



# RECONOCIMIENTO DE GESTOS DINÁMICOS EN LA LENGUA DE SEÑAS CHILENA

DEPARTAMENTO DE CIENCIAS DE LA INGENIERÍA  
INGENIERÍA CIVIL EN INFORMÁTICA  
PUERTO MONTT, CHILE

Patricio Alejandro Trujillo Arias  
[patriciaelenadro.trujillo@alumnos.ulagos.cl](mailto:patriciaelenadro.trujillo@alumnos.ulagos.cl)

Profesor Guía: Joel Torres  
15 de marzo de 2023

# Agradecimientos

Quiero comenzar agradeciendo a mi familia por el apoyo que me han ofrecido todos estos años de estudio, en especial a mi madre Marlene Arias que ha sido una constante motivación en mi aprendizaje y ha estado a mi lado en este proceso, y el motivo de comenzar mis estudios. También a mi padre Luis Trujillo que pese a no poder estar presente en todo este proceso debido a la distancia siempre me ha apoyado y se ha preocupado de mi proceso de aprendizaje. También quiero agradecer a mi profesor guía Joel Torres por el constante apoyo en la mejora del proyecto y correcciones necesarias.

## **Resumen**

La comunicación a través del uso de los gestos es uno de los métodos de comunicación más importante para un sector de la población que sufren de dificultades auditivas, en concreto en Chile esta lengua es llamada Lengua de señas Chilena. Podemos clasificar estos gestos en dos grandes grupos, por una parte los gestos Estáticos, y por otra los dinámicos que implican un movimiento en el espacio en un transcurso de tiempo. Hoy en día, en Chile, existen proyectos dedicados a resolver el reconocimiento de los gestos de la LSCH, pero estas se concentran en los gestos llamados estáticos.

En este proyecto se ofrece una solución para el reconocimiento de gestos dinámicos a través del uso de modelos de aprendizaje automático aplicado a visión por computadora.

Para la recolección de los datos se usa una librería de google llamada Mediapipe Holistic que permite tener puntos de referencia de varias partes de la mano y del cuerpo.

Los datos obtenidos se utilizan para reconocer los gestos dinámicos, esto por medio de Redes neuronales recurrentes(RNNs). En el entrenamiento se consideraron diferentes configuraciones de hiperparámetros.

La experimentación muestra que el modelo basado en GRU con dos capas de 64 neuronas logra una presión del conjunto de testeo del 95.67 %, demostrando que es posible el reconocimiento de señas dinámicas de la lengua de señas chilena.

Finalmente, esta solución plantea un aporte al desarrollo de mejores aplicaciones inclusivas orientadas a un grupo de personas con dificultades auditivas.

**Palabras Clave**— Reconocimiento de gestos, Lengua de señas Chilena, LSCH, Redes Neuronales Recurrentes, RNNs.

# Índice general

<b>1. Introducción</b>	<b>1</b>
<b>2. Marco Teórico</b>	<b>3</b>
2.1. Lengua de Señas Chilena . . . . .	3
2.1.1. Personas con discapacidad auditiva . . . . .	3
2.1.2. Antecedentes . . . . .	4
2.1.3. Tipos de gestos . . . . .	5
2.2. Inteligencia Artificial . . . . .	6
2.2.1. Inteligencia Artificial . . . . .	6
2.2.2. Aprendizaje Automático . . . . .	6
2.2.3. Aprendizaje Profundo . . . . .	10
2.2.4. Visión por Computador . . . . .	12
2.2.5. Redes Neuronales Recurrentes . . . . .	14
2.2.6. Limitaciones de las RNN . . . . .	17
2.2.7. LSTM(Long-Short Term Memory) . . . . .	18
2.2.8. Gated Recurrent Unit(GRU) . . . . .	20
2.2.9. Optimizador . . . . .	24
2.2.10. Validación cruzada . . . . .	26
2.3. Estado del Arte . . . . .	27
2.3.1. <b>Sistemas de reconocimiento de gestos con visión por computadora</b> . . . . .	27
2.3.2. <b>Sistemas de reconocimiento de gestos con alternativas a visión por computadora</b> . . . . .	28
<b>3. Formulación del Proyecto</b>	<b>29</b>
3.1. Objetivos . . . . .	29
3.1.1. Específicos . . . . .	29
3.2. Metodología de Trabajo . . . . .	30
3.2.1. Planificación . . . . .	31
<b>4. Búsqueda del Conjunto de Datos</b>	<b>33</b>
4.1. Conjunto de datos de gestos estáticos . . . . .	33
4.2. Dataset gestos Dinámicos . . . . .	34
4.2.1. <b>Human pose</b> . . . . .	34
4.2.2. <b>Hand tracking</b> . . . . .	36
4.2.3. <b>Mediapipe Holistic</b> . . . . .	37
4.3. Descripción del Conjunto de datos recopilado . . . . .	38

4.3.1. Abecedario . . . . .	38
4.3.2. Saludos . . . . .	38
4.3.3. Verbos . . . . .	39
4.3.4. Adjetivos . . . . .	39
<b>5. Modelo de Reconocimiento de Señas Dinámicas</b>	<b>41</b>
5.1. Estructura propuesta . . . . .	41
5.1.1. Primera estructura . . . . .	42
5.1.2. Segunda estructura . . . . .	44
<b>6. Experimentación</b>	<b>45</b>
6.0.1. Equipo . . . . .	45
6.0.2. Lenguaje de programación . . . . .	45
6.1. Experimentación con subconjunto de datos . . . . .	45
6.1.1. Tabla de resultados subconjunto . . . . .	47
6.1.2. Optimizador . . . . .	50
6.2. Modelos RNNs . . . . .	52
6.2.1. Elección de modelo . . . . .	53
6.2.2. Resultados . . . . .	54
<b>7. Arquitectura de Solución de reconocimiento de señas</b>	<b>57</b>
7.1. Esquema general de la solución . . . . .	57
7.1.1. Descripción estructura . . . . .	58
7.2. Implementación de captura de imágenes . . . . .	58
7.2.1. Alcances del Desarrollo . . . . .	60
<b>8. Trabajo Futuro</b>	<b>61</b>
<b>9. Conclusión</b>	<b>62</b>
<b>Referencias</b>	<b>63</b>

# Índice de figuras

2.1.	Distribución de personas con discapacidad . . . . .	3
2.2.	Alfabeto Lengua de Señas Chilena . . . . .	4
2.3.	Gesto estático A . . . . .	5
2.4.	Gesto dinámico Hola . . . . .	5
2.5.	Tipos de aprendizaje automático . . . . .	6
2.6.	Matriz de confusión con 2 etiquetas de clase. . . . .	8
2.7.	Curva ROC . . . . .	9
2.8.	ejemplo de una red neuronal totalmente conectada . . . . .	10
2.9.	Partes de una Red Neuronal . . . . .	11
2.10.	Diagrama de bloques de las etapas mas comunes en un sistema de visión por computadora . . . . .	12
2.11.	Misma imagen con diferentes tipos de transformaciones afines . . . . .	13
2.12.	Misma imagen con diferentes tipos de transformación de color . . . . .	13
2.13.	Añadir un buffer de memoria a una red neuronal con una capa oculta. . . . .	14
2.14.	Se escribe la activación de la capa oculta en el buffer de memoria. . . . .	15
2.15.	Fusionar el buffer de memoria con la siguiente entrada. . . . .	15
2.16.	Ciclo de escritura en la memoria y fusión con la siguiente entrada . . . . .	16
2.17.	Una Red RNN de una sola neurona desenrollada en el tiempo . . . . .	17
2.18.	Arquitectura de una LSTM con una compuerta de olvido . . . . .	18
2.19.	Arquitectura de una GRU . . . . .	20
2.20.	Función de activación sigmoide . . . . .	21
2.21.	Función de activación tangente hiperbólica . . . . .	22
2.22.	Función de activación ReLU . . . . .	23
2.23.	Validación cruzada con k-fold=10 . . . . .	26
3.1.	Carta Gantt hasta Agosto . . . . .	31
3.2.	Carta Gantt hasta Diciembre . . . . .	32
4.2.	Gesto A LSC . . . . .	34
4.3.	Hombre de Vitruvio alineado a través de dos puntos clave predichos por el detector BlazePose, incluye un recuadro que delimita la cara . . . . .	35
4.4.	Puntos de referencia . . . . .	35
4.5.	Puntos de referencia de la mano . . . . .	36
4.6.	Mediapipe Holistic modelo general . . . . .	37
4.7.	Alfabeto de la Lengua de señas chilena . . . . .	38
4.8.	Hola . . . . .	38

4.9. Chao . . . . .	38
4.10. Abrir . . . . .	39
4.11. Comer . . . . .	39
4.12. Caminar . . . . .	39
4.13. Ayudar . . . . .	39
4.14. Dibujar . . . . .	39
4.15. Disculpar . . . . .	39
4.16. Hablar . . . . .	39
4.17. Planchar . . . . .	39
4.18. Vender . . . . .	39
4.19. Flaco . . . . .	39
4.20. Bonito . . . . .	39
 5.1. Estructura para la obtención de los mejores resultados de un modelo RNN-LSTM	42
5.2. Estructura para la obtención del modelo final . . . . .	44
 6.1. 4 Modelos escogidos a partir de los resultados obtenidos . . . . .	49
6.2. Gráfico de caja de la primera estructura . . . . .	50
6.3. Gráfico de caja de la segunda estructura . . . . .	50
6.4. Gráfico de caja de la tercera estructura . . . . .	51
6.5. Gráfico de caja de la cuarta estructura . . . . .	51
6.6. Modelo GRU de 64 neuronas cada capa . . . . .	53
6.7. Precisión Entrenamiento vs Validación . . . . .	55
6.8. Perdida Entrenamiento vs Validación . . . . .	55
 7.1. Esquema de detección de gestos . . . . .	57
7.2. Letra dinámica J . . . . .	58
7.3. Palabra dinámica Ayudar . . . . .	59

# Índice de tablas

4.1. Datasets Gestos Estáticos . . . . .	40
4.2. Datasets Gestos Dinámicos una mano . . . . .	40
4.3. Datasets Gestos Dinámicos dos manos . . . . .	40
5.1. Estructura . . . . .	43
6.1. Subconjunto de datos . . . . .	46
6.2. Resultados subconjunto de datos 1 RNN . . . . .	47
6.3. Resultados subconjunto de datos 2 RNN . . . . .	47
6.4. Resultados subconjunto de datos 3 RNN . . . . .	48
6.5. Resultados subconjunto de datos 2 RNN y 1 Densa . . . . .	48
6.6. Resultados de la prueba con distintos modelos . . . . .	52
6.7. tabla Dataset Estático . . . . .	54
6.8. tabla Dataset Dinámico . . . . .	54
6.9. tabla Dataset Dinámico dos manos . . . . .	54
6.10. Resultados finales . . . . .	54
6.11. tabla con resultados NO COMPARABLES . . . . .	56

# Capítulo 1

## Introducción

A través de la historia, las personas sordas han sido percibidas como sujetos incapaces de comunicarse, excluyéndolas de varios aspectos en la participación en sociedad. Actualmente, en el mundo existe una gran cantidad de personas con sordera; más concretamente, según la Organización Mundial de la Salud (OMS) existen aproximadamente 72 millones de personas sordas en todo el mundo. En Chile, el grado de pérdida de audición alcanza las 712.005 personas del total de la población que tienen algún grado de pérdida, y de ellas se estima que 179.268 personas tienen sordera total.(Ramos y cols., 2015)

A raíz de ello, la comunicación a través de los gestos ha sido uno de los medio de comunicación más importantes entre las personas que sufren de dificultades auditivas parciales o completas, permitiéndoles expresar sus emociones o comunicar información a través de los gestos o señas realizadas con su torso, brazos y manos. Este método de comunicación es llamado Lengua de señas y es ampliamente usado internacionalmente, existiendo más de 300 diferentes lenguas de señas/gestos dependiendo de la región el la que se encuentre. Las modificaciones que existen en estas lenguas de señas se deben a particularidades locales al igual que lo hace el lenguaje hablado.

En chile, se utiliza la llamada *Lengua de Señas Chilena* (LSCh), en el cual podríamos clasificar los gestos en dos grupos, dinámicos y estáticos, la señal o gesto dinámico es aquel que requiere de movimiento de algún parte de cuerpo para dar un significado. Por otra parte, la señal o gesto estático solo necesita una pose en un solo instante para proporcionar un significado o idea.(Ronchetti, 2018)

Sumado a lo anterior, según el Informe 2008 de la Federación Mundial de Sordos, el gobierno Chileno cubre al menos una parte de los servicios de interpretación a la LSCh. Además, se declara que no existe oferta para la formación profesional de los intérpretes en el país. Por lo que la integración de las personas con algún grado de sordera en la sociedad es una tarea compleja y difícil de conseguir. Es por esto, que es necesario proveer una solución para contrarrestar esta falta de intérpretes, ya sea con cursos online o usos novedosos de tecnología.

Hoy en día, existen avances tecnológicos que permiten resolver problemas complejos de manera automática e inteligente. Para ser más concreto, estas tecnologías son referentes a la inteligencia artificial, que es la capacidad de las máquinas para usar algoritmos, aprender de los datos y utilizar lo aprendido en la toma de decisiones. Dentro de esta disciplina, existen áreas más especializadas como el *Aprendizaje Automático* que es la capacidad que tienen las máquinas de

aprender sin estar programados para ello.

Así también, existen algunos proyectos realizados en cuanto al reconocimiento de gestos en la lengua de señas chilena con las tecnologías mencionadas, pero estos se centran en gestos estáticos y excluyen a los gestos dinámicos utilizando tecnologías específicas para la interpretación de las imágenes. En contraste, existe la llamada *Visión por Computadora* que consiste básicamente en la deducción automática de la estructura y propiedades de un mundo tridimensional, posiblemente dinámico, a partir de una o varias imágenes.

De esta manera, la principal problemática identificada es la escasa variedad de signos en lengua de señas interpretados correctamente, lo que genera interpretaciones con errores o vacíos de contenido. Esto debido a que, las soluciones disponibles se centran solamente en señas estáticas, omitiendo el contenido en señas dinámicas.

A partir de lo mencionado, el propósito de este trabajo consiste en desarrollar un algoritmo que sea capaz de reconocer los gestos de la Lengua de Señas Chilena, en concreto los gestos denominados dinámicos, esto a través del uso de visión por computadora.

En adelante, se presenta el capítulo 2 que presenta los conceptos necesarios de este trabajo, además del estado del arte donde se presentan trabajos similares, en el capítulo 3 se definen los objetivos y la estructura que se utilizó para completar este proyecto, en el capítulo 4 se define el conjunto de datos, esto incluye tanto gestos estáticos como dinámicos, en el capítulo 5 se plantea la estructura a seguir para obtener el modelo óptimo, en el capítulo 6 se realiza la experimentación del conjunto de datos, en el penúltimo capítulo 8 se describen trabajos futuros y mejoras al algoritmo y finalmente en el último capítulo 9 se concluye el proyecto.

# Capítulo 2

## Marco Teórico

### 2.1. Lengua de Señas Chilena

La Lengua de señas es conjunto de señas que se realiza principalmente con las manos y otras partes del cuerpo como el rostro, el movimiento de los brazos, el cuerpo, para la comunicación entre la comunidad Sorda. En Chile se utiliza la Lengua de Señas Chilena (LSCH), que representan las letras de la lengua oral del país, a esto se le conoce como alfabeto manual. Las personas hispanohablantes utilizan un alfabeto manual similar pero no igual debido a variaciones en la representación de algunas letras.

En la LSCH existen gestos que son denominados estáticos ya que no representan un movimiento, como se ve en la figura 2.2 y los denominados gestos dinámicos que los que realizan un movimiento en el tiempo.

#### 2.1.1. Personas con discapacidad auditiva

En Chile, según el Segundo Estudio Nacional de la Discapacidad, de 2015, existen 2.836.818 personas con discapacidad, figura 2.1. De este total, 27,3 % de las personas con discapacidad tiene algún grado de pérdida de audición (712.005) y de ellas se estima que 179.268 personas tendrían sordera total. (Ramos y cols., 2015)

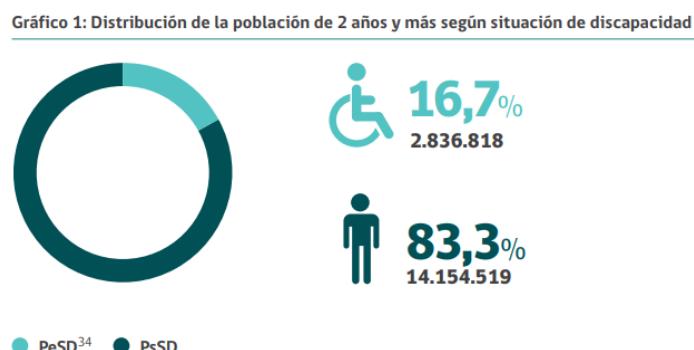


Figura 2.1: Distribución de personas con discapacidad

## 2.1.2. Antecedentes

Respecto a los antecedentes de la lengua de señas chilenas, existen registros asociados a comienzos del siglo pasado, en donde evoluciona, se expande y sistematiza esta lengua, por medio de un grupo organizado llamado “Asociación de Sordomudos de Chile”. Este grupo fue creado en 1926 con fines deportivos y recreativos, organizado por Robert Kelly Gray(Demartini y Letelier, 2006). Con apoyo de esta institución se crea la LSCh realizada por la comunidad sorda. Hoy en día en lo referente a mostrar una visibilidad a esta comunidad en Chile existe una ley(N° 21.303) que reconoce a la lengua de señas chilena como la lengua oficial de las personas sordas y promueve su uso en la educación, el mercado laboral, la salud y demás ámbitos de la vida.(Herrera Fernández, 2017)



Figura 2.2: Alfabeto Lengua de Señas Chilena

### 2.1.3. Tipos de gestos

Los parámetros que conforman la lengua de señas se pueden clasificar en 4 grupos(ADAMO QUINTELA, ACUÑA ROBERTSON, y CABRERA RAMÍREZ, 2013):

- Configuración de las manos(CM): Formas que una o dos manos adoptan al realizar una señal/gesto.
- Movimiento(M): El movimiento que se ejecuta durante la señal.
- Lugar(L): Donde se realiza la señal.
- Orientación(O): La orientación que adoptan las palmas de las manos durante la realización de la señal.

#### Gestos Estáticos

Son gestos que no implican movimiento en el espacio, el gesto hace uso de la configuración de las manos(CM),Lugar(L) y la orientación de la misma(O).



Figura 2.3: Gesto estático A

#### Gestos Dinámicos

Son gestos que tiene en cuenta el movimiento espacial y temporal, lo que constituye esta señal es la configuración de la mano(CM), el Movimiento(M), el lugar(L) y la orientación(O).



Figura 2.4: Gesto dinámico Hola

## 2.2. Inteligencia Artificial

### 2.2.1. Inteligencia Artificial

La inteligencia artificial es la capacidad que tienen las máquinas para utilizar algoritmos que le permitan aprender de una cierta cantidad de datos agrupados y de esta manera utilizar lo que aprendió para, de manera independiente, hacer uso de este a través de decisiones, intentando simular el comportamiento humano. Con esta característica se puede analizar una gran cantidad de información al mismo tiempo. También hay que destacar que la cantidad de errores producidos por las máquinas es significativamente menor al que una persona se ve expuesta o realiza. El hecho de que programas informáticos pueden tomar decisiones es de gran importancia ya que a diferencia de las personas estas crecen de manera exponencial a medida que avanza el tiempo. Teniendo en cuenta estas dos características o capacidades los sistemas que utilizan inteligencia artificial pueden realizar y desempeñar tareas que antes estaban sólo reservadas a los humanos.(Madrid:, s.f.)

### 2.2.2. Aprendizaje Automático

El aprendizaje automático(Machine Learning) es una categoría de la inteligencia artificial, la cual profundiza en la capacidad de aprender de una máquina/ programa informático. Esta área de la informática permite o se enfoca en que las máquinas tengan la capacidad de aprender sin que hayan sido programadas para ello. Para entender esto solo hay que ver la manera en la que están personalizadas las páginas de redes sociales, como Facebook o los resultados que te devuelve el motor de búsqueda de google. En concreto el aprendizaje automático hace uso de algoritmos para aprender y encontrar patrones en los datos. Un ejemplo de esto serían los filtros que están en el correo electrónico, estos filtros detectan y a través de patrones los correos spam y de esta manera separarlo del correo no basura. Esto se logra a través la toma de decisiones de patrones y conocimientos aprendidos.(Madrid:, s.f.) En el aprendizaje automático que pueden utilizarse: aprendizaje supervisado, no supervisado y de refuerzo [Figura 2.5]

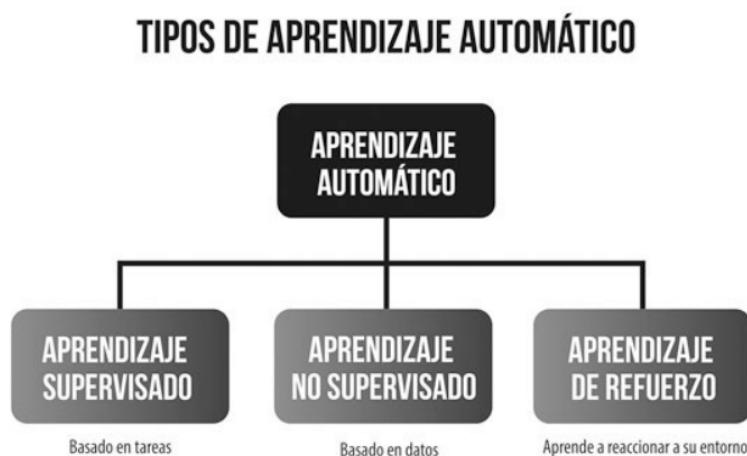


Figura 2.5: Tipos de aprendizaje automático

## Aprendizaje Supervisado

El aprendizaje supervisado es donde los algoritmos utilizan datos que ya han sido clasificados con anterioridad, esto se hace para que los nuevos datos que vayan a ingresar ya tengan una categoría en la cual ser encasilladas. Este método requiere de intervención humana para entregarles retroalimentación.(Madrid; s.f.) Estos algoritmos serán los que se utilizaran para la identificación de gestos de la lengua de señas chilena realizado en tiempo real.

En aprendizaje supervisado se suelen usar:

### ■ Clasificación

Al utilizar la clasificación forzamos a que los resultados estén etiquetados en una clase en específico, esta clase está entre varias clases definidas anteriormente. Las clases son categorías en las cuales encasillamos los resultados según el tipo de problema planteado. Un ejemplo de esto sería un hospital donde se tiene que clasificar si el paciente sobrevive o no. Allí la variable de respuesta es una ficticia que toma valores de 1 y 0. Esto sería un problema de clasificación debido a que intentamos predecir a qué clase pertenece la población. Este será el tipo de aprendizaje es el indicado para el reconocimiento de gestos etiquetados con anterioridad como lo son los gestos de la lengua de señas chilena.

En un contexto de marketing, podríamos predecir qué clientes tienen más probabilidades de comprar un determinado producto. En un contexto de visión por ordenador, podemos querer predecir si una imagen contiene un determinado objeto.

Las aplicaciones de clasificación son muy comunes. Y en muchos casos hay más de dos clases, como en la identificación de muchos caracteres impresos de caracteres impresos en la visión por ordenador.(Matloff, 2017)

### ■ Regresión

La regresión es la tarea de aproximar una función ( $f$ ) de las variables de entrada ( $X$ ) a una variable de salida continua ( $y$ ).

Por variable a menudo se trata de valores cuantitativos, como cantidades y tamaños.(Brownlee, 2017)

Por ejemplo, se puede predecir que una casa se venderá por un valor específico en dólares, quizás en el rango de 100.000 a 200.000 dólares.

## Métricas de Clasificación

Las métricas se eligen para evaluar los algoritmos de aprendizaje automático . La elección de la métrica influye en la forma de medir y comparar el rendimiento. Algunas de las cuales son Matriz de confusión, Clasificación Accuracy, Precision, Recall, F1, Curva Roc-acc(Dangeti, 2017)(Bonacorso, 2017)

### Matriz de confusión

La matriz de confusión es una presentación práctica de la precisión de un modelo con dos o más clases. Es una tabla que en el eje X se presentan las predicciones y los resultados de precisión en el eje Y.(Dangeti, 2017)

		Predicted Values	
		Negative	Positive
Actual Values	Negative	TN True Negative	FP False positive
	Positive	FN False Negative	TP True Positive

Figura 2.6: Matriz de confusión con 2 etiquetas de clase.

- True positives (TPs)= Cuando la clase real del punto de datos era 1 (Verdadero) y la predicha es también 1 (Verdadero)
- True negatives (TNs)= Cuando la clase real del punto de datos fue 0 (Falso) y el pronosticado también es 0 (Falso).
- False positives (FPs)= Cuando la clase real del punto de datos era 0 (False) y el pronosticado es 1 (True).
- False negatives (FNs)= Cuando la clase real del punto de datos era 1 (Verdadero) y el valor predicho es 0 (Falso).

**Clasificación Accuracy:** Es el porcentaje total de elementos clasificados correctamente.

$$Accuracy = \frac{(TP + TN)}{(TP + TN + FP + FN)} \quad (2.1)$$

**Precision:** Es el número de elementos identificados correctamente como positivo de un total de elementos identificados como positivos.

$$Precision = \frac{TP}{TP + FP} \quad (2.2)$$

**Recall:** Es el número de elementos identificados correctamente como positivos del total de positivos verdaderos.

$$Recall = \frac{TP}{TP + FN} \quad (2.3)$$

**F1:** Es la media armónica de la precisión(P) y Recall(R). Si se multiplica la constante de 2.

$$F1 = \frac{2 * P * R}{P * R} \quad (2.4)$$

**Curva Roc:** La curva Roc es una herramienta para comparar diferentes clasificadores que pueden asignar una puntuación a sus predicciones. Esta puntuación puede interpretarse como una probabilidad, por lo que está acotada entre 0 y 1. En la figura 2.7 se representa esta curva.(Bonaccorso, 2017)

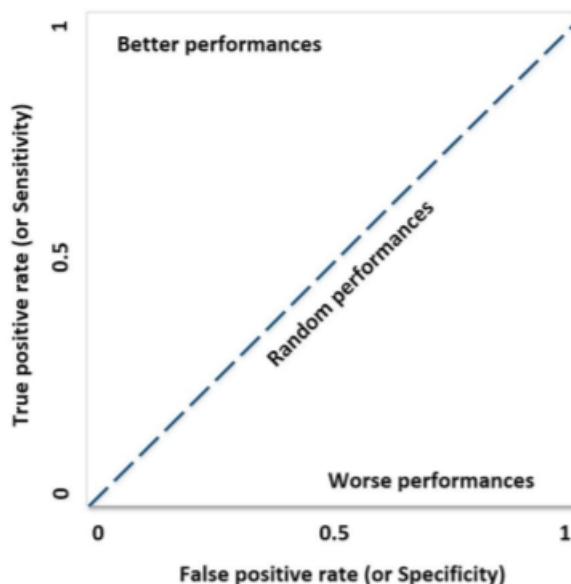


Figura 2.7: Curva ROC

El eje X representa la tasa creciente de falsos positivos (especificidad), mientras que el eje Y representa la tasa de verdaderos positivos (sensibilidad). La línea segmentada representa un clasificador perfectamente aleatorio, por lo que todas las curvas por debajo de este umbral tienen peor rendimiento que una elección aleatoria, mientras que las que están por encima muestran un mejor rendimiento.(Bonaccorso, 2017)

### 2.2.3. Aprendizaje Profundo

El aprendizaje profundo es el subcampo de la inteligencia artificial que se centra en la creación de grandes modelos de redes neuronales capaces de tomar decisiones precisas basadas en datos. El aprendizaje profundo es especialmente adecuado para contextos en los que los datos son complejos.(Kelleher, 2019)

El aprendizaje profundo moderno proporciona un potente marco para el aprendizaje supervisado. Al añadir más capas y más unidades dentro de una capa, una red profunda puede representar funciones de complejidad creciente. La mayoría de las tareas que consisten en asignar un vector de entrada a un vector de salida.(Goodfellow, Bengio, y Courville, 2016)

El aprendizaje profundo es ahora la tecnología estándar para el reconocimiento del habla, y también para la detección de rostros y por ende gestos en las cámaras digitales.

#### Redes Neuronales

El propósito de una red neuronal es imitar algunas características presentes en el ser humano, como la capacidad de memorizar y entrelazar sucesos o hechos. Si tomamos en cuenta los problemas que no pueden resolverse a través de un simple algoritmo, todos tienen algo en común, el hecho que requieren de experiencia para funcionar. Esto es algo que el ser humano puede resolver de manera natural a través de la experiencia que ha acumulado en el tiempo. A partir de lo dicho una manera de resolver este problema es la construcción de sistemas que tengan la capacidad de imitar esta característica. En otras palabras, las redes neuronales son un modelo artificial(muy simplificado) del cerebro humano, que es la guía más cercana que tenemos de un sistema capaz de adquirir conocimientos a través de la experiencia. Una red neuronal “un nuevo sistema para el tratamiento de la información, cuya unidad básica de procesamiento está inspirada en la célula fundamental del sistema nervioso humano: la neurona”.(Matich, 2001)

#### Elementos básicos que componen una red neuronal.

Figura 2.6, Esquema red neuronal:

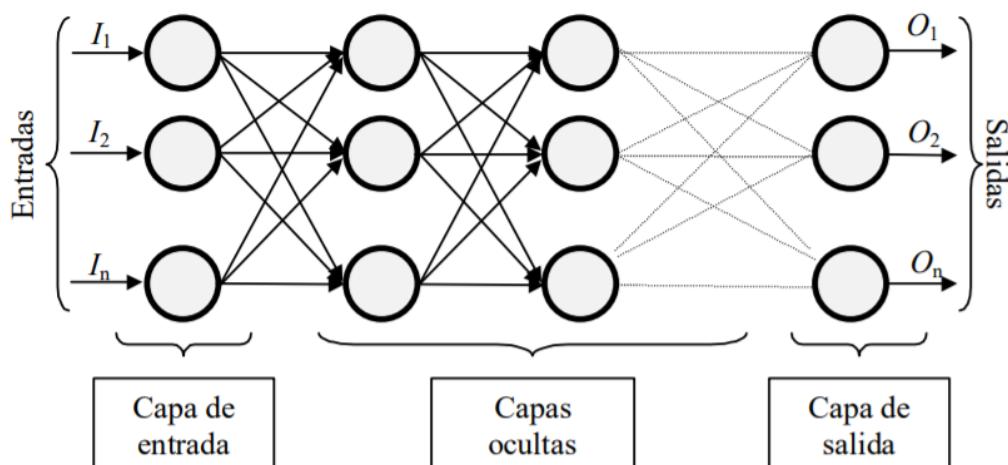


Figura 2.8: ejemplo de una red neuronal totalmente conectada

La misma está constituida por neuronas interconectadas y arregladas en tres capas (esto último puede variar). Los datos ingresan por medio de la “capa de entrada”, pasan a través de la “capa oculta” y salen por la “capa de salida”. Cabe mencionar que la capa oculta puede estar constituida por varias capas.

A continuación se muestra una Red neuronal con sus respectivas partes, figura 2.9

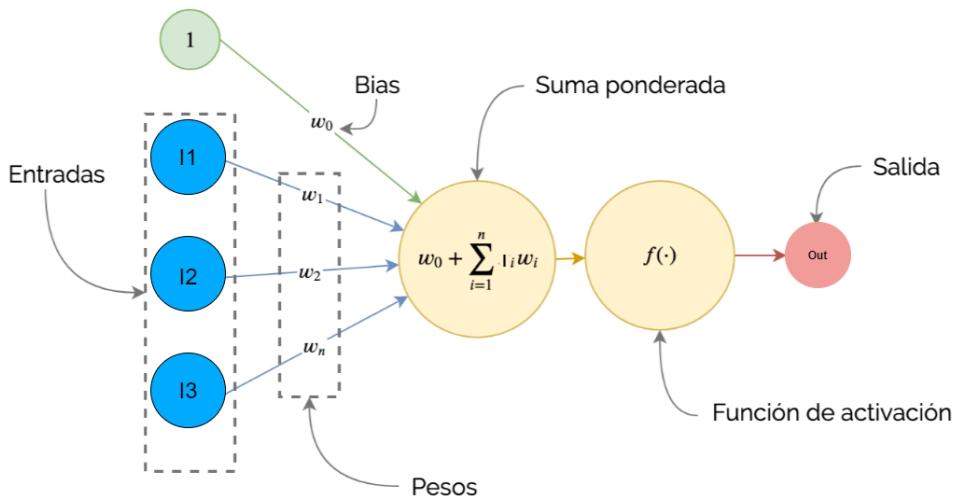


Figura 2.9: Partes de una Red Neuronal

### Función de entrada.

La función de entrada puede describirse como sigue:  $\text{input}_i = (\text{ini}_1 \cdot w_{i1}) * (\text{ini}_2 \cdot w_{i2}) * \dots * (\text{ini}_n \cdot w_{in})$

Donde "n" representa al número de entradas a la neurona  $N_i$  y  $w_i$  al peso.(Matich, 2001)

### Función de activación

La función activación calcula el estado de actividad de una neurona; transformando la entrada global (menos el umbral,  $O_i$ ) en un valor (estado) de activación, cuyo rango normalmente va de (0 a 1) o de (-1 a 1). Esto es así, porque una neurona puede estar totalmente inactiva (0 o -1) o activa (1).

### Función de Salida

La función de salida determina qué valor se transfiere a las neuronas vinculadas. Si la función de activación está por debajo de un umbral determinado, ninguna salida se pasa a la neurona subsiguiente. Normalmente, no cualquier valor es permitido como una entrada para una neurona, por lo tanto, los valores de salida están comprendidos en el rango [0, 1] o [-1, 1]. También pueden ser binarios 0, 1 o -1, 1.

## 2.2.4. Visión por Computador

La visión por computadora es un campo de inteligencia artificial (IA) que permite a las computadoras y sistemas derivar información significativa de imágenes digitales, videos y otras entradas visuales, y tomar acciones o hacer recomendaciones basadas en esa información.(Mery, 2004)

La visión artificial consiste básicamente en la deducción automática de la estructura y propiedades de un mundo tridimensional, posiblemente dinámico, a partir de una o varias imágenes bidimensionales de ese mundo.(Sánchez, Esteban, Vélez, y Moreno, 2003)

### Visión por computadora Etapas:

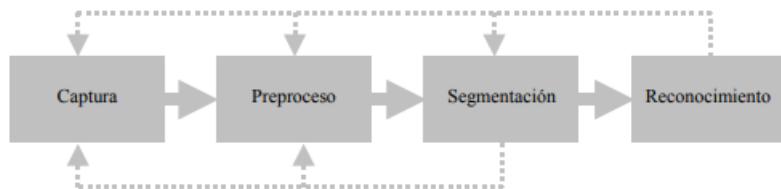


Figura 2.10: Diagrama de bloques de las etapas más comunes en un sistema de visión por computadora

- Captura de imagen: Consiste en la captura o adquisición de las imágenes digitales o video a estudiar mediante algún tipo de sensor.
- Preproceso: Es el tratamiento digital de las imágenes para facilitar las etapas que le siguen y obtener una mejor calidad de imagen. En esta etapa se aplican filtros y transformaciones, como eliminación de ruido en la imagen o aumento de contraste.
- Segmentación: Consiste en aislar e identificar los elementos u objetos que se requieren para el estudio de la imagen.
- Reconocimiento: En esta etapa se distinguen los objetos segmentados, partiendo del análisis de determinadas características que se establecen con anterioridad para diferenciarlos o clasificarlos.

### Data Image Augmentation

Aumento de datos para la clasificación de imágenes y vídeo. Un algoritmo de aprendizaje profundo puede ser más eficaz en cuanto tenga más acceso a datos. Por eso es posible a través de un pequeño conjunto de datos estructurados añadir copias ligeramente modificadas del mismo conjunto y de esta manera mejorar el rendimiento.(Perez y Wang, 2017)

#### Métodos

Los métodos de aumento de imagen existentes pueden clasificarse en una dos categorías muy generales: los métodos tradicionales de caja blanca o métodos de caja negra basados en redes neuronales profundas.(Mikołajczyk y Grotowski, 2018)

- Transformaciones tradicionales

Las transformaciones tradicionales consisten en utilizar una combinación de transformaciones afines para manipular los datos de entrenamiento, Para cada imagen de entrada, generamos una imagen que es duplicada de diferentes formas como: rotación, reflexión, escalado (zoom in/out), Shear.

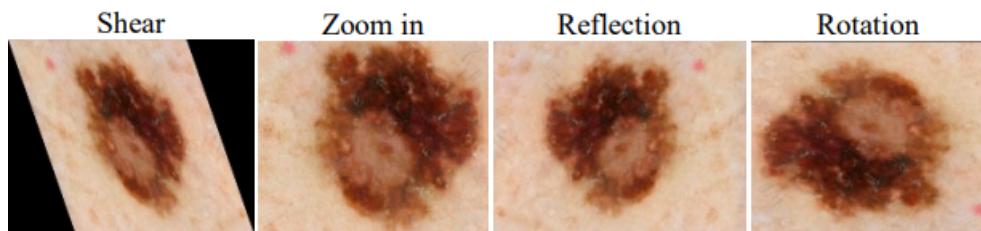


Figura 2.11: Misma imagen con diferentes tipos de transformaciones afines

Además de estas están la modificación del color: Ecualización del histograma, mejora del contraste, equilibrio de blancos, nitidez y desenfoque.

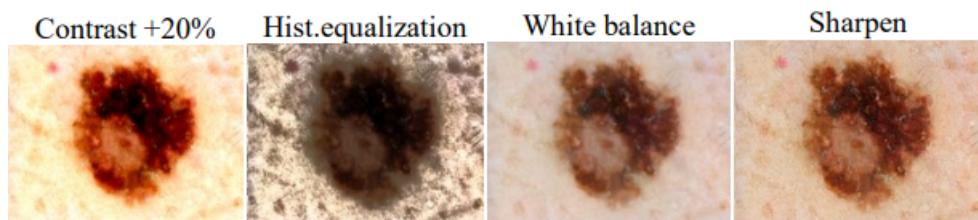


Figura 2.12: Misma imagen con diferentes tipos de transformación de color

### Clasificación de imágenes

**OpenCV** OpenCV(Open Source Computer Vision) es una biblioteca de visión por computador más grandes en términos de funcionalidades poseídas, contiene implementaciones de más de 2500 algoritmos, algunos son detección de movimiento, reconocimiento de objetos, entre otros.

## 2.2.5. Redes Neuronales Recurrentes

Las redes neuronales recurrentes (RNN) son un grupo de redes neuronales para el procesamiento de datos secuenciales de valores  $X = X_1 \dots X_T$ . Estas redes pueden escalar secuencias de mayor longitud de la que lo haría una red no especializada. La mayoría de las redes recurrentes además pueden procesar secuencias de una longitud que varía, como por ejemplo entradas de video con diferentes cantidades de fotogramas captados. La idea de estas redes recurrentes surgió alrededor de 1980 con la finalidad de compartir parámetros dentro del modelo, esto hizo posible aplicar modelos de diferente longitud.

Por ejemplo si tuviéramos la frase “Fui a Santiago el 2010”, luego creamos un modelo de aprendizaje automático que lea esta frase y determine en qué año se fue a Santiago. Una red tradicional que esté totalmente conectada tendría parámetros asignados para cada característica de entrada, esto implica que el algoritmo tendría que aprender las reglas del lenguaje de manera separada en cada una de las posiciones de la frase. A diferencia de las RNNs que comparten los pesos a lo largo del tiempo.(Goodfellow y cols., 2016)

Las redes neuronales tradicionales actúan de tal manera que la función de activación actúa en una dirección(mientras se realiza el entrenamiento), comienza en la capa de entrada y termina en la capa de salida. Una RNN en cambio también poseen conexiones “Hacia atrás”. Las salidas de algunas neuronas de entrada se retro alimentan siendo parte de la siguiente entrada. Para crear esta red neuronal se aumenta esta red con un buffer de memoria, figura 2.13 .(Kelleher, 2019)

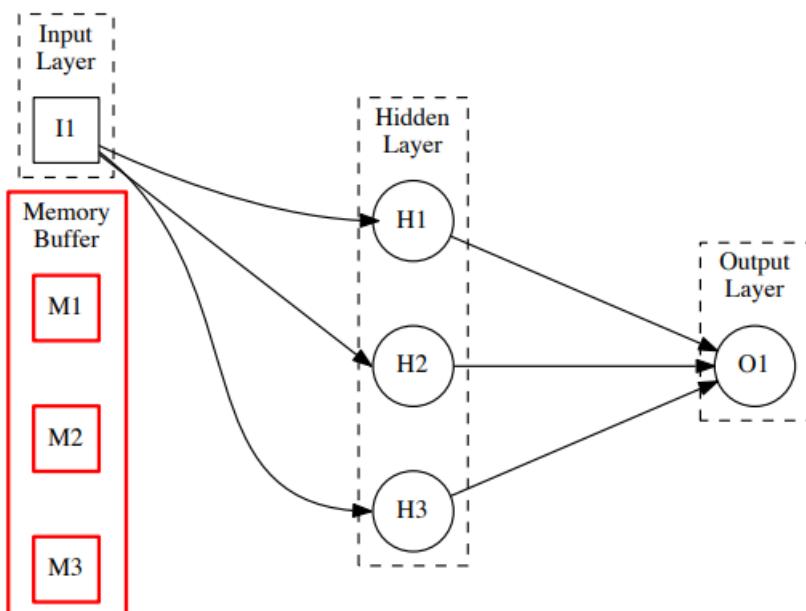


Figura 2.13: Añadir un buffer de memoria a una red neuronal con una capa oculta.

Por cada entrada a la red, la salida de las unidades en la capa oculta en cada input se almacenan en un buffer de memoria, esta nueva información sobreescribe lo que había en la memoria con anterioridad, figura 2.14.

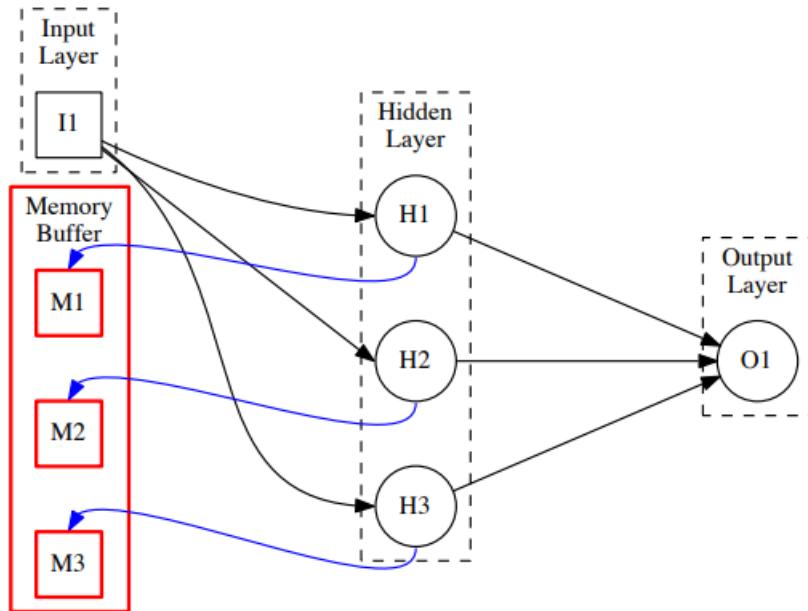


Figura 2.14: Se escribe la activación de la capa oculta en el buffer de memoria.

A continuación en el siguiente instante de tiempo (llamado timestep) se procede a combinar los datos almacenados en el buffer con la entrada en ese paso de tiempo, figura 2.15.

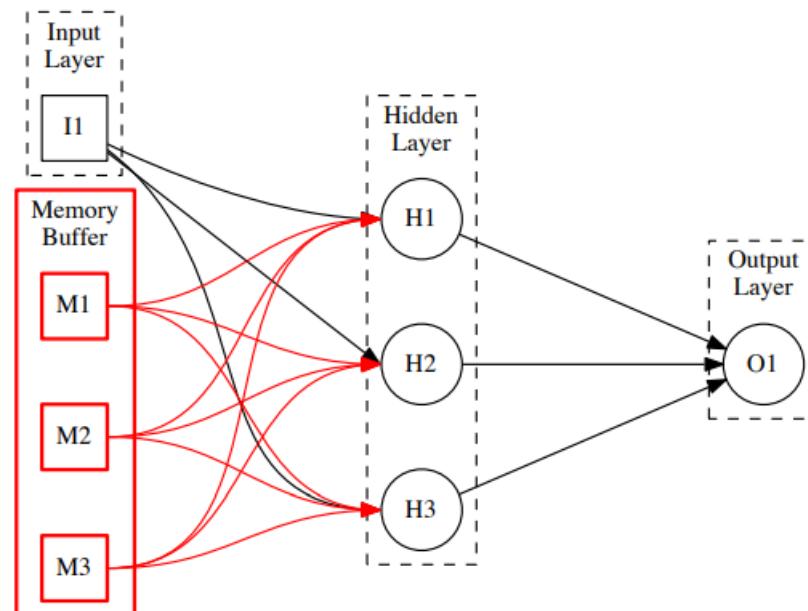


Figura 2.15: Fusionar el buffer de memoria con la siguiente entrada.

Por lo tanto a medida que se avanza por la secuencia se repite un ciclo de almacenar y utilizar el estado anterior en el siguiente instante de tiempo, figura 2.16.

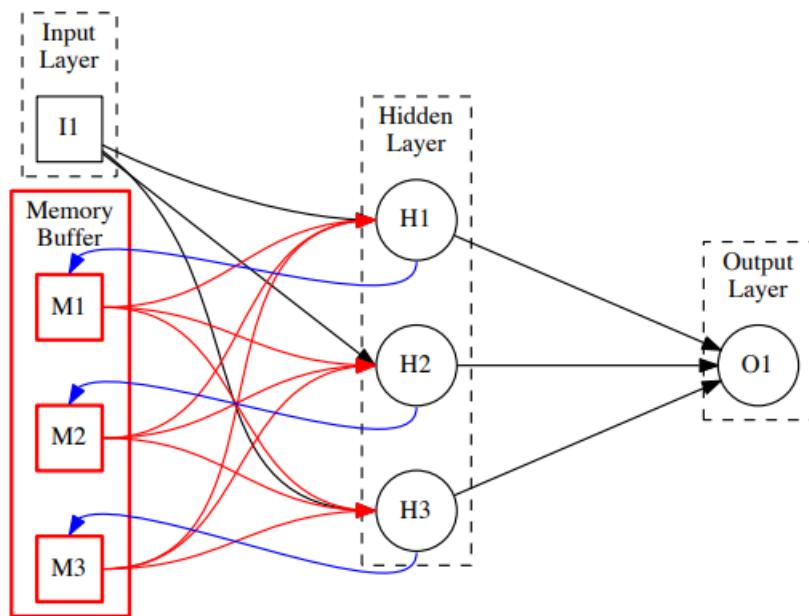


Figura 2.16: Ciclo de escritura en la memoria y fusión con la siguiente entrada

La Función matemática es la siguiente; dada una secuencia de entrada  $x = (x_1, \dots, x_T)$ , una red neuronal recurrente estándar calcula los vectores de la capa oculta  $h = (h_1, \dots, h_T)$  y la secuencia de vectores de salida  $y = (y_1, \dots, y_T)$  iterando las ecuaciones mostradas desde el tiempo 1 hasta T queda:

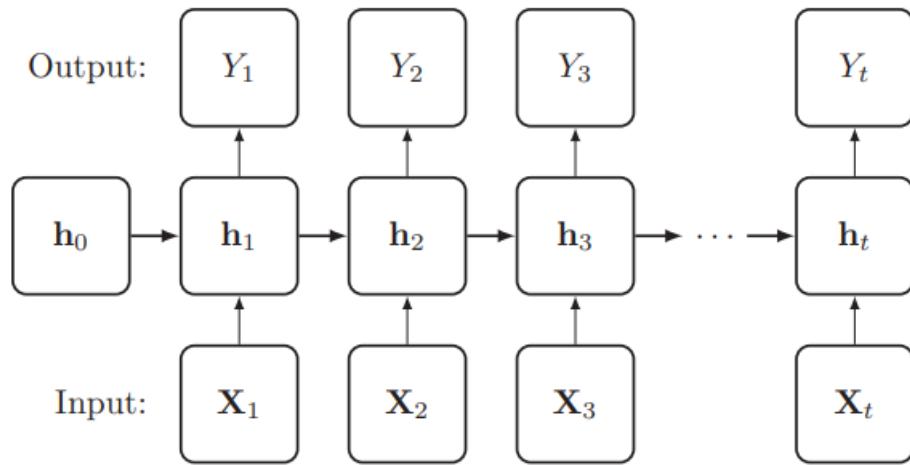


Figura 2.17: Una Red RNN de una sola neurona desenrollada en el tiempo

$$\begin{aligned} y_t &= W_{hy}h_t + b_y \\ h_t &= H(W_{xh}x_t + W_{hh}h_{t-1} + b_h) \end{aligned}$$

Donde  $W$  es la matriz de los pesos,  $b$  son los vectores de sesgo y  $H$  es la función de la capa oculta.(Graves, Mohamed, y Hinton, 2013)

## 2.2.6. Limitaciones de las RNN

Las Redes Neuronales Recurrentes o sus siglas en inglés RNN(Recurrent Neural Network) han sido adoptadas ampliamente en áreas de investigación relacionadas con datos secuenciales, como el texto, audio, videos. Sin embargo estas redes poseen un problema que afecta a los resultados de las predicciones, esto debido a que los algoritmos utilizados para actualizar los pesos se basan principalmente en el gradiente, esto conduce a que existen problemas de Vanish Gradient, esto afecta a los valores que son Recordados.<sup>a</sup> largo plazo, mientras mayor sea la base de datos y por lo tanto mayor información enviada a las entradas, esto hará más ineficaz la capacidad de aprender de la información que sea relevante.(Kelleher, 2019)

Para resolver esto se darán a conocer en este proyecto dos estructuras que resuelven esta problemática, Long-Short Term Memory(LSTM) y Gated Recurrent Unit(GRU)

## 2.2.7. LSTM(Long-Short Term Memory)

Para manejar las dependencias a largo plazo, en 1997 Hochreiter y Schmidhuber propusieron la celda LSTM, esta celda mejora en gran medida una RNN estándar, esto se debe al agregar la compuerta(o puerta) a la celda. Desde ese primer acercamiento a esta nueva celda las LSTM se han ido modificando y mejorando a través de varias investigaciones, más concretamente en el año 2001.(Gers, Schmidhuber, y Cummins, 2000)

La propiedad de las LSTM que les permite propagar las activaciones a lo largo de periodos de tiempo extensos les permite procesar secuencias que incluyen dependencias a gran distancia, con esto me refiero a interacciones que suceden entre elementos que están separados por dos o más posiciones). Por ejemplo la dependencia que existe en la siguiente frase entre el sujeto y el verbo: "El perro de esa casa es agresivo", sin un recuerdo.<sup>a</sup> largo plazo una RNN estándar no recordaría(o tendría dificultades) que sujeto se considera agresivo.(Yu, Si, Hu, y Zhang, 2019)

### LSTM con una Compuerta de olvido

Esta arquitectura, posee tres compuertas, una compuerta para "Olvidar"(Forget), una de entrada(Input) y otra de Salida(Output). La clave de esta estructura es la celda de estado, donde se almacenan los cambios para recordar u olvidar cierta información de menor a mayor importancia.

Compuerta de Entrada ( Input): Esta controla cuales activaciones en la celda deberían ser actualizadas en respuesta a una nueva entrada. Compuerta del Olvido (Forget): Esta cumple la labor de determinar las activaciones de la celda deben ser olvidadas en cada paso del tiempo. Compuerta de Salida (Output): Controla que activaciones deberían ser generadas en la salida en respuesta a la entrada actual.(Kelleher, 2019)

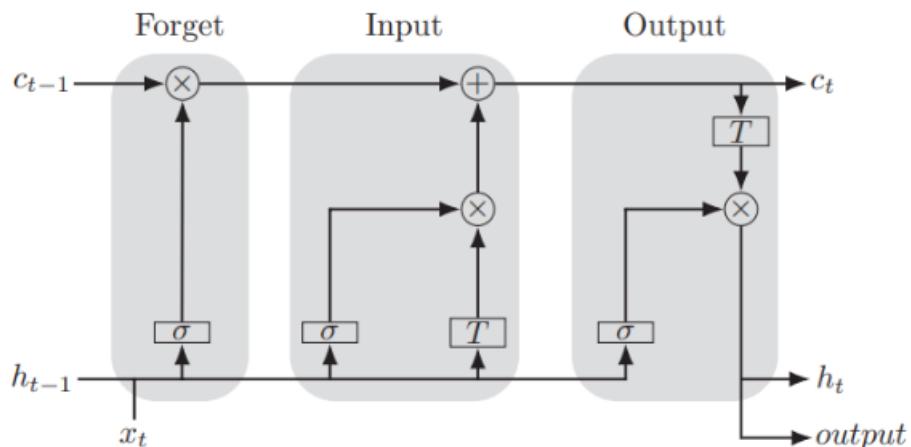


Figura 2.18: Arquitectura de una LSTM con una compuerta de olvido

En esta figura se muestra la estructura interna de una LSTM. En donde cada flecha en esta imagen indica un vector de activación, la celda cruza en la parte superior de  $c_{t-1}$  a  $c_t$ . Los valores en esta celda pueden tomar valores entre -1 a 1. En el procesamiento de una entrada, el vector de entrada  $x_t$  es sumado al vector de estado oculto en el tiempo anterior,  $h_{t-1}$ .

La manera en que se procesa es de izquierda a derecha , comenzando por la compuerta Forget , lo que hace esta es pasar el vector de la suma del input  $x_t$  y el estado oculto  $h_{t-1}$  a una función de activación sigmoide. Como resultado la salida de la compuerta Forget es un vector de rango entre

0 y 1. Luego de obtener el resultado la celda de estado es multiplicada por este vector Forget. El resultado de esta multiplicación provoca que los valores cercanos a 0 sean olvidados y valores cercanos a 1 sean recordados. Esto actúa como un filtro.

La compuerta de Input decide cuál información será agregada a la celda de estado. Este proceso se divide en dos partes, la primera decide qué valores en la celda de estado deben ser actualizadas y la otra parte qué información debe incluirse en esta actualización. La primera parte se realiza con una suma de la entrada  $x_t$  y el estado oculto  $h_{t-1}$  pasada a través de una función de activación sigmoide. La segunda parte consiste en el mismo vector anterior  $x_t + h_{t-1}$  pero esta vez por una función de activación tangente hiperbólica. Tanh es utilizado para que el rango de los valores a actualizar estén entre -1 a 1 y de esta manera evitar overfitting. Una vez hecho esto se procede a hacer una multiplicación de los dos vectores, es resultado es añadido a la celda de estado.

En la Compuerta Output se decide qué elementos saldrán como respuesta de los elementos ingresados. Se crea un vector de Output candidato que es pasado a una capa de Tanh. Además se vuelve a crear otro filtro. El vector de la salida candidata se multiplica con el Filtro, con esto se logra obtener el resultado de la capa de salida y pasarlo al siguiente estado oculto.

La Expresión matemática de la figura es la siguiente:

$$\begin{aligned} f_t &= O(W_{fh}h_{t-1} + W_{fx}x_t + b_f), \\ i_t &= O(W_{ih}h_{t-1} + W_{ix}x_t + b_i), \\ ce_t &= \tanh(W_{ceh}h_{t-1} + W_{cex}x_t + b_{ce}), \\ c_t &= f_t c_{t-1} + i_t ce_t, \\ o_t &= O(W_{oh}h_{t-1} + W_{ox}x_t + b_o), \\ h_t &= o_t \tanh(c_t). \end{aligned}$$

$C_t$  es referente a la celda de estado de la LSTM.  $W_i, W_c$  y  $W_o$  son los pesos y el operador "X" es la multiplicación puntual entre dos vectores. Cuando se realiza una modificación a la celda de estado, la compuerta que entra en acción es la compuerta input la cual decide qué información nueva será agregada a la celda de estado, la compuerta forget decide qué información será desechada de la celda de estado y la compuerta output decide qué información puede salir.

## 2.2.8. Gated Recurrent Unit(GRU)

La capacidad de aprendizaje de las LSTM es superior a una RNN estándar. Sin embargo los parámetros que se le agregan implican una carga computacional mayor.

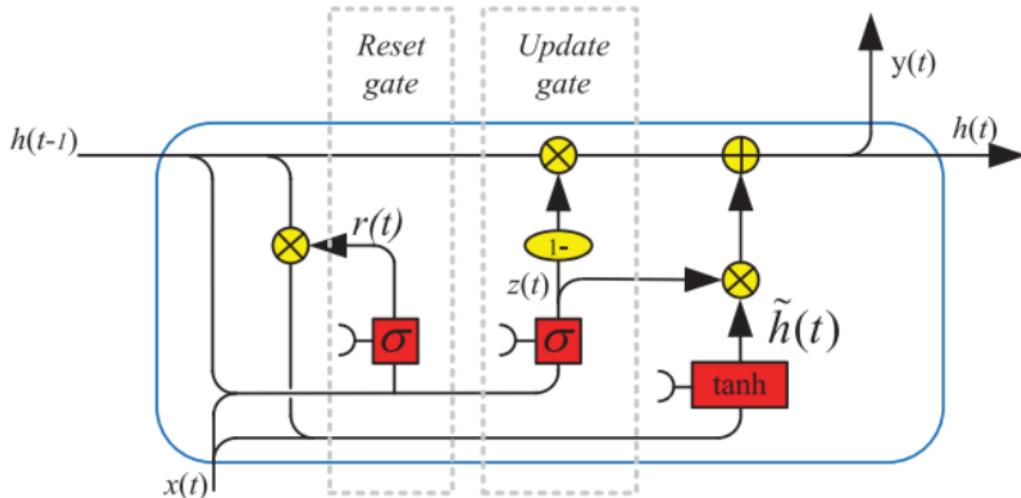


Figura 2.19: Arquitectura de una GRU

$$\begin{aligned}
 r_t &= O(W_{rh}h_{t-1} + W_{rx}x_t + b_r), \\
 z_t &= O(W_{zh}h_{t-1} + W_{zx}x_t + b_z), \\
 h * t &= \tanh(W_{h*h}(r_t h_{t-1}) + W_{h*x}x_t + b_z), \\
 h_t &= (1 - z_t)h_{t-1} + z_t h * t.
 \end{aligned}$$

Teniendo en cuenta esta deficiencia en las LSTM, lo que hacen las GRU para solventar es reducir el número de parámetros, a través de integrar la compuerta de Forget y la compuerta Input de las LSTM en una compuerta llamada Update(actualización). Las GRU poseen dos compuertas: Update y Reset(Reinicio). Se pueden considerar las GRU como variantes de las LSTM. En cuanto a su desventaja es que una GRU es menos potente que una LSTM estándar u original.

### Funciones de activación

Para que una red neuronal tenga un comportamiento no lineal es necesario aplicar una función de activación para que la red tenga un comportamiento dinámico y sea capaz de extraer información compleja de los datos.

Las entradas en una red son consideradas las partes más importantes de la red, estos resultados se procesan para obtener un resultado de salida llamado activación(función umbral), esta es una transformación de la escala. Permite que los datos estén en un rango limitado.(Sharma, Sharma, y Athaiya, 2017)

- Sigmoid

La función sigmoide transforma los valores en un rango entre 0 y 1. Se define como:

$$f(x) = \frac{1}{e^{-x}} \quad (2.5)$$

Su expresión derivada seria:

$$f'(x) = 1 - \text{sigmoide}(x) \quad (2.6)$$

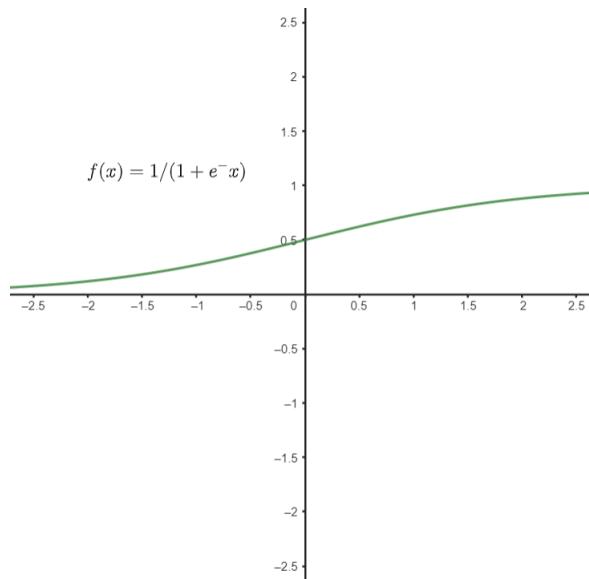


Figura 2.20: Función de activación sigmoide

Esta función no es simétrica con respecto al origen, esto implica que los valores de salida de las neuronas serán iguales.

- Tangente Hiperbólica

La función hiperbólica o Tanh es similar a la función sigmoidea pero a diferencia de ésta, la función tanh es simétrica respecto al origen, esto afecta en los resultados de salida que en este caso si poseen números negativos, la función se define como:

$$f(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}} \quad (2.7)$$

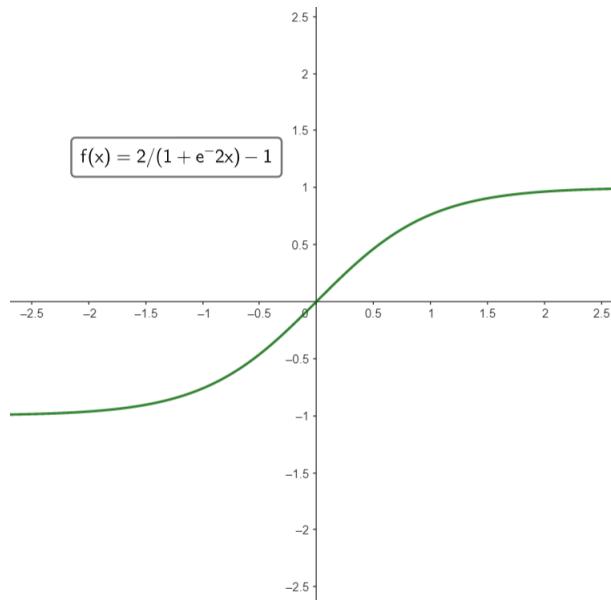


Figura 2.21: Función de activación tangente hiperbólica

La función Tanh es continua y diferenciable, los valores están en una rango entre -1 y 1.

- Relu

Rectified linear unit(unidad de línea rectificada) o Relu, es una función de activación no lineal. En la función Relu no se activan todas las neuronas, esto debido a qué una neurona se desactiva cuando una salida de transformación lineal es cero, la función se define como:

$$f(x) = \max(0, x) \quad (2.8)$$

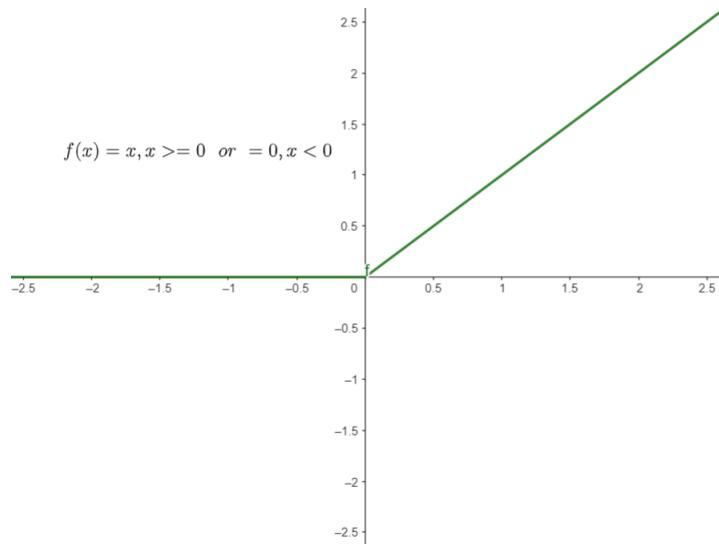


Figura 2.22: Función de activación ReLU

Existen casos en el cual el gradiente es cero, por lo que los pesos y sesgos no se actualizan.

## 2.2.9. Optimizador

Los algoritmos de optimización se definen por sus reglas de actualización, estas reglas se rigen por los hiper-parámetros tales como el Learning rate(velocidad de aprendizaje).(Choi y cols., 2019)

- Descenso de gradiente estocástico(SGD)
- Root Mean Square Propagation(RMSprop)
- Estimación Adaptativa del Momento(ADAM)

### Descenso de gradiente estocástico(SGD)

En esta optimización se reemplaza el vector de gradiente, que originalmente esta presenta en un descenso de gradiente, por una estimación estocástica de dicho vector, en el caso de este proyecto la estimación se refiere al gradiente del error para un punto de datos.

La siguiente función hace referencia al error de la red en instancia  $i$ .

$$f_i = l(x_i, y_i, w)$$

En este optimizador se actualizan los pesos de acuerdo con el gradiente sobre  $f_i$ .

$$w_{t+1} = w_t - y_t \Delta f_i(w_t)$$

donde:

- $w_{t+1}$  es el valor actualizado luego de la iteracion t
- $w_t$  es el valor inicial antes de la iteracion t+1
- $y_t$  es el learning rate

### Root Mean Square Propagation(RMSprop)

Root Mean Square Propagation el funcionamiento de este optimizador consiste en normalizar el gradiente con su respectiva raíz cuadrada del valor medio de los cuadrados.

$$v_{t+1} = \alpha v_t + (1 - \alpha) \Delta f_i(w_t)^2$$

$$w_{t+1} = w_t - y \frac{\Delta f_i(w_t)}{\sqrt{v_{t+1}} + \epsilon}$$

Donde:

- $y$  es el learning rate
- $\epsilon$  es un valor muy pequeño utilizado para evitar errores producidos al dividir por 0
- $v_{t+1}$  es la estimación del momento  $t + 1$

### Estimación Adaptativa del Momento(ADAM)

Adam es una mezcla entre RMSprop mas Momento

$$m_{t+1} = \beta m_t + (1 - \beta) \Delta f_i(w_t)$$

$$v_{t+1} = \alpha v_t + (1 - \alpha) \Delta f_i(w_t)^2$$

$$w_{t+1} = w_t - y \frac{m_t}{\sqrt{v_{t+1}} + \epsilon}$$

Donde  $m$  y  $v$  representan dos momentos,  $m$  es el que modela la media de los gradientes a través del tiempo y  $v$  la varianza de los gradientes a través del tiempo.

## 2.2.10. Validación cruzada

La validación cruzada es un método de muestreo o revisión de datos con el objetivo de evaluar qué capacidad de que los modelos predictivos funcionen de manera generalizada y por lo tanto no se sobreajuste.

El problema al construir un modelo muy robusto o perfecto es que como resultado este modelo se ajuste demasiado bien al entrenamiento hecho a los datos, provocando un sobreajuste y por lo tanto teniendo dificultades a la hora de detectar datos no vistos anteriormente. Hay que tener en cuenta que cuando ajustamos tanto los datos no solo afecta a las variables dependientes e independientes sino también en el ruido existente es los datos.

Por el contrario si creáramos un modelo más simple, esta estaría menos afectada por el ruido en los datos, pero no encontraría una relación fuerte entre las variables dependientes e independientes, este modelo estaría infraajustado.

El punto en un modelo en estar entre los dos puntos, encontrando un equilibrio. Para evitar esto se utilizar técnicas de muestreo con es en este caso para el proyecto se hará uso de k-fold validación cruzada.

### K-fold cross-validation

La validación cruzada es una técnica de muestreo que se realiza de tal manera que no se solapen dos conjuntos de prueba. Esta validación consiste en dividir el conjunto de datos en  $k$  subconjuntos del mismo tamaño o similar. La división de los datos se realiza de manera aleatoria(muestreo aleatorio). La cantidad de subconjuntos que se entrena es  $k-1$  y el subconjunto restante se utiliza para la validación, esta mide el rendimiento del entrenamiento. Este proceso de entrenamiento se repite hasta que cada uno de los  $k$  subconjunto se haya utilizado como subconjunto de validación. Una vez hecho esto se obtiene un promedio de las validaciones.(Berrar, 2019)

En la siguiente figura se muestra el comportamiento, con un  $k = 10$ , esto quiere decir que los valores se validan 10 veces, en el primer fold este sirve como subconjunto de validación  $D_{val,1}$  y los 9 restantes funcionan como entrenamiento  $D_{train,1}$ . En el segundo fold el  $k=2$  servirá de validación  $D_{val,2}$  y los restantes 9 subconjuntos serán entrenamiento  $D_{train,2}$ .

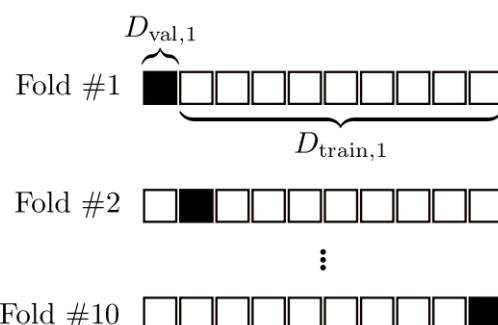


Figura 2.23: Validación cruzada con k-fold=10

## 2.3. Estado del Arte

### 2.3.1. Sistemas de reconocimiento de gestos con visión por computadora

#### ■ Sistema de reconocimiento gestual de señas chilena mediante cámara digital

En este proyecto resuelve parte de la problemática planteada, centrándose exclusivamente en los gestos denominados estáticos(Riveros y Inostroza, 2016), esto por medio de visión por computadora e inteligencia artificial, utilizando como método de entrada de datos un dispositivo android.

En lo referente a preprocesamiento de las imágenes se utilizó la Manipulación de contraste para incrementar el cambio de luminosidad entre las zonas más oscuras o más claras de una fotografía, Eliminación de ruido en referencia a valores distorsionados, debido a la cámara o al medio de transmisión de la señal representado como píxeles aislados y Realce de Bordes que se utiliza para transformar una imagen de manera que exhibe solo el detalle de bordes, estos contornos se utilizan para el análisis de imágenes para el reconocimiento de gestos.

Además para el reconocimiento de los gestos se hace uso de la detección del fondo y de la mano por separados, es una solución para el fondo irregular de la imagen, se detecta el fondo antes de enfocar la mano para posteriormente detectar la mano, a partir de estos dos datos se obtiene una mano sin fondo.

#### ■ Un sistema de reconocimiento de ASL en tiempo real utilizando sensores de movimiento Leap.(Fok, Ganganath, Cheng, y Chi, 2015)

En este trabajo se utilizan múltiples sensores para capturar el gesto de una mano desde diferentes ángulos de visión. A partir del conjunto de datos proporcionados por los distintos sensores, se puede calcular un gesto de la mano fusionada, esto a través de un algoritmo de fusión. Además para el reconocimiento de gestos se utilizó un modelo oculto de Markov(es una técnica estadística que ha sido ampliamente adoptada en muchos sistemas de reconocimiento de patrones) con múltiples sensores de profundidad(Leap Motion). En los resultados se obtuvo que los datos fusionados de 2 sensores dieron una tasa de reconocimiento mínima del 84,68 %, mientras que con un solo sensor se obtuvo un mínimo del 68.78 %.

También existen trabajos más cercanos al trabajo que se pretende realizar, como en **Reconocimiento dinámico de gestos en el Internet of Things**,(Li, Wu, Jiang, Xu, y Liu, 2018) donde se propone un método de reconocimiento dinámico de gestos basado en el HMM(modelo oculto de markov). Dado que los gestos dinámicos complejos implican transformaciones temporales y espaciales, el movimiento de la mano puede ser identificado por la información de movimiento y los cambios del gesto. El modelo de parámetros del HMM(modelo oculto de markov) se entrena mediante el algoritmo de backpropagation y forward propagation.

Este documento divide un gesto dinámico completo en cambios de gesto y cambios en las trayectorias de los gestos. Este trabajo utiliza Kinect para recoger diez tipos de muestras de gestos de diferentes personas, los números del 0 al 9. Para las trayectorias de movimiento de los gestos, esta sección define diez tipos de trayectorias de movimiento, como arriba, abajo, izquierda, derecha en el sentido de las agujas del reloj, tiempo inverso, arriba a la izquierda, abajo a la izquierda, derecha arriba y derecha abajo. Los gestos definidos y las trayectorias. Las trayectorias de movimiento definidas se combinan para formar un gesto combinado.

### 2.3.2. Sistemas de reconocimiento de gestos con alternativas a visión por computadora

Además del reconocimiento de los gestos por medio de visión por computadora también existen proyectos que hacen el uso de guantes para el reconocimiento de los gestos.

#### ■ **Glove-Based Sign Language Recognition Solution to Assist Communication for Deaf Users**

Esta investigación realizada por (López-Noriega, Fernández-Valladares, y Uc-Cetina, 2014) ayuda a la comunicación de los sordomudos mediante la identificación de la posición de los dedos de la mano con guantes 5DT, este guante tiene 5 sensores de presión que miden la presión entre el nudillo y la primera articulación de cada dedo. También existen otros proyectos que utilizan otro tipo de guantes(Universidad de Sonora's Project ) de diseño propio, (Fiel y cols., 2013). La topología que se utilizó fue un perceptrón multicapa con cinco neuronas de entrada recibe la información de los cinco sensores dactilares, esta información se pasa por tres capas ocultas de veinte neuronas cada una. La capa de salida está formada por veintiséis neuronas binarias, una para cada letra del alfabeto de signos. El reconocimiento de los gestos se lleva a cabo mediante una red neuronal probada con cinco algoritmos de entrenamiento diferentes: Backpropagation, Quick propagation, Resilient backpropagation, Manhattanpropagation y the Conjugate gradient method, los algoritmos escogidos fueron aquellos con un valor del error medio era inferior a 0,001, Backpropagation, Quick propagation y Manhattanpropagation.

En este proyecto se ha demostrado que los perceptrones multicapa perceptrones con entradas binarias, salidas binarias y tres capas ocultas con funciones de activación sigmoidales es una configuración robusta para el reconocimiento de patrones.

#### ■ **Simulación y evaluación de técnicas de clustering para el reconocimiento de gestos estáticos en la traducción de la lengua de señas peruana**

En este proyecto de (Espinoza Hoyos, 2020) simuló redes neuronales artificiales y evaluó su proceso de clustering(agrupamiento de conjuntos de objetos no etiquetados, para lograr construir subconjuntos de datos conocidos como Clusters) de vectores de datos digitales que serán captados por un guante. Con una cantidad de 24 señas estáticas del alfabeto peruano. Se utilizaron 3 parámetros de evaluación: Tabla de contingencia, Precisión y el Recall.

Estos guantes electrónicos implementados poseen sensores flex ubicado en cada uno de los dedos imitando las falanges de los dedos( Huesos de los dedos) un Modulo (MPU6050) que capta el movimiento sobre este sensor en conjunto con un arduino que se encarga de la lectura de los sensores.

Los resultados de este proyecto fueron una precisión y recall de 98.14 % y 97.43 % como valores máximos y 90.25 % y 88.05 % como valores mínimos.

# Capítulo 3

## Formulación del Proyecto

### 3.1. Objetivos

Desarrollar un modelo de reconocimiento automático para la identificación de gestos dinámicos en la lengua de señas chilena a través de algoritmos de visión por computadora utilizando imágenes estéreo e información de profundidad de objetos.

#### 3.1.1. Específicos

1. Recolectar fuentes de datos para la definición de un dataset de gestos/señas en el idioma español chileno.
2. Definir modelo de aprendizaje automático para la identificación de gestos/señas estáticas y dinámicas
3. Analizar el rendimiento del modelo, a través de la experimentación utilizando datasets de lengua de señas de otros países, para la validación de la calidad de la identificación en gestos/señas dinámicos en la lengua de señas chilena.

## 3.2. Metodología de Trabajo

**En el objetivo específico 1 :** Recolectar fuentes de datos para la definición de un dataset de gestos/señas en el idioma español chileno"se definen las siguientes actividades.

- A1** Formulación de Proyecto de Título.
- A2** Revisión de la literatura respecto al proyecto de Título.
- A3** Desarrollo del Marco teórico.
- A4** Escritura del Informe de Título.
- A5** Definir conjunto de datos a buscar en la Lengua de señas chilena.
- A6** Buscar imágenes y videos de los gestos de la LSCH en fuentes de datos chilenas.
- A7** Buscar en la literatura similitudes con Lenguas de señas de otros países.
- A8** Buscar en fuentes de datos extranjeras ya previamente identificadas.
- A9** Crear los gestos de la lengua de señas chilena que no se hayan encontrado en otros países, o para complementar los gestos ya encontrados, a través de una recopilación fotográfica.
- A10** Definición del conjunto de datos definitivo para trabajar en el modelo de visión por computadora.

**En el objetivo específico 2:** Definir modelo de aprendizaje automático para la identificación de gestos/señas estáticas y dinámicas"se definen las siguientes tareas.

- A1** Investigar la diferencia entre señas/gestos dinámicos y estáticos para ver que modelo sea adecuado a ocupar.
- A2** Adaptar una metodología para la comparación de diferentes modelos por visión por computadora dependiendo de la calidad de su resultado.
- A3** Evaluar diferentes modelos de visión por computadora para evaluar los gestos estáticos y dinámicos.
- A4** Escoger el modelo a utilizar dependiendo de su desempeño en la clasificación de gestos estáticos y dinámicos.

**En el objetivo específico 3:** Analizar el rendimiento del modelo, a través de la experimentación utilizando datasets de lengua de señas de otros países, para la validación de la calidad de la identificación en gestos/señas dinámicos en la lengua de señas chilena"se definen las siguientes actividades.

- A1** Implementar la predicción de los gestos estáticos y dinámicos a través de la visión por computadora.
- A2** Obtener rendimiento del modelo realizado a través de diferentes métricas.
- A3** Realizar comparación de rendimiento con otros proyectos realizados en países extranjeros.
- A4** Recopilación de los resultados de las predicciones y la retroalimentación desde el usuario.

### 3.2.1. Planificación

A continuación, se presenta el diagrama de Carta Gantt que incorpora todas las tareas asignadas en la Metodología de Trabajo junto con su tiempo estimado a la ejecución:

OBJETIVOS Y ACTIVIDADES	FECHA DE INICIO	FECHA DE TERMINO	DIAS DE DURACIÓN	ABRIL				MAYO				JUNIO				JULIO				AGOSTO			
				S1	S2	S3	S4	S1	S2	S3	S4	S1	S2	S3	S4	S1	S2	S3	S4	S1	S2	S3	S4
<b>OB1</b>																							
A1	1/04/2021	31/05/2021	43																				
A2	1/05/2021	31/07/2021	65																				
A3	1/06/2021	31/08/2021	66																				
A4	1/07/2021	31/12/2021	89																				
A5	1/09/2021	1/09/2021	1																				
A6	2/09/2021	3/09/2021	2																				
A7	3/09/2021	4/09/2021	1																				
A8	4/09/2021	7/09/2021	2																				
A9	8/09/2021	22/09/2021	11																				
A10	23/09/2021	7/10/2021	11																				
<b>OB2</b>																							
A1	1/10/2021	2/10/2021	1																				
A2	3/10/2021	3/11/2021	23																				
A3	4/11/2021	30/11/2021	19																				
A4	1/12/2021	12/12/2021	8																				
<b>OB3</b>																							
A1	8/11/2021	28/11/2021	15																				
A2	29/11/2021	16/12/2021	14																				
A3	17/12/2021	22/12/2021	4																				
A4	23/12/2021	31/12/2021	7																				

Figura 3.1: Carta Gantt hasta Agosto

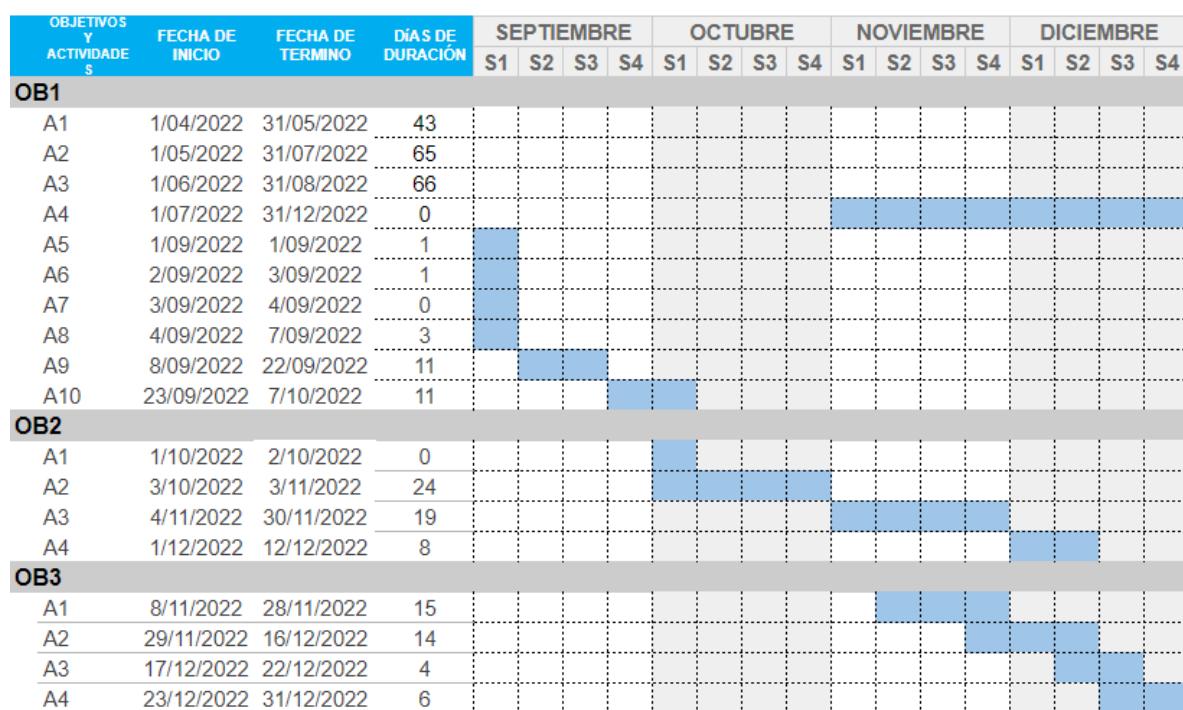


Figura 3.2: Carta Gantt hasta Diciembre

La primera parte del proyecto se realizó en el primer semestre del año 2021 de Abril a agosto, mientras la segunda parte se continuó en el segundo semestre del año 2022 desde los meses de Septiembre a Diciembre

## Capítulo 4

# Búsqueda del Conjunto de Datos

### 4.1. Conjunto de datos de gestos estáticos

Hoy en día en Chile existen varios proyectos de reconocimiento de gestos, estos proyectos están a disposición de las personas para comprender qué métodos utilizaron para su realización, qué tecnología se usó referente a la captura de las imágenes, etc, sin embargo es diferente cuando se refiere a los datos utilizados para el entrenamiento del modelo, ya que la mayoría de estos datasets no se encuentran a disposición para utilizados ya sea porque aún están en proceso de realización o por propia decisión del autor,(se encontró un dataset chileno pero la cantidad de datos era insuficiente para realizar pruebas), por lo que se procedió a utilizar Datasets de diferentes países para construir uno de la Lengua de Señas chilena.

- ".ASL Alphabet", colgado por Akash en Kaggle.
- ".ASL Alphabet Test", colgado por Dan Rasband.
- ".American sign language alphabet static", colgado por Jordi Viader.
- "Lengua de señas mexicana", Erik Ibarra.
- "Lengua de Señas Colombiana", colgado por Kaggle Kerneler en Kaggle.
- "Lengua de Señas Chilena", realización personal.





Figura 4.2: Gesto A LSC

Para la realización de el nuevo dataset chileno se procedió a investigar las similitudes y diferencias con los datasets de otros países dando un enfoque mayor a países hispanohablantes debido a la mayor similitud en los gestos de la lengua de señas del abecedario, de lo cual se procedió a obtener los datos de los siguientes países: Estados Unidos, de donde procede el dataset original, México y Colombia. Los demás países se dejaron de lado ya sea debido a la falta de bases de datos que sean públicos o que ya se tenía datos suficientes de determinada letra del abecedario. A pesar de la búsqueda por en diferentes países la base de datos era bastante limitada para la realización del entrenamiento o simplemente no se tenía ningún dato de determinada letra, por lo que se procedió a la realización personal.

La realización del abecedario de la LSCH fueron los siguientes: D,E,F,K,M,N,P,Q,T,U, con alrededor de 3000 imágenes por gesto.

## 4.2. Dataset gestos Dinámicos

En cuanto a la recolección de los gestos dinámicos se hizo uso de la librería de google llamada "Mediapipe Holistic".

Esta librería integra varios modelos para su funcionamiento las cuales son:

- Human pose
- Face landmark
- Hand tracking

De las cuales en este proyecto se hará uso de **Human pose** y **Hand tracking**.

### 4.2.1. Human pose

Human Pose es una solución de machine learning para el seguimiento de la pose del cuerpo, con un total de 33 puntos de referencia 3D y segmentación de la máscara de fondo a partir de fotogramas de vídeo RGB. Este modelo utiliza dos pasos principales para su funcionamiento, **detector-tracker ML pipeline**, utilizada también en Hand tracking y **Mediapipe Face Mesh**. Lo primero que se hace es localizar a la persona/pose de la región que se tiene interés(o por sus siglas en inglés ROI) dentro de un marco definido. Una vez hecho esto el tracker predice los puntos de referencia de la pose, utilizando el recuadro de ROI como entrada.

### Modelo de detección para la pose/persona

Predice dos puntos claves que describen el centro de la persona, además de la rotación y la escala del cuerpo como un círculo. Se predice el punto medio de las caderas de una persona y el ángulo de inclinación entre los hombros y las caderas, descrito en la fig 4.3.

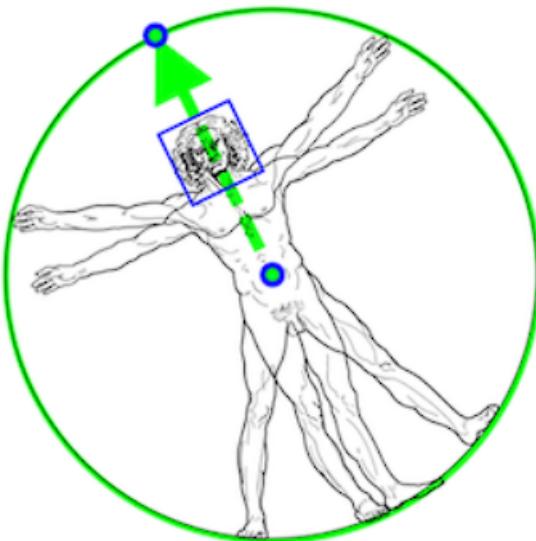


Figura 4.3: Hombre de Vitruvio alineado a través de dos puntos clave predichos por el detector BlazePose, incluye un recuadro que delimita la cara

### Puntos de referencia de Human Pose

Por parte de Human Pose son un total de 33 puntos de referencia, con cada uno de sus índices.

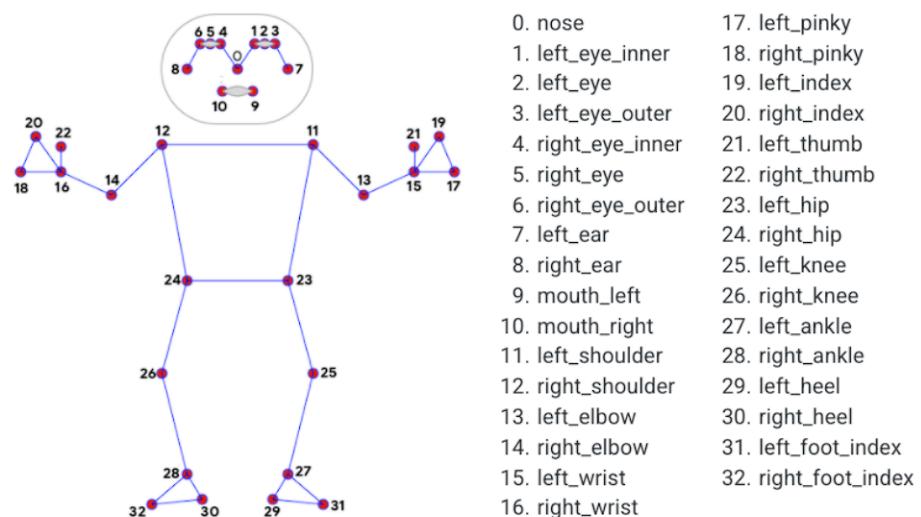


Figura 4.4: Puntos de referencia

### 4.2.2. Hand tracking

Hand tracking es una solución al seguimiento de las manos y dedos. Utiliza machine learning con 21 puntos de referencia 3D para cada una de las manos(42 puntos de seguimiento en total).

Este modelo es un conjunto de varios modelos, un modelo para la **detección de la palma** que está presente en la imagen completa y devuelve un cuadro que delimita la mano. Otro modelo de los **puntos de referencia de la mano** que opera en la región delimitada por el modelo anterior y te devuelve puntos de referencia 3D.

#### Modelo de Detección de la palma

Se entrena un detector de palma, ya que es más fácil detectar objetos rígidos(como la palma o la mano).

#### Modelo de puntos de referencia de la mano

Lo que hace este modelo es la localización de los 21 puntos de referencia 3D

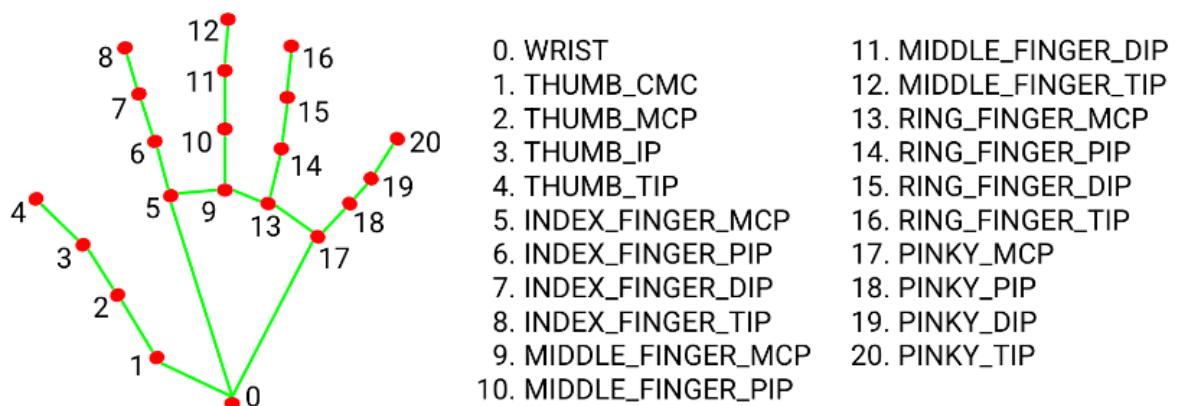


Figura 4.5: Puntos de referencia de la mano

### 4.2.3. Mediapipe Holistic

Una vez explicado su comportamiento por separado, la manera en que trabaja en conjunto es la siguiente:

Primero se estima la pose con detector de pose BlazePose y posterior el modelo de referencia. A partir de los puntos de referencia inferidos se derivan las demás regiones, en el caso de este proyecto son ambas manos. Luego se recorta las zonas de interés para aplicar los modelos mencionados con anterioridad para detectar sus puntos de referencia. El conjunto hace un total de 75 puntos de referencia.

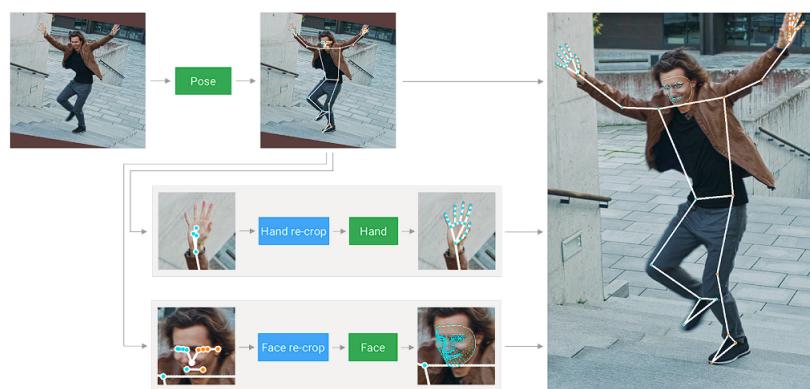


Figura 4.6: Mediapipe Holistic modelo general

#### Modelos de puntos de referencia

Mediapipe holistic utiliza modelos de pose con 33 puntos y hand con 21 puntos por cada mano

#### Modelos de recorte

Cuando la precisión del modelo pose no es el suficiente se ejecuta un modelo adicional de recorte de transformación espacial.

## 4.3. Descripción del Conjunto de datos recopilado

El conjunto de datos incluye tanto valores estáticos en concreto 21 gestos como gestos dinámicos los cuales serán 28. Con un total de 50 gestos, se agregó un gesto "None" para indicar que no se está realizando ningún gesto en particular. De los gestos dinámicos se toman en cuenta dos grupos, el grupo de gestos que hace uso de una sola mano y el grupo se utilizan ambas manos para realizar el gesto. Los gestos serán los siguientes:

- Abecedario
- Saludos
- Verbos
- Adjetivos

### 4.3.1. Abecedario

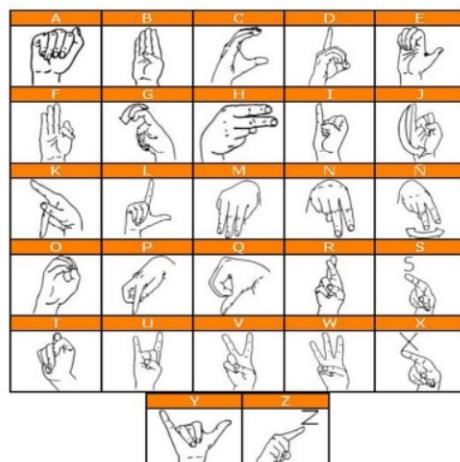


Figura 4.7: Alfabeto de la Lengua de señas chilena

### 4.3.2. Saludos



Figura 4.8: Hola



Figura 4.9: Chao

### 4.3.3. Verbos



Figura 4.10: Abrir



Figura 4.11: Comer



Figura 4.12: Caminar



Figura 4.13: Ayudar



Figura 4.14: Dibujar



Figura 4.15: Disculpar



Figura 4.16: Hablar



Figura 4.17: Planchar



Figura 4.18: Vender

### 4.3.4. Adjetivos



Figura 4.19: Flaco



Figura 4.20: Bonito

Datasets Gestos	
Estáticos	
A	
B	
C	
D	
E	
F	
H	
I	
K	
L	
M	
N	
O	
P	
Q	
R	
T	
U	
V	
W	
Y	

Tabla 4.1: Datasets Gestos Estáticos

Datasets Gestos	
Dinámica	una mano
G	
J	
Ñ	
S	
X	
Z	
Hola	
Chao	
Comer	
Caminar	
Flaco	
Bonito	
Alto	
Bajo	

Tabla 4.2: Datasets Gestos Dinámicos una mano

Datasets Gestos	
Dinámica	dos mano
Hablar	
Ayudar	
Abrir	
Disculpar	
Vender	
Dibujar	
Planchar	
Amigo	
Responsable	
Arrendar	
Chocar	
Vestir	
Estudiar	
Jugar	

Tabla 4.3: Datasets Gestos Dinámicos dos manos

## Capítulo 5

# Modelo de Reconocimiento de Señas Dinámicas

### 5.1. Estructura propuesta

En esta sección se propondrá y creo una estructura para el funcionamiento del modelo, en primera instancia se utilizo un modelo RNN-LSTM para crear la base de las pruebas de los diferentes hiper-parámetros y modelos que serán propuestos.

En esta primera estructura se tomaron valores por defecto, los cuales serán:

- Optimizador: Adam
- Velocidad de aprendizaje(Learning rate): 0,001
- Función de activación: Tangente hiperbólica

Una vez definida esta configuración inicial se procedió a escoger los parámetros que varían en el modelo para de esta manera conseguir un valor óptimo. Estos hiper-parámetros a utilizar son la cantidad de capas ocultas y de neuronas hay por cada capa.

En la fig 5.1 se muestra los aspectos que considero para encontrar el/los modelo con mejores resultados.

Se encogieron los valores por defecto , a partir de ahí se fueron actualizando los valores para encontrar los mínimos de la función objetivo.

### 5.1.1. Primera estructura

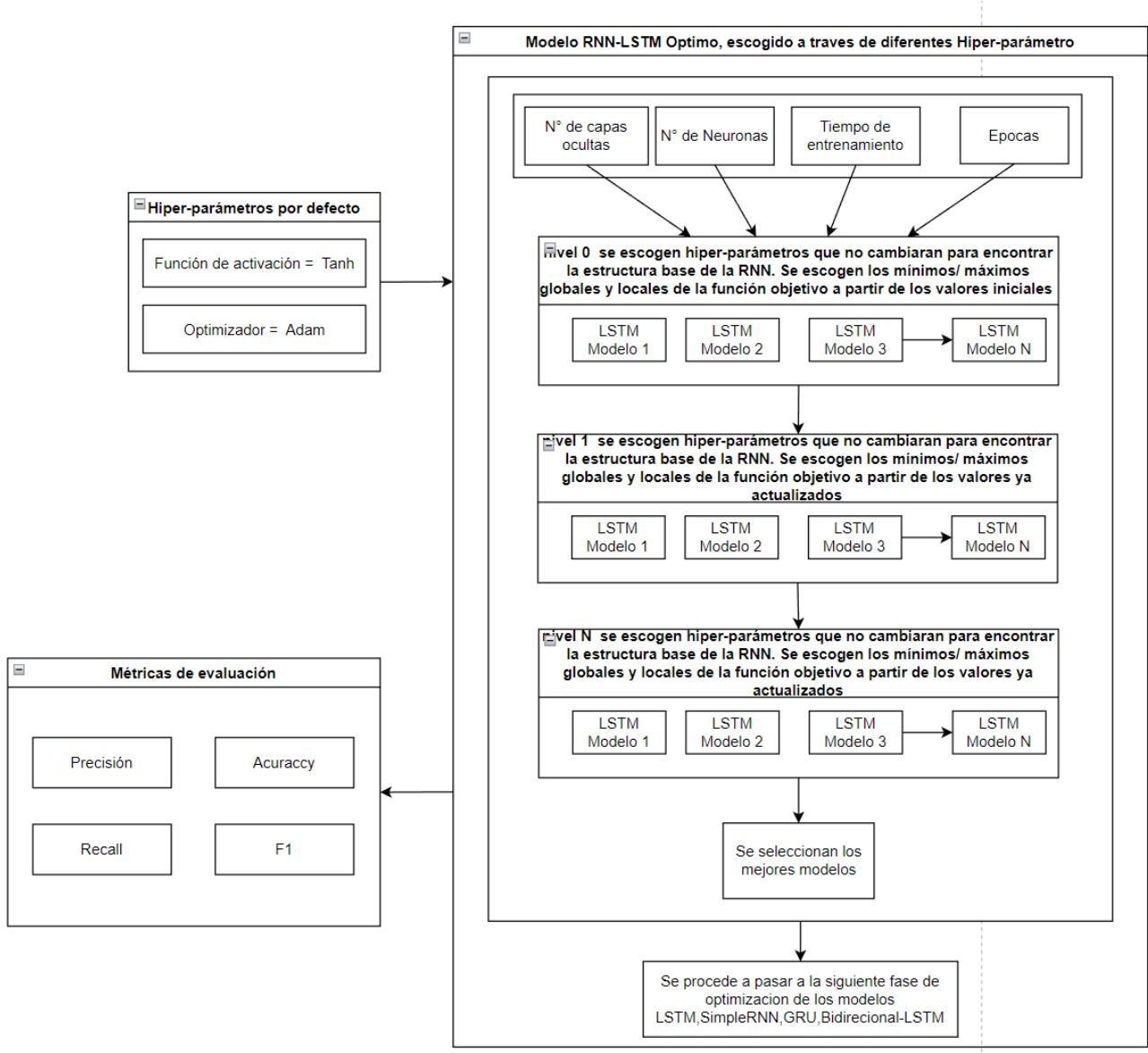


Figura 5.1: Estructura para la obtención de los mejores resultados de un modelo RNN-LSTM

### Capas y Neuronas

A continuación se muestra la manera en que se agregaron las capas y neuronas para realizar las pruebas de rendimiento del modelo.

Estructura		
Modelo	Capas ocultas	Neuronas
LSTM	1	4;8;16;32;64;128
LSTM	2	16,16;32,32;32,64;64,32;64,64;64,128;128,64;128,128
LSTM	3	32,64,32;32,64,64;32,64,128;64,64,32;64,64,64;64,64,128
LSTM	2 LSTM, 1 Densa	32,64,32;32,64,64;32,64,128;64,64,32;64,64,64;64,64,128

Tabla 5.1: Estructura

En cuanto a las neuronas se irán aumentando en valores de  $2^n$  hasta que su rendimiento vaya empeorando o el tiempo de entrenamiento sea demasiado elevado, por lo tanto donde no exista una mejora en el modelo. Cuando se encuentre este punto de inflexión se comenzó nuevamente desde una cantidad de neuronas menores, pero en este caso se agregara una capa de RNN-LSTM o una capa densa. Con un total de 26 modelos a entrenar y comprobar su rendimiento.

### 5.1.2. Segunda estructura

Una vez se realizaron las pruebas para obtener los mejores modelos RNN, se procederá a una comparación con la estructura de capas y neuronas de los modelos obtenidos, esta vez utilizando los siguientes 4 modelos: LSTM, GRU, SimpleRNN y bidireccional LSTM, escogiendo el que proporcione mejor precisión y tiempo de entrenamiento.

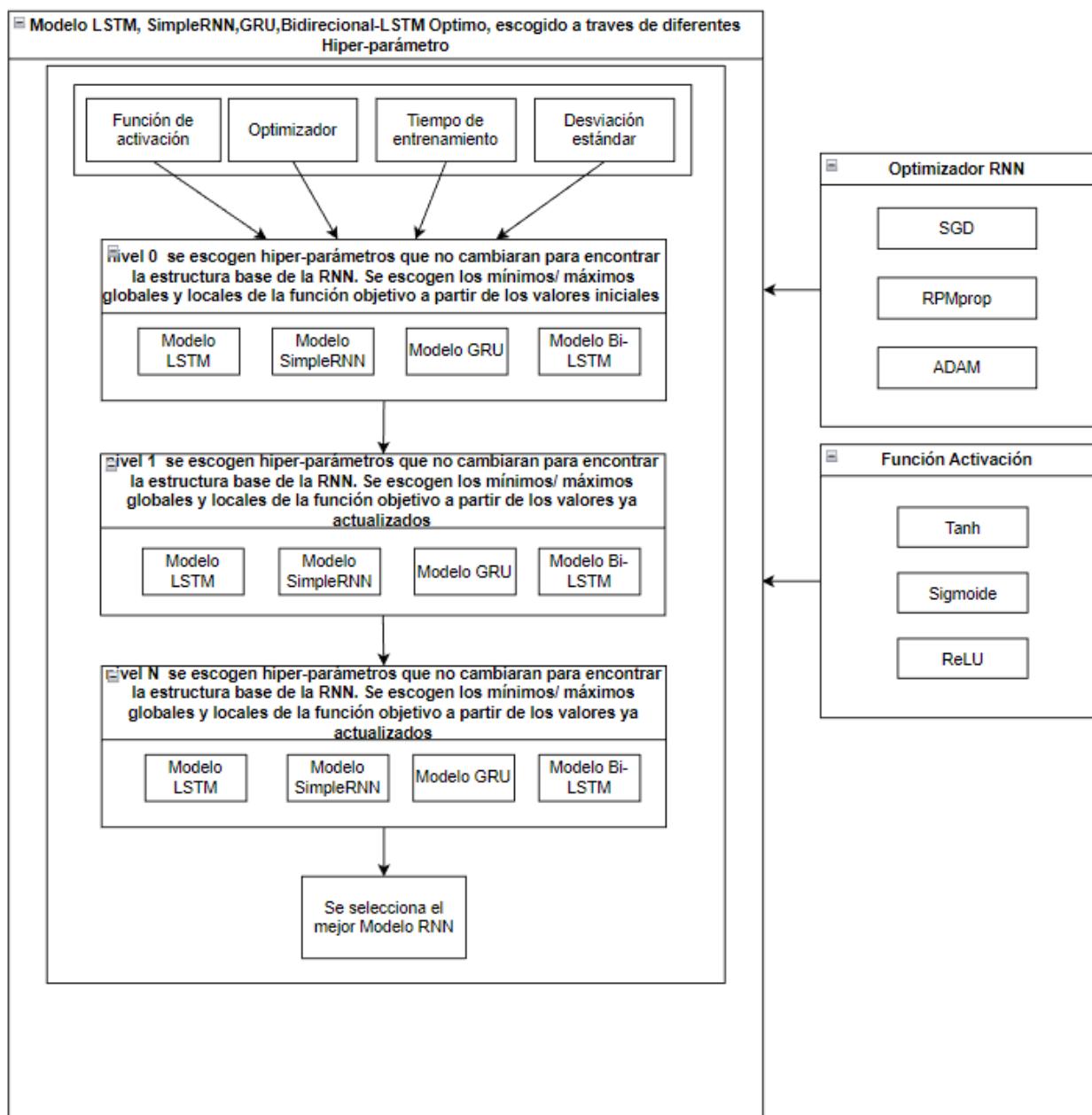


Figura 5.2: Estructura para la obtención del modelo final

# Capítulo 6

## Experimentación

### 6.0.1. Equipo

Para la obtención de los gestos dinámicos se hizo uso de la librería mediapipe holistic, esta librería fue ocupada en un ordenador portátil con las siguientes características.

- **Sistema operativo:** Windows 10 home single
- **Procesador:** AMD Rysen 5 3550H
- **Núcleos:** 4 nucleos 8 hilos
- **Velocidad de procesador:** 2.1 GHz
- **Ram:** 16384MB ram
- **DirectX:** DirectX 12
- **Cámara:** Cámara 720 píxeles a 30 fotogramas por segundo

En este equipo se hicieron todas la pruebas del proyecto, esto incluye, además de lo anterior mencionado, el procesamiento de las imágenes, el entrenamiento del modelo, las pruebas para obtener la estructura del modelo y los respectivos gráficos y resultados obtenidos.

### 6.0.2. Lenguaje de programación

El lenguaje de programación a ocupar fue python, esto se hizo de manera local por medio de jupyter Notebook. Lo que hace Jupyter Notebook es ejecutar desde la aplicación web cliente del navegador que se tenga por defecto.

## 6.1. Experimentación con subconjunto de datos

Se escogió un subconjunto de datos, esto con propósito de encontrar un modelo adecuado con una cantidad de datos limitados, para posterior utilizar el conjunto total de los datos y de esta

manera acotar tiempos de entrenamiento debido a los grandes volúmenes de datos.

La cantidad de imágenes utilizadas en esta experimentación son un total de 900 imágenes por gestos. Este proyecto pretende resolver la identificación tanto de gestos estáticos como dinámicos, por lo que para la muestra se tomó un 50 % por cada grupo:

1. Estáticos

- A,B,C,D

2. Dinámicos

- G,J,Ñ,X

El subconjunto de datos sera el siguiente:

Subconjunto de datos			
Acción/Gesto	Nº de Secuencias	Fotogramas por frecuencia	Puntos Clave
A	30	30	75
B	30	30	75
C	30	30	75
D	30	30	75
G	30	30	75
J	30	30	75
Ñ	30	30	75
X	30	30	75

Tabla 6.1: Subconjunto de datos

En los resultados obtenidos en la experimentación de la muestra se pueden destacar algunos puntos:

- **Límite de Neuronas:** A partir de las 256 neuronas el tiempo de entrenamiento aumenta considerablemente, a más del doble del modelo anterior(128 neuronas), además de esto el error en la muestra aumenta, por lo tanto a partir de este antecedente los demás modelos no superan esta cantidad de neuronas.
- **Función de activación en capas densas:** A partir de los resultados se obtuvo que las capas densas aumenta el error al trabajar en funciones de activación como Tanh y sigmoide, por lo tanto en esta capa solo utilizo ReLU.

### 6.1.1. Tabla de resultados subconjunto

A continuación se muestra una tabla con los resultados obtenidos, los datos a tener en cuenta a la hora de escoger el modelo serán el porcentaje de **precisión**, la **pérdida** y el **tiempo transcurrido** en el entrenamiento. Para la precisión se ha usado la métrica de accuracy, que nos devuelve la cantidad de aciertos obtenidos dándole importancia a los **True positive** y los **True negatives**. En cuanto a la razón por la cual incluir el tiempo como método de decisión del modelo, es debido a que nuestro problema requiere de inmediatez de respuesta mientras se interactúa con el usuario.

Resultados en un modelo LSTM						
Modelo	Neuronas	F. Activación	Optimizer	Métrica Accuracy %	F. perdida C.Cross.E %	Tiempo(s)
1 LSTM	32,8	Tanh	Adam	0,8851	0,2775	137,67
1 LSTM	64,8	Tanh	Adam	0,89	0,499	192,79
1 LSTM	128,8	Tanh	Adam	0,9372	0,4306	634,01
1 LSTM	256,8	Tanh	Adam	0,9268	0,5851	1469,86

Tabla 6.2: Resultados subconjunto de datos 1 RNN

En esta primera tabla en la cual se hace uso de una RNN de una sola capa podemos ver que a medida que se aumenta las neuronas la precisión de los modelos va mejorando hasta llegar a las 256 neuronas donde se aumenta el tiempo de entrenamiento a más del doble.

Resultados en un modelo LSTM						
Modelo	Neuronas	F. Activación	Optimizer	Métrica Accuracy %	F. perdida C.Cross.E %	Tiempo(s)
2 LSTM	16,16	Tanh,Tanh	Adam	0,8599	0,5138	163,28
2 LSTM	32,32	Tanh,Tanh	Adam	0,9214	0,4281	284,01
2 LSTM	32,64	Tanh,Tanh	Adam	0,9373	0,3221	230,58
2 LSTM	64,32	Tanh,Tanh	Adam	0,9063	0,4336	272,6
2 LSTM	64,64	Tanh,Tanh	Adam	0,9473	0,3754	301,46
2 LSTM	64,128	Tanh,Tanh	Adam	0,9218	0,5231	495,07
2 LSTM	128,64	Tanh,Tanh	Adam	0,9375	0,3421	590,9
2 LSTM	128,128	Tanh,Tanh	Adam	0,9319	0,4495	822,27

Tabla 6.3: Resultados subconjunto de datos 2 RNN

En esta segunda tabla muestra los resultados obtenidos al utilizar dos capas RNN, debido al aumento de capas se aumenta el número de pruebas a realizar. Se destacan las pruebas que utilizan grandes cantidades de neuronas, donde nuevamente el tiempo juega un factor decisivo a la hora de escoger el modelo.

Resultados en un modelo LSTM						
Modelo	Neuronas	F. Activación	Optimizer	Métrica Accuracy %	F. perdida C.Cross.E %	Tiempo(s)
3 LSTM	32,64,32	Tanh,Tanh,Tanh	Adam	0,8909	0,4507	309,71
3 LSTM	32,64,64	Tanh,Tanh,Tanh	Adam	0,896	0,6014	343,03
3 LSTM	32,64,128	Tanh,Tanh,Tanh	Adam	0,8905	0,6675	525,3
3 LSTM	64,64,32	Tanh,Tanh,Tanh	Adam	0,9059	0,4309	388,22
3 LSTM	64,64,64	Tanh,Tanh,Tanh	Adam	0,9322	0,429	407,4
3 LSTM	64,64,128	Tanh,Tanh,Tanh	Adam	0,9269	0,5093	636,62

Tabla 6.4: Resultados subconjunto de datos 3 RNN

En la tercera tabla se hace uso de 3 capas RNN, los primeros tres resultados están con una precisión menor al 90 %, y de los demás resultados el más óptimo es el que hace uso sólo de un conjunto de 64 neuronas. De igual manera la utilización de 128 neuronas empeora el tiempo.

Resultados en un modelo LSTM						
Modelo	Neuronas	F. Activación	Optimizer	Métrica Accuracy %	F. perdida C.Cross.E %	Tiempo(s)
2 LSTM 1 Densa	32,64,64	Tanh,Tanh,Tanh	Adam	0,8959	0,5736	226,22
2 LSTM 1 Densa	32,64,64	Tanh,Tanh,relu	Adam	0,9063	0,5088	238,38
2 LSTM 1 Densa	32,64,128	Tanh,Tanh,relu	Adam	0,9324	0,399	234,18
2 LSTM 1 Densa	64,64,32	Tanh,Tanh,relu	Adam	0,9216	0,7577	228,23
2 LSTM 1 Densa	64,64,64	Tanh,Tanh,relu	Adam	0,9375	0,4624	308,03
2 LSTM 1 Densa	64,64,128	Tanh,Tanh,relu	Adam	0,9114	0,5553	339,50
2 LSTM 1 Densa	64,64,128	Tanh,Tanh,relu	Adam	0,9113	0,5586	300,91

Tabla 6.5: Resultados subconjunto de datos 2 RNN y 1 Densa

Esta última tabla hace uso de dos capas RNN y una capa densa totalmente conectada, la implementación de esta capa extra tiene el objetivo de ver los resultados que se obtienen al conectar cada una de las neuronas de la capa anterior a cada una de las neuronas siguientes.

A partir del subconjunto de datos se encogieron los modelos con mejores resultados, los cuales son los siguientes:

- En la primera configuración se encuentra una red neuronal RNN con dos capas ocultas, con 32 y 64 neuronas respectivamente.
- En la segunda configuración se encuentra una red neuronal RNN con dos capas ocultas, con 64 y 64 neuronas respectivamente.
- En la tercera configuración se encuentra una red neuronal RNN con tres capas ocultas, con 64,64 y 64 neuronas respectivamente.
- En la cuarta configuración se encuentra una red neuronal RNN con dos capas ocultas y una capa densa, con 32, 64 y 64 neuronas respectivamente.

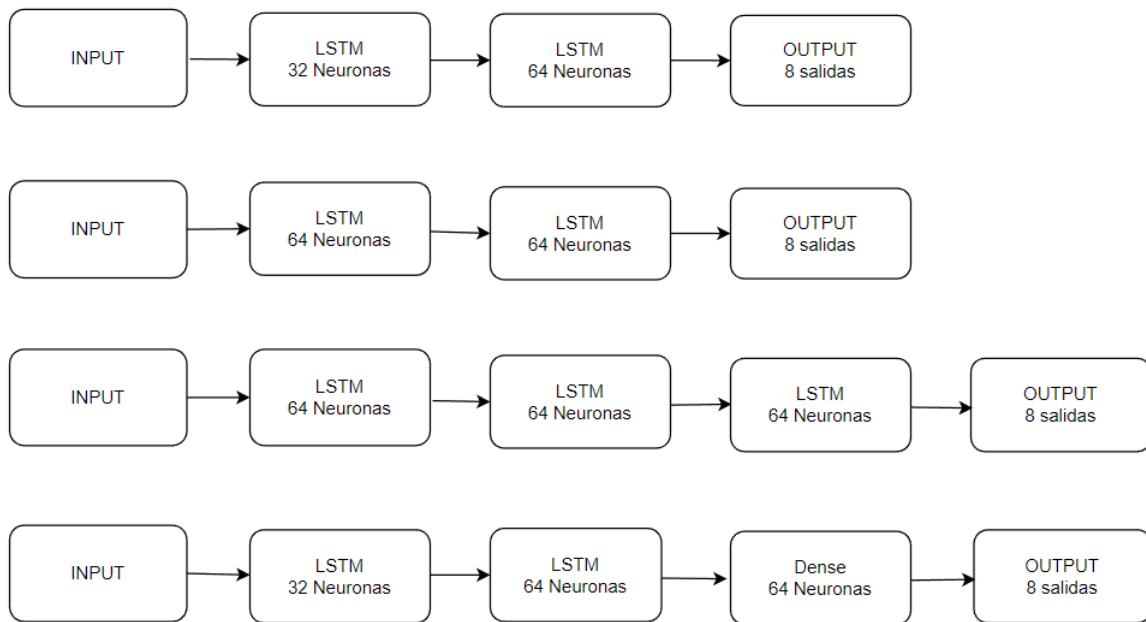


Figura 6.1: 4 Modelos escogidos a partir de los resultados obtenidos

### 6.1.2. Optimizador

Se hicieron pruebas con tres optimizadores: Adam, SGD y RMSprop. Estas pruebas se hicieron en las cuatro estructuras escogidas anteriormente (capas y neuronas) y utilizando los modelos de LSTM, GRU, SimpleRNN y Bi-LSTM.

De las pruebas realizadas se obtuvo el rendimiento y la pérdida de los optimizadores, a continuación se muestran los gráficos de caja creados a partir de error obtenido, este error es el resultado de utilizar Cross-Validation.

Estos gráficos nos muestran los cuartiles en los que está distribuido los datos, mientras más alto sea el gráfico más disperso están los datos.

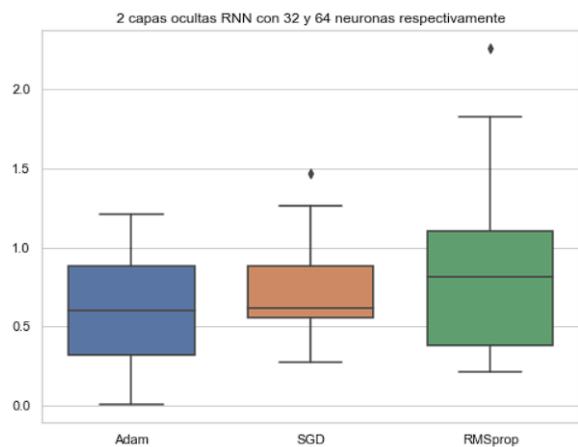


Figura 6.2: Gráfico de caja de la primera estructura

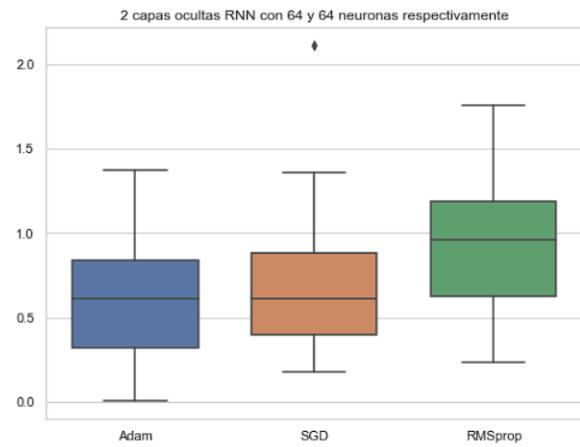


Figura 6.3: Gráfico de caja de la segunda estructura

De el primer gráfico podemos observar que en el optimizador Adam los valores inter-cuartiles (distancia de la mediana al cuartil 1 y 3) son similares, por parte de el optimizador SGD los valores están más acotados y por ende menos dispersos, pero su forma nos muestra que tiene peores resultados que Adam, además de esto posee un valor atípico. Finalmente RMSprop tiene un rango más amplio de valores (menos estable) y posee valores atípicos.

Del segundo gráfico podemos observar un comportamiento similar al caso anterior, sin embargo notamos que los valores de SGD son más dispersos y posee valores atípicos mayores.

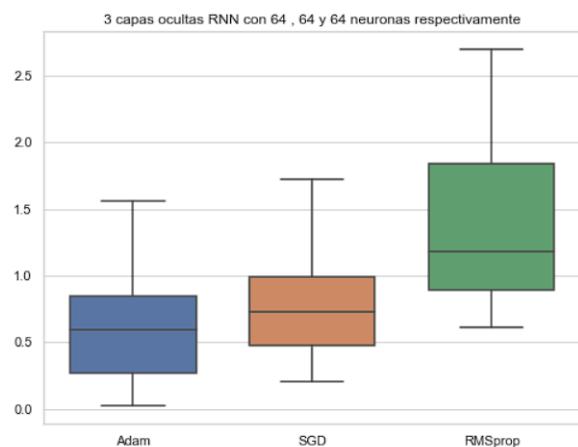


Figura 6.4: Gráfico de caja de la tercera estructura

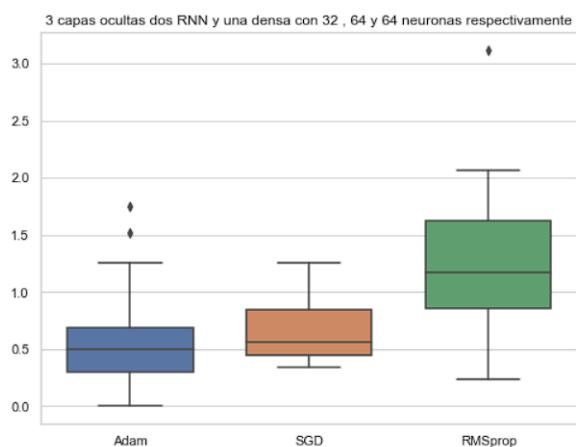


Figura 6.5: Gráfico de caja de la cuarta estructura

En esta tercera estructura notamos que los valores de RMSprop empeoran tanto en el error como en la dispersión de los datos, Adam sigue teniendo los mejores resultados seguidos por SGD.

En la última estructura notamos mejores resultados en todos los optimizadores pero hay mayor cantidad de valores atípicos.

Por los gráficos mostrados, se concluye que en este proyecto el optimizador a utilizar será Adam, pese a tener valores más dispersos que SGD esta posee valores de error inferiores.

## 6.2. Modelos RNNs

- SimpleRNN
- LSTM
- GRU
- Bidirectional LSTM

Resultados								
Modelo	Neuronas	F. Activación	Optimizer	Métrica	F. perdida	% Métrica	% loss	Tiempo(s)
2 LSTM	32,64	Tanh,Tanh	Adam	accuracy	C.Cross.E	0,8847	0,7338	333.49
2 GRU	32,64	Tanh,Tanh	Adam	accuracy	C.Cross.E	0,911	0,5941	255,12
2 BLSTM	32,64	Tanh,Tanh	Adam	accuracy	C.Cross.E	0,9211	0,4579	486,99
2 SimpleRNN	32,64	Tanh,Tanh	Adam	accuracy	C.Cross.E	0,9317	0,4671	148,15
Modelo	Neuronas	F. Activación	Optimizer	Métrica	F. perdida	% Métrica	% loss	Tiempo(s)
2 LSTM	64,64	Tanh,Tanh	Adam	accuracy	C.Cross.E	0,9006	0,5371	394,12
2 GRU	64,64	Tanh,Tanh	Adam	accuracy	C.Cross.E	0,9214	0,507	321,39
2 BLSTM	64,64	Tanh,Tanh	Adam	accuracy	C.Cross.E	0,9218	0,4614	730,03
2 SimpleRNN	64,64	Tanh,Tanh	Adam	accuracy	C.Cross.E	0,9371	0,4122	114,83
Modelo	Neuronas	F. Activación	Optimizer	Métrica	F. perdida	% Métrica	% loss	Tiempo(s)
3 LSTM	64,64,64	Tanh,Tanh,Tanh	Adam	accuracy	C.Cross.E	0,9112	0,5671	516,21
3 GRU	64,64,64	Tanh,Tanh,Tanh	Adam	accuracy	C.Cross.E	0,9113	0,5248	383,99
3 BLSTM	64,64,64	Tanh,Tanh,Tanh	Adam	accuracy	C.Cross.E	0,9109	0,637	1117,331
3 SimpleRNN	64,64,64	Tanh,Tanh,Tanh	Adam	accuracy	C.Cross.E	0,9319	0,4814	167,37
Modelo	Neuronas	F. Activación	Optimizer	Métrica	F. perdida	% Métrica	% loss	Tiempo(s)
2 LSTM 1 Dense	32,64,64	Tanh,Tanh,Relu	Adam	accuracy	C.Cross.E	0,9004	0,7587	282,51
2 GRU 1 Dense	32,64,64	Tanh,Tanh,Relu	Adam	accuracy	C.Cross.E	0,9058	0,6745	239,09
2 BLSTM 1 Dense	32,64,64	Tanh,Tanh,Relu	Adam	accuracy	C.Cross.E	0,8901	0,74498	523,75
2 SimpleRNN 1 Dense	32,64,64	Tanh,Tanh,Relu	Adam	accuracy	C.Cross.E	0,9109	0,7562	100,97

Tabla 6.6: Resultados de la prueba con distintos modelos

### 6.2.1. Elección de modelo

A partir de los datos podemos concluir varios puntos:

- El modelo con mejores resultados es el SimpleRNN pero como este modelo está basado en una RNN estándar, esta no será capaz de recordar valores importantes a largo plazo, por lo que se optara por otro modelo.
- El siguiente mejor modelo es el modelo de tres RNNs con Bi-LSTM, pero hay que tener en cuenta que los resultados importantes no son solo la precisión y la pérdida sino también el tiempo de entrenamiento de la misma, la cual duplica a la mayoría de los demás entrenamientos, por esta razón no se escogerá este modelo.
- El tercer mejor resultado es el modelo que utiliza dos RNNs GRU con 64 neuronas cada capa, con una precisión del 92,14 % y perdida de 0,507, además del tiempo de entrenamiento es la mitad del modelo anterior, por lo cual este será el modelo escogido para el entrenamiento final con todos los datos.

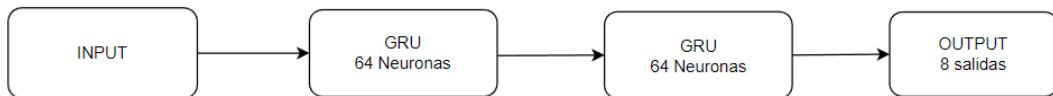


Figura 6.6: Modelo GRU de 64 neuronas cada capa

### 6.2.2. Resultados

Una vez hecho las pruebas y encontrado el modelo más eficiente se procederá a aplicar ese modelo en el dataset completo mencionado en el capítulo 4. Todos los datos del experimento pasaron por una validación de los resultados llamada validación cruzada con k=5, correspondiente al 80 % de los datos son para el entrenamiento y al 20 % para la validación.

En las tablas 6.7,6.8 y 6.9 se muestran los resultados obtenidos del experimento con la base de datos de 21 gestos estáticos, 14 dinámicos con una mano y 14 dinámicos con dos manos.

Dataset Estático			
	Entrenamiento	Validación	Testeo
Precisión	97.95 %	95.25 %	96.21%
Perdida	0.0676	0.3399	0.39933
Tiempo	445.715	no aplica	no aplica

Tabla 6.7: tabla Dataset Estático

Dataset Dinámico una mano			
	Entrenamiento	Validación	Testeo
Precisión	99.41%	96.11%	95.56 %
Perdida	0.0232	0,2447	0.2130
Tiempo	303.05	no aplica	no aplica

Tabla 6.8: tabla Dataset Dinámico

Dataset Dinámico dos mano			
	Entrenamiento	Validación	Testeo
Precisión	98.5 %	99.16 %	98.88 %
Perdida	0.0690	0.08256	0.0870
Tiempo	286.87	no aplica	no aplica

Tabla 6.9: tabla Dataset Dinámico dos manos

A partir de las tablas se puede concluir que los gestos realizados por ambas manos tiene un mayor precisión que solo utilizar una mano, además de un menor error en el modelo. La razón del aumento en la precisión es debido a que al haber un mayor número de puntos de referencia en pantalla el modelo es capaz de obtener una muestra más compleja y precisa de la imagen/gesto.

Resultados Finales			
	Entrenamiento	Validación	Testeo
Precisión	98.5 %	94.25 %	95.67 %
Perdida	0.0692	0,2408	0.195

Tabla 6.10: Resultados finales

En la tabla 6.10 se muestran los resultados finales de la colección de datos, con un total de 50 gestos. Con testeo del 20 % se logró una precisión del 95.67 % y una pérdida de 0.195.

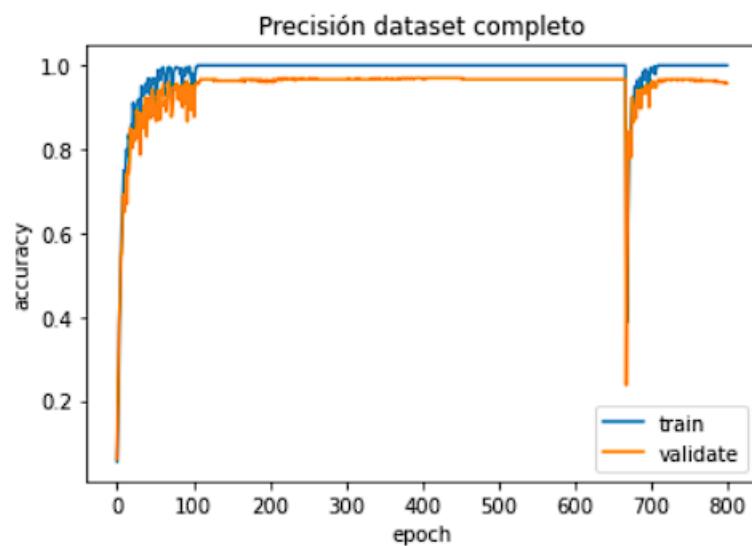


Figura 6.7: Precisión Entrenamiento vs Validación

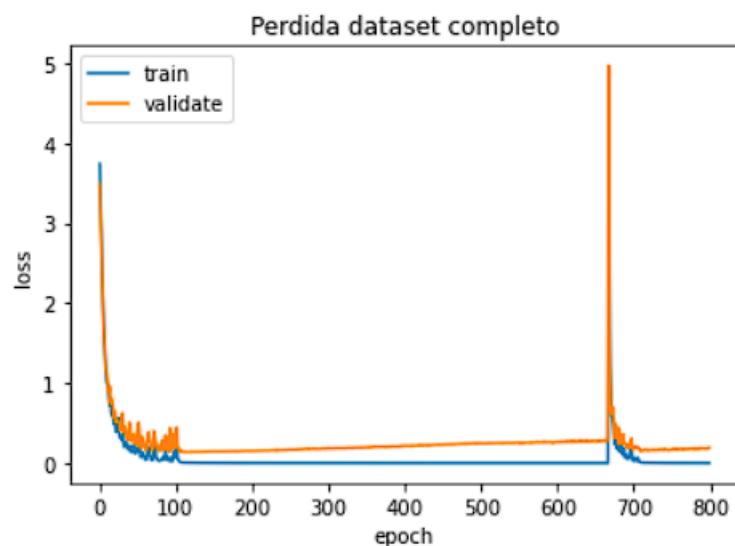


Figura 6.8: Perdida Entrenamiento vs Validación

Existen otros trabajos realizados en otros países que usaron otros modelos para resolver esta problemática, como Reconocimiento de gestos dinámicos y su aplicación al lenguaje de señas"por Franco Ronchetti donde se utilizaron clasificadores ProbSOM y métodos como Máquinas de Soporte Vectorial(SVM).(Ronchetti y Lanzarini, 2019)

En el proyecto realizado en Argentina se utilizó una vestimenta específica, ropa negra, y con un fondo blanco para resaltar la silueta, además de una iluminación controlada. Para mejorar el reconocimiento de las manos se utilizó guantes con colores fluorescentes, con formato de vídeo FullHD con 60fps y se realizaron 64 gestos interpretados por 10 personas.

Diferentes modelos				
	Modelo Propio GRU	Propio ProbSOM	SVM	
Precisión	95.35 %	92.3 %	91.2 %	

Tabla 6.11: tabla con resultados NO COMPARABLES

A pesar de que se hace una comparación de ambos modelos hay que tener varios factores que las diferencian, como el entorno de captura de datos, en este proyecto no se utilizó ningún tipo de guante para mejorar la captura, además se utilizó luz natural y no controlada, la cantidad de Frames utilizada en este proyecto fueron de 30 y una cámara de 720p, pero principalmente no se utiliza la misma colección de datos por lo que no es posible realizar una comparación estadísticamente correcta y vinculante.

# Capítulo 7

## Arquitectura de Solución de reconocimiento de señas

### 7.1. Esquema general de la solución

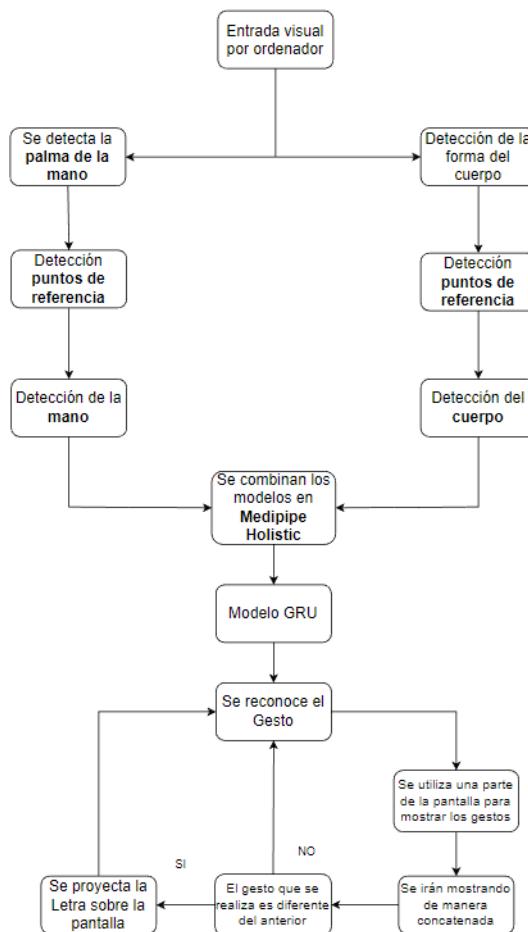


Figura 7.1: Esquema de detección de gestos

### 7.1.1. Descripción estructura

Esta estructura comienza por la entrada visual de los gestos que se hará con un dispositivo lo suficientemente potente para que el modelo funcione de manera fluida. Posterior a esto el reconocimiento se divide en dos partes, una centrada en las manos y otra en el cuerpo. Por la parte de la mano esta comienza detectando la palma de la mano para posteriormente detectar sus puntos de referencia. En cuanto al cuerpo se obtienen dos puntos para de esta manera detectar el centro de la persona, su rotación y su escala, además de detectar puntos de referencia. Una vez hecho esto ambos modelos, de mano y cuerpo, se combinan en un solo modelo.

Ya obtenido las posiciones de una persona se procede a utilizar el modelo RNN de reconocimiento, en este caso es una GRU, esto se mostrará por consola pero para obtener un resultado más visual para el usuario se deberá crear o utilizar una parte de la pantalla e ir mostrando los gestos realizados, para evitar que se muestren repetidamente los mismos gestos cuando no se cambia de postura se debe agregar una condición que evite gestos repetidos, cada uno de estos gestos se irán mostrando de manera concatenada sobre la pantalla.

## 7.2. Implementación de captura de imágenes

La implementación de los gestos se realizó en un entorno con luz natural y fondo blanco, de la misma manera en la que se obtuvieron los datos dinámicos, para tener un entorno controlado. Se hace uso de una cámara web, la cual es mostrada por pantalla, a medida que se vayan realizando los gestos estos se irán mostrando por pantalla de manera concatenada, además se agrego una condición que evita que muestre por pantalla el gesto anteriormente realizado para de esta manera evitar la repetición(por ejemplo un gesto estático detectado varias veces)

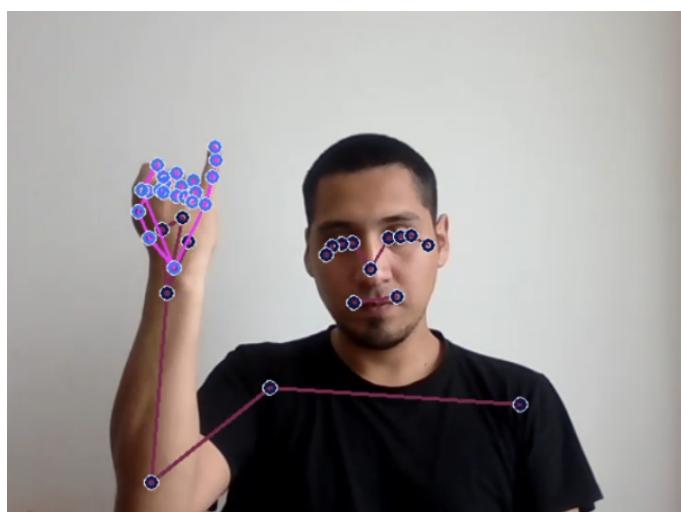


Figura 7.2: Letra dinámica J

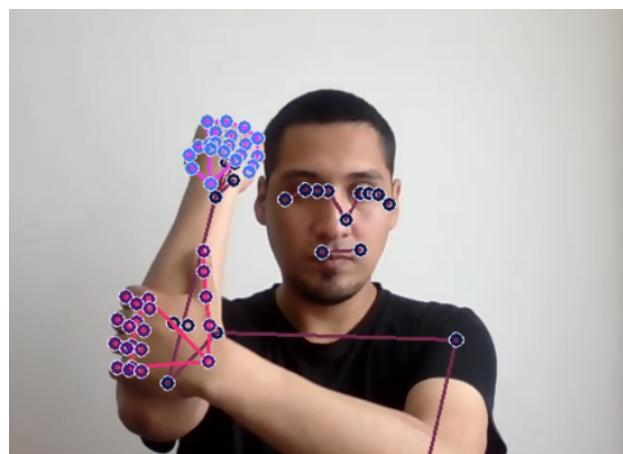


Figura 7.3: Palabra dinámica Ayudar

### 7.2.1. Alcances del Desarrollo

A la hora de ejecutar el programa este muestra una inusual lentitud a la hora de mostrar los fotogramas de la cámara, provocando una detección lenta de los gestos y por lo tanto errónea. Esto puede ser provocado por factores como la cámara de baja velocidad o por insuficiencia de procesamiento en el equipo ocupado en el proyecto.

Esto se podría resolver agregando una entrada(cámara) de mayor capacidad, como por ejemplo la cámara OAK-D, capacitada para trabajar con algoritmos de inteligencia artificial. Otra manera de resolver este problema sería mejorando el rendimiento del programa, para esto existen librerías como Cython que permite combinar código de Python con código de C.

Esta implementación con estas características debiese llevarse a cabo en un proyecto enfocado en el desarrollo de software, ya que requiere de experiencia en tecnologías móviles y web, considerando la forma más simple de uso para un usuario del grupo de personas con discapacidad auditiva.

# Capítulo 8

## Trabajo Futuro

Este proyecto posee un gran margen de mejora para trabajos posteriores.

**Agrandar la muestra de señas** para mejorar la robustez del reconocimiento permitiendo hacer uso de un mayor número de palabras, además de variar la cantidad de personas implicadas en la recolección de datos, este proyecto tiene contemplado los resultados de una persona, por lo que variar los tamaños de manos y cuerpos es un punto a considerar. Esto se podría lograr a través de la recolección de datos creando un **datasets abierto** a todo el mundo que quiera realizar un aporte al reconocimiento de los gestos y ser utilizado por cualquiera que la requiera.

**Concatenar palabras** es el paso siguiente para conseguir un algoritmo más completo y de esta manera poder ser capaz de traducir frases.

Finalmente crear un entorno para la **implementación del software**, permitiendo mostrar por pantalla los gestos que se van realizando

# Capítulo 9

## Conclusión

El reconocimiento de todos los gestos de la LSCH es un trabajo arduo, por lo que este proyecto pretende formar parte de una manera mejorar la calidad de vida de las personas con dificultades auditivas. En este proyecto de título logra crear un modelo capaz de reconocer un conjunto de los gestos de la lengua de señas chilena, en concreto los gestos dinámicos, utilizando un modelo basado en GRU con dos capas de 64 neuronas.

Para poder lograr este objetivo general se cumplieron los objetivos específicos.

Realizando una búsqueda de información de varios países, en especial hispanohablantes, para ver las similitudes entre los gestos. A partir de esto, se encontraron varios conjuntos de datos de gestos estáticos, pero insuficientes para las pruebas. Por lo que se procedió a obtener los datos de manera manual, creando un conjunto de datos propio, cumpliendo con lo planteado en el **Objetivo Específico 1**.

Además, se presenta una investigación sobre los modelos capaces de reconocer patrones temporales y dimensionales, que en el caso de este proyecto se usó Redes neuronales recurrentes. Se probaron cuatro modelos: SimpleRNN, LSTM, GRU y Bi-LSTM, donde se obtuvo el modelo óptimo a través de experimentación y ajustes de los hiper-parámetros. El **Objetivo Específico 2** se cumple al escoger al modelo basado en GRU con dos capas de 64 neuronas.

Y finalmente, para el **objetivo específico 3**, una vez realizado el modelo se procedió a usar el conjunto de datos completo que contó con 50 gestos incluyendo gestos estáticos como dinámicos, como resultado se obtuvo una precisión del 95.67% utilizando el conjunto de datos de testeo.

Por último, se resalta el aporte realizado al lograr el reconocimiento de un conjunto de señas dinámicas que, en la investigación realizada, han sido complejas de reconocer. Por lo que, el desarrollo de esta línea de trabajo abre las puertas a nuevas aplicaciones inclusivas enfocadas a personas con dificultad auditiva, siendo este proyecto de título un real aporte en el reconocimiento automático del lenguaje de señas chilena.

# Referencias

- (s.f.).
- ADAMO QUINTELA, D., ACUÑA ROBERTSON, X., y CABRERA RAMÍREZ, I. (2013). Diccionario bilingüe lengua de señas chilena/español: Un desafío lexicográfico. *RLA. Revista de lingüística teórica y aplicada*, 51(2), 173–192.
- Berrar, D. (2019). *Cross-validation*.
- Bonacorso, G. (2017). *Machine learning algorithms*. Packt Publishing Ltd.
- Brownlee, J. (2017). Difference between classification and regression in machine learning. *Machine Learning Mastery*, 25, 985–1.
- Choi, D., Shallue, C. J., Nado, Z., Lee, J., Maddison, C. J., y Dahl, G. E. (2019). On empirical comparisons of optimizers for deep learning. *arXiv preprint arXiv:1910.05446*.
- Dangeti, P. (2017). *Statistics for machine learning*. Packt Publishing Ltd.
- Espinosa Hoyos, C. A. (2020). Simulación y evaluación de técnicas de clustering para el reconocimiento de gestos estáticos en la traducción de la lengua de señas peruana.
- Fiel, C. P., Castro, C. C., Arvizu, V. V., Pitalúa-Díaz, N., Jiménez, D. S., y Carvajal, R. R. (2013). Design of translator glove for deaf-mute alphabet. En *3rd international conference on electric and electronics* (pp. 485–488).
- Fok, K.-Y., Ganganath, N., Cheng, C.-T., y Chi, K. T. (2015). A real-time asl recognition system using leap motion sensors. En *2015 international conference on cyber-enabled distributed computing and knowledge discovery* (pp. 411–414).
- Gers, F. A., Schmidhuber, J., y Cummins, F. (2000). Learning to forget: Continual prediction with lstm. *Neural computation*, 12(10), 2451–2471.
- Goodfellow, I., Bengio, Y., y Courville, A. (2016). *Deep learning*. MIT press.
- Graves, A., Mohamed, A.-r., y Hinton, G. (2013). Speech recognition with deep recurrent neural networks. En *2013 ieee international conference on acoustics, speech and signal processing* (pp. 6645–6649).
- Herrera Fernández, V. (2017). Estudio de la población sorda en chile: evolución histórica y perspectivas lingüísticas, educativas y sociales.
- Kelleher, J. D. (2019). *Deep learning*. MIT press.
- Li, G., Wu, H., Jiang, G., Xu, S., y Liu, H. (2018). Dynamic gesture recognition in the internet of things. *IEEE Access*, 7, 23713–23724.
- López-Noriega, J. E., Fernández-Valladares, M. I., y Uc-Cetina, V. (2014). Glove-based sign language recognition solution to assist communication for deaf users. En *2014 11th international conference on electrical engineering, computing science and automatic control (cce)* (pp. 1–6).
- Matich, D. J. (2001). Redes neuronales: Conceptos básicos y aplicaciones. *Universidad Tecnológica Nacional, México*, 41, 12–16.

- Matloff, N. (2017). *Statistical regression and classification: from linear models to machine learning.* Chapman and Hall/CRC.
- Mery, D. (2004). Visión por computador. *Santiago de Chile. Universidad Católica de Chile.*
- Mikołajczyk, A., y Grochowski, M. (2018). Data augmentation for improving deep learning in image classification problem. En *2018 international interdisciplinary phd workshop (iiphdw)* (pp. 117–122).
- Perez, L., y Wang, J. (2017). The effectiveness of data augmentation in image classification using deep learning. *arXiv preprint arXiv:1712.04621.*
- Ramos, Y., y cols. (2015). Senadis-resultados ii estudio nacional de la discapacidad.
- Riveros, C. G., y Inostroza, F. Y. (2016). *Sistema de reconocimiento gestual de lengua de señas chilena mediante cámara digital* (Tesis Doctoral no publicada). Tesis de Grado, Pontificia Universidad Católica de Valparaíso, Chile.
- Ronchetti, F. (2018). Reconocimiento de gestos dinámicos y su aplicación al lenguaje de señas. En *Xx workshop de investigadores en ciencias de la computacion (wicc 2018, universidad nacional del nordeste).*
- Ronchetti, F., y Lanzarini, L. C. (2019). Reconocimiento de gestos dinámicos y su aplicación a la lengua de señas. *Investigación Joven, 6(Especial),* 174–175.
- Sánchez, A., Esteban, J. L., Vélez, J. F., y Moreno, A. B. (2003). *Visión por computador.* Dykinson.
- Sharma, S., Sharma, S., y Athaiya, A. (2017). Activation functions in neural networks. *towards data science, 6(12),* 310–316.
- Yu, Y., Si, X., Hu, C., y Zhang, J. (2019). A review of recurrent neural networks: Lstm cells and network architectures. *Neural computation, 31(7),* 1235–1270.