but using the 3 pose landmarks we extracted earlier to define a plane, obtain a vector normal to that plane, using this one and two which define a plane as a base for a new frame of reference with the nose as the origin. Then we compute the mean of the coordinates of the hand in the original frame of reference, and move this to our new body-centered frame of reference.

###### Finally, Time-based movement emerges naturally when we later aggregate per video.

**4.3.3.1 PH2 Output Description**

As mentioned earlier, this phase of data preparation is "optional", meaning that we will keep both this version of the data and that without it to train models. The data that does go through this process exits as a pandas dataframe of 69572 rows and 83 columns, namely:

* 4 video tag columns
  + fileid
  + person\_id
  + cycle\_num
  + handedness
* 3 label columns
  + class\_name and class\_numeric, which indicate the sign
  + active\_hand
* One column to later concatenate rows coming from a single video in the correct order if needed
  + current\_frame
* triplet columns for each of the three vectors of the base of the "wrist's frame of reference"

###### h\_v{i}x, h\_v{i}y, h\_v{i}z, for i in [1, 2, 3]

###### 21 hand landmark triplet columns in the wrist's frame of reference

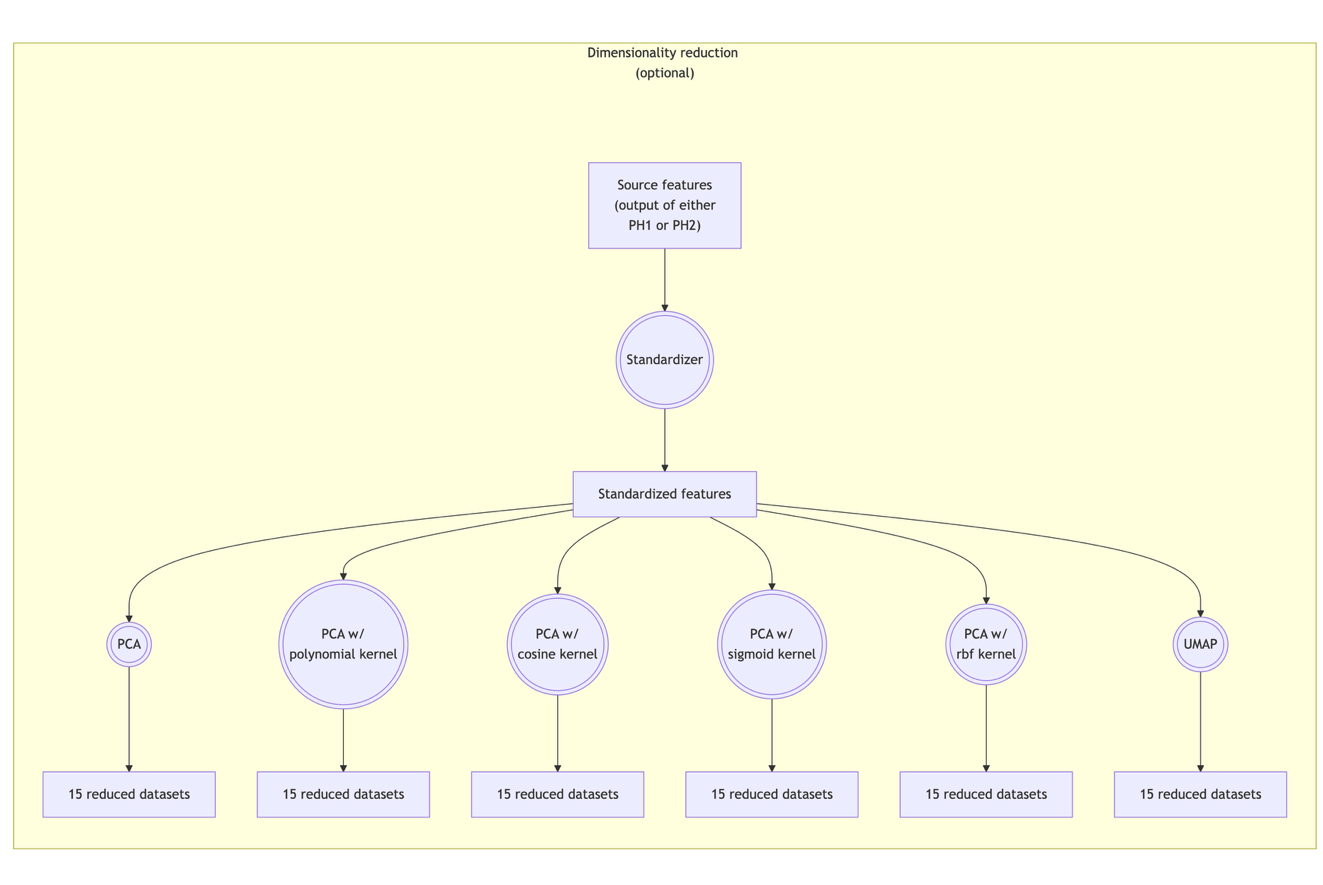
###### wh{i}x, wh{i}y, wh{i}z for i in [0, ..., 20]

###### 1 triplet of colums for the mean hand's coordinate in the "body's" frame of reference

###### p{i}x, p{i}y, p{i}z for i in [0, 11, 12]

###### Note that we no longer keep any pose landmark data.

###### 4.3.4 (PH3) Dimensionality Reduction



|  |
| --- |
| Figure 8: Phase 3: Dimensionality Reduction |

##### This phase of data preparation explores reducing feature dimensions via PCA and UMAP. We initially looked into this option early in this projects development when we still weren't using BERT based models, but rather K Nearest Neighbors models, which struggle to balance high-dimensional data. However, it proved to be a useful process in our final pipeline, and also provided us a way to visually analyze the quality of our data through graphics using 3 component reducers.

##### Figure [8] shows an overview of the actions performed in this phase. It begins with the output of either PH1 or PH2, which is optional. We first standardize all feature columns—excluding video tags, labels, and frame indices—to zero mean and unit variance. This normalization ensures that each dimension contributes equally to downstream calculations and mitigates scale disparities.

##### Next, we systematically apply dimensionality reduction across multiple reducer–kernel combinations and component counts. We test PCA and UMAP, as well as PCA with polynomial, RBF, sigmoid, and cosine kernels, for reduction component counts going from 1 to 15. We selected this range as roughly a quarter of our ∼ ⁣70 landmark features, balancing information retention with computational cost. Our choice of reduction techniques is due to their mature, easy-to-use implementations in scikit-learn and umap-learn, and their potential for on-device real-time use.

##### Using kernels prior to the PCA reducer proved to slow the process down significantly. Computing the full kernel matrix requires evaluating pairwise similarities for every sample in the dataset—an operation whose time and memory requirements scale quadratically with the number of points. To overcome this bottleneck, we turn to the scikit-learn’s implementation of Nyström approximation: rather than forming the complete n×n kernel matrix, a randomly sampled subset of m≪n “landmark” points is selected, with which only the n×m and m×m kernel blocks are computed, and then use these to reconstruct an approximation of the full matrix. This approximation reduces both computation and storage from O(n2) to O(nm). We used 500 as our value number of landmarks for the approximation, as it greatly reduced the time while mantaining visually similar results in the 3 dimensional case.

##### Using kernels prior to the PCA reducer proved to slow the process down significantly. Computing the full kernel matrix requires evaluating pairwise similarities for every sample in the dataset—an operation whose time and memory requirements scale quadratically with the number of points. To overcome this bottleneck, we use scikit‐learn’s implementation of the Nyström approximation: rather than forming the complete kernel matrix, a randomly sampled subset of “landmark” points is selected, only the and kernel blocks are computed, and then these are used to reconstruct an approximation of the full matrix. This approximation reduces both computation and storage from to We used 500 as our number of landmarks for the approximation, as it greatly reduced the time while maintaining visually similar results in the three‐dimensional case.



##### Each (reducer, kernel, n) combination yields a new dataset fork of the same 69572 rows and columns, where 8 corresponds to the same video tagging, labeling and frame indexing columns as before, and is the number of reduction components.



**4.3.4.1 3-Component Reduction Visualization Insights**

**[?] [?] [?] [?] [?] [?] [?] [?] [?] [?] [?] [?] [?] [?] [?] [?] [?] [?] [?] [?] [?] [?] [?] [?] [?] [?] [?]**

##### 4.4 Dataset and Model Selection

###### 4.4.1 Goal

###### We aim to deploy transformer-encoder classifiers for live MSL inference under strict latency and compute constraints. Our objective is to decode continuous sign input with minimal delay, maintain robust accuracy across signer variations, and support on-device inference on standard hardware. To this end, we select five lightweight encoders—bert-tiny, bert-mini, bert-small, bert-medium, and DistilBERT— whose respective sizes are compared in Table [?] and use a structured pruning pipeline to identify the best dataset-model combination without exhaustively training every pair.

|  |  |  |  |
| --- | --- | --- | --- |
| **Encoder** | **Number of Layers** | **Hidden Dimension size** | **Number of parameters** |
| **bert-tiny [?]** | 2 | 128 | 4.43 million |
| **bert-mini [?]** | 4 | 256 | 11.3 million |
| **bert-small [?]** | 4 | 512 | 29.1 million |
| **bert-medium [?]** | 8 | 512 | 41.7 million |
| **DistilBERT [?]** | 6 | 768 | 66 million |

Table 2: Size comparison of different BERT inspired architectures as seen in [?]

###### As our dataset preparation pipeline can generate hundreds of variants—each defined by different combinations of preprocessing steps, dimensionality reducers, and data‐unit formats—pairing every one of these with each of our five candidate encoder architectures would quickly become both too computationally and time consuming for this project.

###### To address this, we implement a two-stage pruning strategy that filters both datasets and models before committing to full training. First, we employ lightweight proxy evaluations—such as K-Nearest Neighbors on reduced data—to identify the most promising preprocessing and reduction settings without training the encoders. Once the top dataset variants are determined, we then perform a second screening across our encoder architectures using a reduced number of epochs. With this two-staged approach we are able to focus our resources on the dataset-model configurations that appear to have the most potential. Figure [?] summarizes this process.

|  |
| --- |
| Figure 9: Dataset and Model Selection |

###### 4.4.2 Dataset Selection

As mentioned in the previous section, the challenge at hand is reducing the number of candidate dataset-model combinations. We began with 546 dataset variants generated by combining:

* PH2 (2 options)
  + Applied
  + Not applied
* PH3 options (91 total):
  + PCA for n components, with n going from 1 to 15, making for 15 options
  + PCA and one of four kernels (Polynomial, Cosine, Sigmoid or RBF) for n components, with n going from 1 to 15, making for options



* + UMAP for n components, with n going from 1 to 15, making for 15 options
  + PH3 not applied (1 option)
* Data-unit formats (3):
  + all rows
  + active-hand rows per frame
  + active-hand rows aggregated per video

As we can see, the largest factor here is the number of PH3 options. Therefore, we first focused on finding what the best combinations in that area of choices would be, and discarding the rest. Figure [?] shows the strategy used to do so.

|  |
| --- |
| Figure 10: Dataset Selection |

**4.4.2.1 Selection of subsets using K-Means**

To have a more robust interpretation of the value of the different data preparation processes, we wanted to evaluate their impact as the number of classes grew. To do so, we decided to train KNN models on each of the datasets generated throughout the data preparation phases, but to do so on filtered versions of those datasets where only rows corresponding to certain subsets of classes were kept.

To select the subsets of classes we would filter our datasets with, we decided to use the data generated directly after landmark extraction with MediaPipe, and use a K-Means model to obtain the centroid for each class. Then, for each possible subset of classes, we computed a difficulty score based on the cosine similarity between the centroid corresponding to the classes in that subset. The closer the centroids were, the more difficult they should be to differentiate in theory. Finally, for each number of classes going from 2 to the total number of classes (37), we sorted all of the subsets for that number of classes by their difficulty score, and chose 3 for number of classes: one “easy”, which had the difficulty score closest to the the 10th percentile among candidate subsets; one “average”, which had the difficulty score closest to the the 50th percentile among candidate subsets; and one “hard”, which had the difficulty score closest to the the 90th percentile among candidate subsets.

The description above was a simplification of the process. We actually did not compute difficulty scores for every possible subset of classes due to computational limitations. Instead, for each number of classes from 2 to 37, we randomly sampled across all the candidate subsets at most 1000 of them, and computed the difficulty scores for those.

**4.4.2.2 Fit K Nearest Neighbors models for each subset, for each dataset**

Originally, our intention was that now that we had the three selected subsets for each number of classes, we would take each of the 546 datasets, and filtered them for each of the subsets, and fit several K-Nearest Neighbors models to the data. However, once again to be computationally resource-conscious, we only used the subsets corresponding to a reduced number of classes, namely: 2, 9, 16, 23, 30 and 37.

For each the datasets, we fit K-Nearest Neighbors models with the number of neighbors being amogst 1,3,5 and 7.

Then we computed the accuracy of each model. Because we were trying to compare models quality across different numbers of classes, comparing models’ metrics directly gave an advantage to models trained on lower numbers of classes.

To compensate for this, we decided to compute a significance score for the observed accuracy for each of the models, and compare them that way. We did this by considering that a score was more significant the better it was when comparing it to what random guessing would result in for the number of classes in question.

*4.4.2.2.1 Computing significance scores based on observed accuracy*

Let’s first recall the definition of multiclass accuracy. Given an n-class problem with N total examples, let be the number of examples whose true label is i and whose predicted label is j. Then



The accuracy is the proportion of correct predictions:



We assume two things under our random‐guessing model:

1. **Uniformly random true labels.** The dataset has basically no class imbalance, so each class i is equally likely as the true label:



1. **Uniformly random, independent guesses.** The classifier has no information and picks each class jj with equal probability, independently of the true label:



Because we treat “true label” and “guess” as independent under this model, the joint probability of any specific ordered pair (true=i,  guess=j)(true=i,guess=j) is the product of the two marginals:



Now, assuming uniform random guessing among n classes:

1. Each ordered pair (true=i, guess=j) has probability



1. Correct predictions occur when i=j. There are n such diagonal events, each of probability 1/n², so  
   .



Hence the expected (mean) accuracy under random guessing is



Under random guessing, each of the N examples has probability 1∕n of being correctly classified. We model the total number of correct predictions, C, as a Binomial(N, 1∕n) random variable.

1. Expected value of C



1. Variance of C



1. And we have that accuracy is C / N, therefore the variance of accuracy is:



1. And



1. For an observed accuracy Â, the Z-score is



*4.4.2.2.2 Comparing models across number of classes*

Now that we had significance scores for each of the models across number of classes we could compare them more fairly.

Figures [?] and [?] show that PCA using no kernel was generally the best reduction technique, and Figure [?] shows that it was best used with 15 components.

Additionally, we observe that using per frame data generally resulted in better models than per video data for KNN models, as can be seen in Figure [?]. This can intuitively be explained with the fact that KNN models have no concept of sequential data and therefore can not take advantage of the time-series structure of videos. However, we should also note that, as can be seen in Figure [?] all our models performed better for higher numbers of classes, which is proportionally related to the number of samples to train on. Because of this the fact that per frame datasets have both 12 times as many samples and 12 times as many feature columns, we believe that this could also explain why per frame models performed so much better than per video models for KNN.

Another insight that this exercise provided was that generally speaking PH2 landmark transformations negatively impacted our models, as can be seen in Figure [?]. In spite of this, we continue to consider these datasets, even thoug they generally double our traning dataset candidates, as an effort to validate the description of Mexican Sign Language as divided into 4 separate components which is one of the principal hypothesis the present project tries to prove.

In any case, moving forward we greatly reduced the number of candidate datasets, as we fixed PH3 = PCA(15), collapsing 91 options down to two (PCA(15) or none) and reducing our grid to 2 × 2 × 3 = 12 variants.

*4.4.2.2.3 Active Hand model*

In addition to this dataset reduction, we also obtained an excellent active-hand prediction model during exploration using the following configurations:  
[INSERT CONFIGURATION DETAILS HERE]

Therefore, we will no longer be exploring the datasets containing rows for inactive hands, and can further reduce our candidate datasets to 2 × 2 × 2 = 8 final datasets to explore with our 5 candidate model architectures.

###### 4.4.3 Model Selection Methods

We fine-tune each of the 8 datasets with all 5 encoders for a moderate epoch budget approximately proportional to the number of parameters per model seen in Table [?].

For each combination, we log accuracy, top 2 accuracy (meaning the rate at which the true label appears in the top 2 likeliest classes according to the model), macro precision, macro recall and macro F1 score.

We then compare the top 3 scores for different metrics for each model, and based on Figures [?], [?] and [?], where we see that …, we conclude that bert-mini is the best architecture for our usecase.

###### 4.4.4 Final Selection

BERT-mini is trained for ~5× more epochs on each of the 8 datasets than in section 4.4.3. We compare final metrics—prioritizing top-1 accuracy with top-2 and F1 score as tiebreakers—and select the sign-only, PH2-off, PH3-off dataset for deployment.

##### 4.5 Best Model Finetuning

The chosen dataset–BERT-mini pair undergoes extended training for 2× the epochs of Section 4.4.4. The final configurations used are:

# RESULTADOS Y DISCUSIÓN

**Resumen:** [En este capítulo se presentan los resultados obtenidos del desarrollo de este trabajo y una discusión sobre [el objeto de estudio]].

## Resultados

[Teclee los resultados en pasado. Ponga título a sus tablas y gráficos. Hacer referencia explícita utilizando la numeración. Ejemplo: …, como se muestra en la Figura 10. NO referenciar mencionado: como en la siguiente figura, o similar.]

## Discusión

[resultados más relevantes de este trabajo, los más relevantes de otros trabajos, comparar, referir a nuevos trabajos que puedan surgir de aquí, o problemas.

# CONCLUSIONES

**Resumen:** [En este capítulo se presentan las conclusiones y trabajo futuro en relación a [el objeto de estudio]].

## Conclusiones

[Las conclusiones deben responde a los objetivos establecidos]

## Trabajo Futuro

[Se refiere a recomendaciones o descripciones sobre líneas de investigación que abre este trabajo, aplicaciones inmediatas que se derivan, o desarrollo de componentes o extensiones del desarrollo.]

BIBLIOGRAFÍA

[Bibliografía Estilo IEEE: <http://www.ieee.org/documents/ieeecitationref.pdf>

El orden de la bibliografía debe ser el orden en el que aparece en el documento. Toda bibliografía puesta en esta sección deberá estar citada dentro del texto. La forma de hacer la cita cuando es una sola es [1], si son varias y salteadas es [3], [5], [8]. Si son varias consecutivas es [5]-[8]. Combinadas es [2]-[6], [9].

El formato de la bibliografía es como lista numerada simple, sin tabla.

El formato de cada una es como se describe a continuación y dependiendo del tipo:

*[Book Article*

* *For an article in an edited book, use practice similar to that for*[*author w/ editor or translator*](http://www.edshare.soton.ac.uk/77/1/bbieee-help.html#auth-ed/trans)*above, inserting article title between author[s] names and book title.*
* *[Citation Number] Author name[s], "*[*article title*](http://www.edshare.soton.ac.uk/77/1/bbieee-help.html#title-art)*," in*[*book title*](http://www.edshare.soton.ac.uk/77/1/bbieee-help.html#title)*,*[*editor names*](http://www.edshare.soton.ac.uk/77/1/bbieee-help.html#auth-ed/trans)*, publication location: publisher, year,*[*pages.*](http://www.edshare.soton.ac.uk/77/1/bbieee-help.html#pages)
* *Examples:]*

1. E.D. Lipson and B.D. Horowitz, "Photosensory reception and transduction," inSensory Receptors and Signal Transduction, J.L. Spudich and B.H. Satir, Eds. New York: Willey-Liss, 1991. pp. 1-64.
2. J. Lacan. "The insistence of the letter in the unconscious," in Psychoanalysis and Language, David Lodge, Ed., J. Rose, Trans., Ithaca, NY: Cornell University Press, 1992, pp. 123-34.

*[Journal*

* *[Citation Number] Author name[s], "*[*article title*](http://www.edshare.soton.ac.uk/77/1/bbieee-help.html#title-art)*,"*[*journal title*](http://www.edshare.soton.ac.uk/77/1/bbieee-help.html#title-per)*,*[*volume number, issue number, month (abbrv.)*](http://www.edshare.soton.ac.uk/77/1/bbieee-help.html#volume)*,*[*pages*](http://www.edshare.soton.ac.uk/77/1/bbieee-help.html#pages)*, publication year.*
* *Only include information which is pertinent to your source.  For example, many professional and academic journals do not have an issue month.  In that case, or when it seems unnecessary, do not include it in your citation.*
* *Examples:]*

1. K.A. Nelson, R.J. Dwayne Miller, D.R. Lutz, and M.D. Fayer, "Optical generation of turntable ultrasonic waves," Journal of Applied Physics, vol. 53, no. 2, Feb., pp. 1144-1149.
2. J. Allemang, "Social studies in gibberish," Quarterly Reviews of Doublespeak, vol. 20, no. 1, pp. 9-10.

*[Popular Periodical Article (monthly or bimonthly)*

* *In the case of popular monthly or bimonthly periodicals, omit volume number and issue, identifying instead by month and year of publication.*
* *[Citation Number] Author name[s], "*[*article title*](http://www.edshare.soton.ac.uk/77/1/bbieee-help.html#title-art)*,"*[*periodical  title*](http://www.edshare.soton.ac.uk/77/1/bbieee-help.html#title-per)*,*[*month (abbrv.)*](http://www.edshare.soton.ac.uk/77/1/bbieee-help.html#volume)*,*[*pages*](http://www.edshare.soton.ac.uk/77/1/bbieee-help.html#pages)*, publication year.*
* *Examples:]*

1. J. Fallows, "Network technology," Atlantic Monthly, Jul., pp. 34-36, 1994.

*[Popular Periodical Article (Biweekly, weekly, or daily)*

* *In the case of more frequently published periodicals, use day, month, and year to identify.*
* *[Citation Number] Author name[s], "*[*article title*](http://www.edshare.soton.ac.uk/77/1/bbieee-help.html#title-art)*,"*[*periodical  title*](http://www.edshare.soton.ac.uk/77/1/bbieee-help.html#title-per)*,*[*day number month (abbrv.)*](http://www.edshare.soton.ac.uk/77/1/bbieee-help.html#volume)*,*[*pages*](http://www.edshare.soton.ac.uk/77/1/bbieee-help.html#pages)*, publication year.*
* *Examples:]*

1. B. Metcalfe, "The numbers show how slowly the Internet runs today," Infoworld, 30 Sep., p. 34, 1996.
2. J. Turner, "Disorder 'kills without warning,'" The Toronto Star, 26 Jun., pp. F1-F2, 1998.

*[Paper Published in Conference Proceedings or Presented at Conference*

* *Treat a presentation in conference proceedings like an article in an edited book, including all available publication information.  Conference proceedings are often published by the organization holding the conference; in that case, do not cite the publisher.*
* *Example:]*

1. Paez-Borrallo, I.A. Perez-Alavarezz, and S.Z. Bello, "Adaptive foltering in data communications with self improved error reference," in Proc. IEEE ICASSP '94, 1994, pp. 65-68.

*[Treat an unpublished paper presented as a conference in the following manner: [Citation Number] Author name[s], "*[*article title*](http://www.edshare.soton.ac.uk/77/1/bbieee-help.html#title-art)*," presented at conference title. conference location, year.*

* *Example:]*

1. M. Lai, B. Chen, and S. Yuan, "Toward a new educational environment," presented at 4th Int. World Wide Web Conf. Boston, MA, 1995.

*[Web Page*

* *Give the author, title, type of medium (enclosed in brackets), volume and issue number (if on-line journal), page number (if relevant or given), and the year and the month of publication (in parentheses).  Then give the full internet address or the name of the online service provider prefaced by "Available at ".  If not an on-line journal, also put [cited year month day] before "Available at".*
* *Examples:]*

1. A. Harnack and G. Kleppinger, "Beyond the MLA Handbook: Documenting Electronic Sources on the Internet." Kairos, [Online serial] 1 (2), (1996 Sum), Available at HTTP: http://english.ttu.edu.kairos/1.2/
2. P. Curtis, "Mudding: Social Phenomena in text-based virtual realities," [Online document] Aug. 1992, [1996 Aug 30], Available at FTP: parcftp.xerox.com/pub/MOO/papers/DIAC921992.

**\*\*\*\*\*\*\*\*El estilo es Referencia**]

**APÉNDICE A. Título**

**APÉNDICE B. Título**

1. MediaPipe Hands sometimes, although rarely, fails to detect existing hands in a frame. In our case, out of the 3904 videos in our filtered dataset, on each of which we used the model to detect hands in the 12 middle frames, there were only 3 frames were no hands were detected and there should have been. We decided to drop the videos containing those frames, as their low number did not justify the investment needed to adapt our process to them. [↑](#footnote-ref-1)