



FACULTAD DE ESTUDIOS ESTADÍSTICOS

GRADO EN ESTADÍSTICA APLICADA

Curso 2021/2022

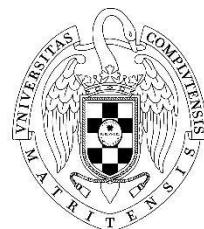
Trabajo de Fin de Grado

TITULO: ¿Puede una máquina leer los labios?

Alumno: Roberto Carlos Saavedra Baylón

Tutor: José Adolfo Gálvez Moraleda

Junio de 2022



UNIVERSIDAD COMPLUTENSE
MADRID

Agradecimientos

A mis padres, Martha y Wilmer, por haberme apoyado incondicionalmente a lo largo de mi vida y por enseñarme la importancia del esfuerzo. También me gustaría agradecer a los grandes profesores que he tenido en la universidad. Por último, me gustaría agradecer a Import Coffee, asociación de estudiantes de la Facultad de Estudios Estadísticos, por los buenos momentos que he pasado en ella y por la gente que he conocido.

Resumen:

En la década de 1980 se empezó a plantear la pregunta si era posible realizar reconocimiento del lenguaje a partir de señales visuales, es decir cómo conseguir que una máquina sea capaz de imitar la habilidad humana de leer los labios, para conseguir esta tarea se han utilizado las técnicas desarrolladas en los campos del Reconocimiento del Lenguaje y de la Visión Artificial, además para poder entrenar estos algoritmos se han ido creando diferentes conjuntos de datos, desde más simples, con dígitos y letras, hasta más complejos, con frases y oraciones, a partir de escenarios reales como pueden ser programas de televisión. De igual manera, las técnicas utilizadas han ido evolucionando, desde las primeras aproximaciones donde se dividía la tarea en bloques, localización de los labios, extracción de características y clasificación, hasta la actualidad, donde se deja que todas las tareas sean resueltas por una red neuronal end-to-end. En este trabajo se ha escogido exponer los resultados obtenidos después de experimentar con algunas de las técnicas utilizadas desde la década de 1990 hasta la actualidad.

Palabras Clave: Lectura de Labios, Reconocimiento del Lenguaje, Visión Artificial, Aprendizaje Estadístico, Aprendizaje Automático

Abstract:

In the 1980s the question if it is possible to perform speech recognition based in visual features emerged, namely how to get a machine to imitate the human skill of reading the lips during a conversation, in order to achieve this, different techniques from Speech Recognition and Computer Vision has been used. Furthermore, in order to train these algorithms different datasets has been created, from simple ones, with digits and letters, to more complex with phrases and sentences, from real world scenarios extracted from television programs, in the same way the techniques used have evolved, the first approaches divided the task in different blocks, lips localization, feature extraction and classification, although, nowadays all the tasks are performed by a single end-to-end neural network. In this work, the obtained results are exposed with some of the techniques that have been used from the 1990s to the present.

Keywords: Lip Reading, Speech Recognition, Computer Vision, Statistical Learning, Machine Learning

Índice

Índice de figuras	9
Índice de cuadros	11
1. Introducción	13
2. Estado del Arte	15
2.1. Conjuntos de Datos	15
2.2. Sistemas Automáticos de Lectura de Labios Tradicionales	17
2.3. Sistemas Automáticos de Lectura de Labios basados en Redes Neuronales	18
2.4. Aplicaciones	20
3. Descripción del conjunto de datos	23
3.1. <i>MIRACL-VC₁</i>	23
4. Sistemas Tradicionales	25
4.1. Detección de los labios	25
4.2. Extracción de las características visuales	27
4.2.1. Eigenlips - Análisis de Componentes Principales	27
4.3. Clasificación	29
4.3.1. Máquina de Soporte Vectorial	29
4.3.2. Modelo Oculto de Markov	31
5. Deep Learning	35
5.1. Redes Neuronales	35
5.2. Redes Neuronales Convolucionales	36
5.3. Redes Neuronales Recurrentes	38
5.4. Modelo basado en Redes Neuronales	39
6. Conclusiones	43
6.1. Trabajo Futuro	44
Referencias	47

Índice de figuras

1.	Persona pronunciando la palabra en inglés “hello”	13
2.	Imagen del conjunto de datos MNIST	14
3.	Bloques que constituyen un Sistema Automático de Lectura de Labios	15
4.	Diferentes Conjuntos de datos (Fernandez-Lopez y Sukno, 2018)	16
5.	Reconocimiento de la cara y puntos de interés faciales (Gurjar, 2016)	17
6.	El número de veces que un extractor de características (izquierda) y un clasificador (derecha) ha sido usado del 2007 al 2017 (Fernandez-Lopez y Sukno, 2018)	18
7.	El número de veces que un extractor de características (izquierda) y un clasificador (derecha) ha sido usado del 2007 al 2018 (Fernandez-Lopez y Sukno, 2018)	19
8.	Fotograma del conjunto de datos Lip2Wav	20
9.	SRAVI: Speech Recognition App for The Voice Impaired (Liopa, 2022)	20
10.	Fotograma inicial a la izquierda y a la derecha región de la cara detectada	25
11.	Puntos de interés o landmarks faciales	26
12.	Área de los labios	26
13.	Recorte de la región de la boca	27
14.	Gráfico de sedimentación	28
15.	Autovectores - eigenlips	29
16.	Clasificación con Máquina de Soporte Vectorial	30
17.	Diagrama de un Modelo Oculto de Markov con 5 observaciones y 5 estados ocultos	32
18.	Diagrama de una red neuronal con 4 entradas, 1 capa oculta y 1 valor como salida (James, Witten, Hastie, y Tibshirani, 2013)	35
19.	Red Neuronal Convolutacional (James y cols., 2013)	36
20.	Demostración de dos filtros aplicados a una misma imagen (Geron, 2019)	37
21.	Demostración de una capa de pooling aplicada a una imagen (Geron, 2019)	38
22.	Red Neuronal Recurrente (James y cols., 2013)	38
23.	Convolución en 3 dimensiones	39
24.	Arquitectura usada	40
25.	Demostración del conjunto de datos incluyendo profundidad (Rekik, Ben-Hamadou, y Mahdi, 2014)	45

Índice de cuadros

1.	Variable dependiente	24
2.	WRR de la Máquina de Soporte Vectorial	30
3.	Matriz de Confusión para palabras, Máquina de Soporte Vectorial, conjunto de test	31
4.	Matriz de Confusión pra frases, Máquina de Soporte Vectorial, conjunto de test	31
5.	WRR de los Modelos Ocultos de Markov	33
6.	Matriz de Confusión palabras, Modelo Oculto de Markov	33
7.	Matriz de Confusión frases, Modelo Oculto de Markov	33
8.	WRR de una Red Neuronal	41
9.	Matriz de Confusión para palabras, Red Neuronal, conjunto test	41
10.	Matriz de Confusión para frases, Red Neuronal, conjunto test	41

1. Introducción

Una computadora puede ser llamada “inteligente” si logra engañar a una persona haciéndole creer que es un humano.

Alan Turing

¿Puede una máquina imitar la habilidad humana de leer los labios de una persona? Para que una máquina sea capaz de interpretar una señal visual en primer lugar tiene que ser capaz de imitar la visión humana, para ello una máquina puede usar cualquier tipo de cámara, una vez se tiene un vídeo ya se puede intentar leer los labios de la persona que aparece en dicho vídeo, este primer problema ha sido relativamente fácil de resolver, sin embargo, queda la parte más complicada en la que se tiene enseñar a una máquina a imitar la habilidad humana de leer los labios.

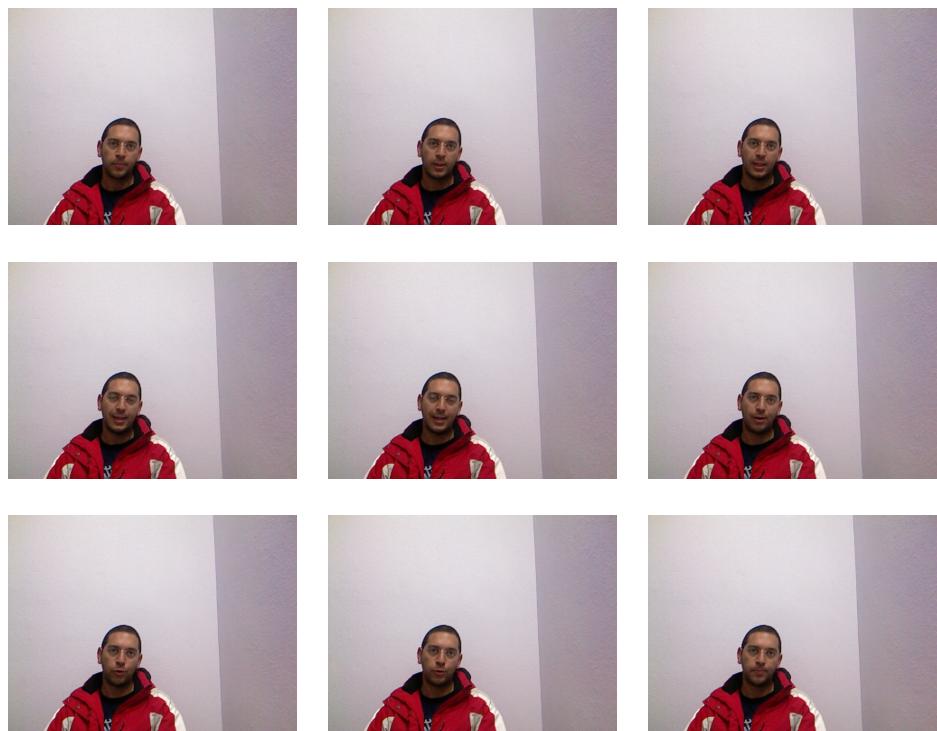


Figura 1: Persona pronunciando la palabra en inglés “hello”

En estas 9 imágenes se puede ver a una persona pronunciando una palabra, para poder conseguir que una máquina pueda distinguir qué palabra es la que está siendo pronunciada en una secuencia de imágenes se van a aprovechar los datos que existen en una imagen. Cada una de las imágenes de la Figura 1 tiene 640 píxeles de ancho y 480 píxeles de alto, además debido al formato RGB, cada pixel cuenta con tres componentes (RED, GREEN, BLUE) (Fulton, s.f.). Es evidente que en una imagen o en conjunto de ellas hay una gran cantidad de datos que pueden ser explotados. Para entender mejor la información que existe en una imagen se puede ver la Figura 2, la imagen de la izquierda es el dígito 0 manuscrito, la imagen de la derecha son los valores correspondientes a cada uno de los píxeles, 28 de alto por 28 de ancho (784 píxeles en total), es importante destacar que esta es una imagen en blanco y negro, para una imagen en color para cada celda habría 3

componentes correspondientes a los valores RGB, pero por simplificar la visualización se utiliza esta imagen.

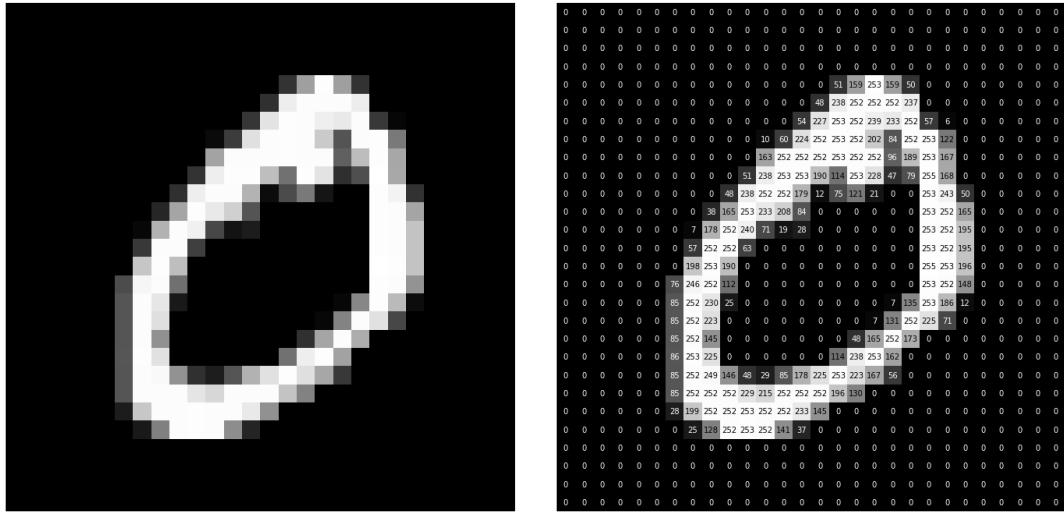


Figura 2: Imagen del conjunto de datos MNIST

Como se puede ver el objetivo es utilizar los datos que existen en una imagen para que un ordenador sea capaz de distinguir que palabra está siendo pronunciada por una persona, el usar datos para resolver un problema no es algo novedoso, a principios del siglo 19 ya se empezaron a resolver problemas utilizando datos en astronomía y geodesia, por ejemplo Gauss calculando la órbita de Ceres, utilizando el método de Mínimos Cuadrados (Stigler, 1986), y desde entonces hasta día de hoy se ha visto como se han usado datos para una gran variedad de tareas como enseñar a una computadora a jugar juegos de mesa, conducir coches, recomendar una película o canción o predecir una enfermedad. Para conseguir que una máquina puede desarrollar estas tareas, las últimas décadas han resultado clave debido al aumento del poder computacional y el desarrollo de nuevas técnicas que resultan más apropiadas en algunos contextos. Estas técnicas se han desarrollado en distintos campos como pueden ser la Estadística, Ciencia de Datos o Inteligencia Artificial, entre otros.

El objetivo de este trabajo es utilizar los datos que proporciona una imagen (921600 valores por imagen) para que un ordenador prediga qué es lo que está pronunciando una persona a partir de una secuencia de imágenes. Para ello se van a aprovechar los avances en el campo del Deep Learning, así como métodos tradicionales usados en el campo del Reconocimiento del Habla. El conjunto de datos que se ha utilizado ha sido MIRACL-VC1 (Rekik y cols., 2014). Este trabajo tiene una fuerte componente práctica, si se tiene interés en la parte experimental de este trabajo se puede consultar el repositorio de GitHub del proyecto (Saavedra, 2022).

Con respecto a la estructura, en la siguiente sección se procederá a revisar el **Estado del Arte**, posteriormente se describirán los datos que se han usado para la parte experimental. A continuación, se compartirán los resultados que se han obtenido utilizando **Sistemas Tradicionales y Deep Learning**. Para finalizar, se compararán los resultados obtenidos y se hablará del posible **Trabajo Futuro**.

2. Estado del Arte

En este capítulo se va a proceder a revisar el Estado del Arte de los sistemas automáticos de lectura de labios (por sus siglas en inglés, ALR, *Automatic Lip-Reading*). Este problema se ha intentado resolver ajustando modelos estadísticos a los datos que se encuentran en las imágenes, dada la importancia de los datos, se va a empezar la revisión comentando algunos conjuntos de datos relevantes que han sido utilizados en este campo desde la década de 1990 (Fernandez-Lopez y Sukno, 2018) hasta la actualidad, además se repasaran las diferentes aproximaciones a la hora de resolver el problema, desde enfoques tradicionales, donde se dividía el problema en reconocimiento facial y localización del área de los labios, extracción de características visuales y clasificación, hasta enfoques modernos donde se sigue usando una estructura similar a los pasos en la Figura 3, pero las tres tareas (localización, extracción y clasificación) recaen en una red neuronal que se encarga de todo el proceso.



Figura 3: Bloques que constituyen un Sistema Automático de Lectura de Labios

Para evaluar el rendimiento de los modelos se utiliza la métrica WRR (*Word Recognition Rate* o Ratio de Reconocimiento de Palabras)

$$WRR = \frac{\# \text{PALABRAS RECONOCIDAS CORRECTAMENTE}}{\# \text{PALABRAS TOTALES}}$$

2.1. Conjuntos de Datos

En la década de 1990, con los primeros conjuntos de datos, los esfuerzos se centraron en resolver tareas sencillas, como reconocer letras del alfabeto o dígitos (Matthews, Cootes, Bangham, Cox, y Harvey, 2002), el problema con estos datos era que debido a

su naturaleza era complicado generalizar los resultados a situaciones realistas, por este motivo los conjuntos de datos construidos posteriormente eran más grandes y más complejos, utilizando palabras y frases. En los últimos años, han surgido conjuntos de datos que trabajan el habla continua, algunos de estos datos han sido construidos a partir de programas de televisión, en español se puede destacar el conjunto de datos de programas de RTVE (Lleida y cols., 2020) y en inglés de la BBC (Chung y Zisserman, 2017). En la Figura 4 se observan algunos conjuntos de datos populares. Además, es importante señalar que los conjuntos de datos suelen estar en inglés, datos en otros idiomas suelen ser menos frecuentes.

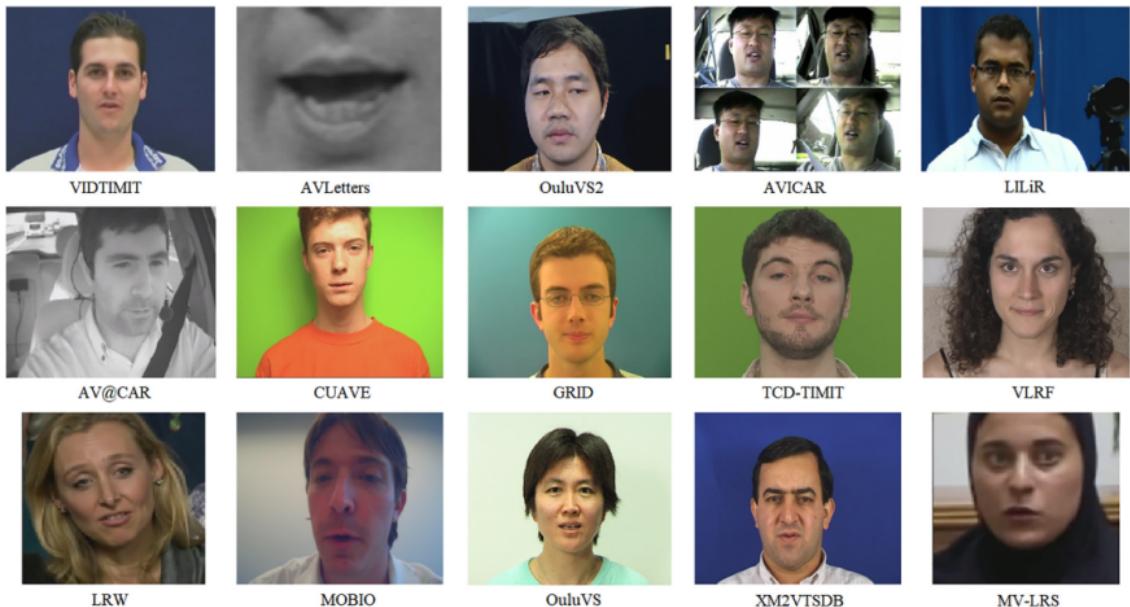


Figura 4: Diferentes Conjuntos de datos (Fernandez-Lopez y Sukno, 2018)

En la literatura se pueden encontrar diferentes tipos de datos, algunos incluyen una transcripción con los fonemas (Fernandez-Lopez, Martínez, y Sukno, 2017) o con las palabras (Rekik y cols., 2014), que serían las clases a predecir, en este trabajo se ha optado por trabajar con palabras completas. Algunos conjuntos de datos incluyen audio y transcripción y otros solo transcripción, la mayoría graban al hablante de frente para facilitar la tarea, aunque otros añaden una grabación lateral. Las grabaciones suelen hacerse en un sitio cerrado, pero algunos conjuntos presentan escenarios alternativos (un coche, exteriores o un plato de televisión). Antes de proceder a la siguiente sección es importante explicar la razón por la que se ha elegido trabajar con palabras completas y no con fonemas, a pesar de que trabajar con fonemas es la metodología adecuada cuando se trabaja con Modelos Ocultos de Markov (Gales y Young, 2008). En primer lugar, se ha elegido trabajar con palabras completas porque el conjunto de datos con el que se empezó a trabajar no contenía un etiquetado a nivel de fonema, sino de palabras completas, dado el desconocimiento se empezó a trabajar de esta manera. En segundo lugar, se ha elegido trabajar con palabras porque parece una tarea más sencilla y dado el tiempo que se ha dedicado a elaborar este trabajo se ha elegido trabajar de esta manera.

Para este trabajo se ha supuesto un entorno no realista, las imágenes del conjunto de datos utilizado son de este tipo. Como ejemplo de un entorno realista se puede observar la Figura 8, se considera que las imágenes del conjunto de datos que se ha utilizado no presenta un entorno realista porque no estudia la posición de los labios desde otro ángulo

que no sea el frontal, además los hablantes no se mueven mientras son grabados, en la Figura 8 se puede ver que el interlocutor no está hablando centrado en la imagen, la tarea sería más complicada si se asumiese un entorno realista, pero debido a qué no se quiere complicar en exceso el trabajo se ha decidido optar por un entorno no realista, si se quisiese optar por un entorno realista se deberían de usar los conjuntos de datos Lip Reading in the Wild o RTVE2020, que están compuestos por programas reales de la BBC y de RTVE respectivamente.

2.2. Sistemas Automáticos de Lectura de Labios Tradicionales



Figura 5: Reconocimiento de la cara y puntos de interés faciales (Gurjar, 2016)

En la Figura 6 se puede ver como se ha aplicado un algoritmo de detección de la cara y otro para la detección de los puntos de interés de la cara, en el apartado **4.1** se detalla como se ha aplicado este procesamiento al conjunto de datos utilizado, para este apartado se supone que los datos son como los de la Figura 13. En las aproximaciones tradicionales, la arquitectura de los sistemas puede cambiar un poco si el objetivo es el reconocimiento de dígitos y letras o el de frases y palabras, pero en líneas generales son muy similares. Estos sistemas empiezan detectando las caras y extrayendo las regiones de la boca y alrededores, una vez los labios se han localizado, técnicas para la extracción de las características visuales se aplican, el objetivo de estas técnicas es conseguir una representación de menor dimensión de las imágenes originales (**apartado 4.2**) diferentes técnicas se han propuesto, algunas de ellas se centran en características geométricas como la altura, anchura y área bucal (ASM, *Active Shape Models*), otras intentan transformar la imagen de tal manera que se quedan con las partes más relevantes de ella (como Análisis de Componentes Principales o los llamados *Active Appearance Models*, AAM), otras intentan capturar el movimiento producido entre dos fotogramas como las técnicas *Optical Flow*. Finalmente, con el paso de los años se empezaron a usar redes neuronales para la extracción de características con la utilización de *Autoencoders* y Redes Neuronales Convolucionales. En la parte de clasificación existe una popularidad del modelo conocido como Modelo Oculto de Markov (en inglés *Hidden Markov Model* o HMM), pero también se ha visto el uso de la Máquina de Soporte Vectorial (en inglés *Support Vector Machine* o SVM). En la Figura 6 se puede ver un conteo del número de veces que estas diferentes técnicas han sido usadas, se puede ver que destacan los HMM y SVM comentados para la clasificación, así como los AAM o PCA (*Principal Component Analysis*) para la extracción de características, entre otros.

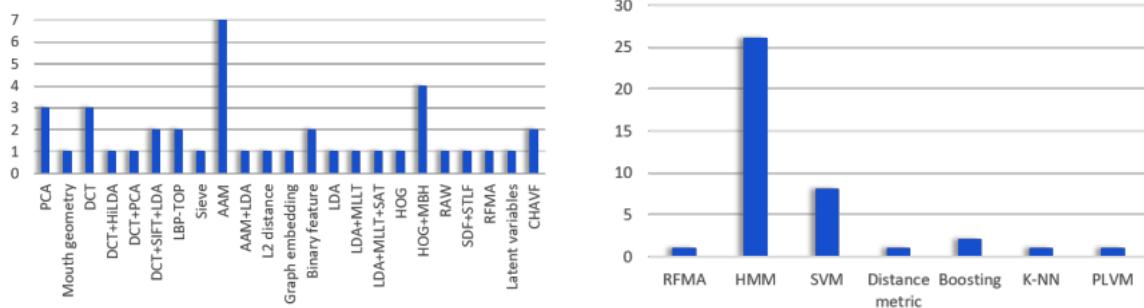


Figura 6: El número de veces que un extractor de características (izquierda) y un clasificador (derecha) ha sido usado del 2007 al 2017 (Fernandez-Lopez y Sukno, 2018)

A continuación se repasan los mejores resultados obtenidos con estos sistemas para reconocimiento de dígitos, letras del alfabeto y frases u oraciones. En el caso de los dígitos se puede destacar una publicación de Papandreou y colaboradores (2009), donde se consiguió un WRR del 83% en el conjunto de datos CUAVE, para la obtención de características se utilizó un modelo AAM y Modelos Ocultos de Markov para la clasificación. En el caso de letras del alfabeto destaca una publicación del año 2008 (Cox, Harvey, y Lan) donde se utilizó Análisis de Componentes Principales, AAM y filtros *Sieve* para la extracción de características y Modelos Ocultos de Markov para la clasificación, se obtuvo un 90 % de WRR (alfabeto) en el conjunto de datos AVLetters2.

Como se había comentado antes, el problema con la lectura de labios cuando los datos eran referentes a letras o a dígitos era que estas no eran situaciones realistas, conforme han avanzado los años los conjuntos de datos centrados en palabras y frases han aumentado en número con respecto a los centrados en letras y dígitos (Fernandez-Lopez y Sukno, 2018). Uno de los mejores resultados para frases u oraciones se encuentran en una publicación del año 2016 (Howell, Cox, y Theobald) con un WRR del 75.58 % usando un modelo AAM y Modelos Ocultos de Markov para la clasificación. Antes de proceder a examinar los sistemas basados en redes neuronales hay que destacar que una de las principales ventajas que ha concedido el usar Deep Learning es que se ha dejado de utilizar esta tecnología para frases, letras o dígitos y se ha empezado a trabajar con datos donde se trata con habla espontánea.

2.3. Sistemas Automáticos de Lectura de Labios basados en Redes Neuronales

Existe un paralelismo en la manera en la que las redes neuronales han sido adoptadas por los sistemas de reconocimiento del habla basados en audio y los basados en video, al principio se propusieron sistemas híbridos, combinando bloques tradicionales con redes neuronales, por ejemplo, se utilizaron las redes neuronales como extractores de características, combinadas con Modelos Ocultos de Markov como clasificadores. Posteriormente, se introdujeron redes neuronales recurrentes como clasificadores (por ejemplo, redes *Long-Short Term Memory*, LSTM). Recientemente, se han utilizado redes neuronales denominadas de principio a fin (end-to-end), que sustituyen todos los bloques de los sistemas tradicionales. (Fernandez-Lopez y Sukno, 2018). En la parte de extracción de características destacan los Redes Neuronales Convolucionales. En la clasificación destacan las Redes Neuronales Recurrentes, como LSTM o GRU (*Gated Recurrent Unit*). Como un

ejemplo de lo descrito hasta ahora se puede ver la Figura 8, en la imagen de la izquierda se ve como las CNN (*Convolutional Neural Networks*) han sido el extractor de características más utilizado, en la imagen de la derecha se ve como las LSTM o GRU (son arquitecturas diferentes, pero los autores de la revisión las han contado en la misma categoría) han sido la arquitectura más utilizada para el bloque de clasificación.

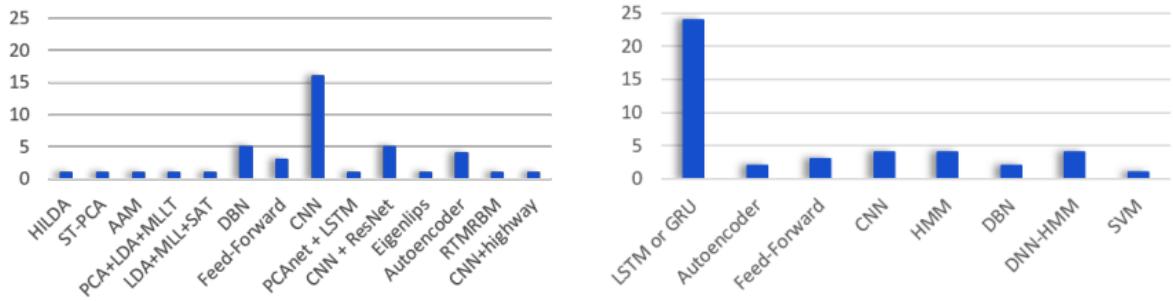


Figura 7: El número de veces que un extractor de características (izquierda) y un clasificador (derecha) ha sido usado del 2007 al 2018 (Fernandez-Lopez y Sukno, 2018)

Uno de los resultados más impresionantes utilizando modelos basados en Deep Learning es del año 2020 (Prajwal, Mukhopadhyay, Namboodiri, y Jawahar), los autores de esta publicación han creado un conjunto de datos, llamado Lip2Wav, con 120 horas de vídeo de 5 hablantes a partir de clases subidas a YouTube, como se puede ver en la Figura 7, uno de los aspectos reseñables de este trabajo es que la variable dependiente es el audio correspondiente a un vídeo y no cuentan con transcripciones, es decir el modelo final genera audio a partir del movimiento de los labios en vídeo. Además, este trabajo se centra en acumular un gran número de horas para cada hablante, algo que no ocurre con otros conjuntos de datos donde se encuentra un gran número de hablantes pero menos tiempo de vídeo para cada uno. La red neuronal usada es end-to-end, cuenta con dos bloques, el primero de ellos ha sido bautizado *Spatio-temporal Face Encoder* con el objetivo de conseguir una representación de los datos de menor dimensión, algo similar a lo que se hace en arquitecturas tradicionales en la extracción de características visuales, el modelo usado es el de Redes Neuronales Convolucionales 3D. El último bloque se ha bautizado como *Attention-based Speech Decoder*, cuya función es decodificar la representación creada por el bloque anterior y generar audio.

LeNet-5

Introduced the (now famous) MNIST dataset (LeCun et. al.)^[21]

3 6 8 1 7 9 6 6 9 1
6 7 5 7 8 6 3 4 8 5
2 1 7 9 7 1 2 8 4 6
4 8 1 9 0 1 8 8 9 4
7 6 1 8 6 4 1 5 6 0
7 5 9 2 6 5 8 1 9 7
2 2 2 2 2 3 4 4 8 0
0 2 3 8 0 7 3 8 5 7
0 1 4 6 4 6 0 2 4 3
7 1 2 8 1 6 9 8 6 1

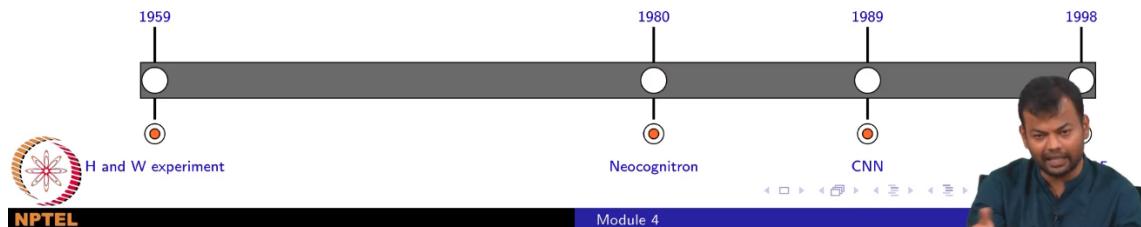


Figura 8: Fotograma del conjunto de datos Lip2Wav

2.4. Aplicaciones

En este apartado destaca la start-up Liopa (O'Halloran, 2021), en la actualidad ofrecen una aplicación para teléfonos inteligentes, esta aplicación está siendo actualmente utilizada en hospitales del Reino Unido dirigida a personas que no pueden hablar, a partir de la cámara del teléfono permite leer los labios de una persona (Figura 9), la empresa declara que el modelo aprende mientras es usado y que cuenta con más de un 90 % de precisión a nivel de frases, según se puede leer en su página web utilizan un sistema basado en redes neuronales pero dado el carácter comercial de la empresa no se han compartido mayores detalles (Liopa, 2022).

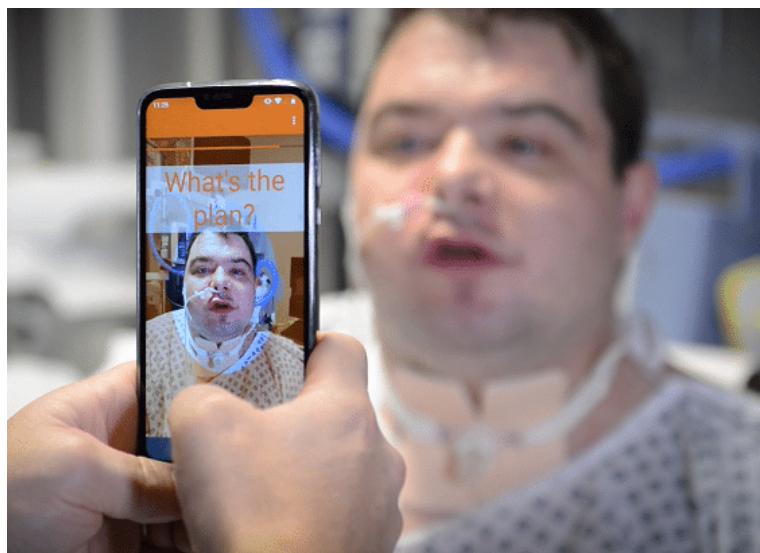


Figura 9: SRAVI: Speech Recognition App for The Voice Impaired (Liopa, 2022)

Otro de los servicios que ofrece esta empresa es la detección de frases o palabras específicas, actualmente están implementando esta tecnología en cámaras de vigilancia. Además de los productos que oferta Liopa otras de las posibles aplicaciones de esta tecnología podrían ser:

- Utilización en la transcripción de películas como complemento al audio, con el objetivo de mejorar el rendimiento de los modelos.
- Contraseñas visuales silenciosas.
- Subtitulado en videollamadas en entornos donde no se puede hacer ruido o con demasiado ruido como para que se pueda entender el audio.

3. Descripción del conjunto de datos

3.1. *MIRACL-VC₁*

El conjunto de datos que se ha utilizado para realizar este trabajo ha sido el llamado *MIRACL-VC₁* (Rekik y cols., 2014), este conjunto de datos cuenta con grabaciones, de vídeo, sin audio, correspondientes a 15 personas, cada persona pronuncia las 10 frases y las 10 palabras que se ven en el Cuadro 1, cada frase y cada palabra es repetida 10 veces por cada hablante. Por lo tanto, cada palabra y frase cuenta con 150 repeticiones, en total se tienen 1500 pronunciaciones de palabras y 1500 pronunciaciones de frases. Como algunas personas hablan más rápido que otras y algunas palabras son más largas que otras, cada observación del conjunto de datos tiene un número distinto de fotogramas, por ejemplo una persona puede pronunciar "hello" en 10 fotogramas y otra en 8, esto añade complejidad al trabajo ya que se tienen que clasificar secuencias de distinta longitud. Este conjunto de datos presenta algunas particularidades con respecto a otros, por ejemplo los datos están almacenados en carpetas como fotogramas (imágenes) con un orden denotado por el nombre de los archivos (color_001.jpg, color_002.jpg, color_003.jpg...), otros conjuntos vienen en formato de vídeo, en ese caso habría que tratar el vídeo para conseguir los fotogramas correspondientes.

Se han creado dos conjuntos de datos, uno con palabras y otro con frases, se ha decidido dividir en dos conjuntos de datos en lugar de tener uno con 20 clases porque las secuencias de fotogramas de palabras son considerablemente más cortas que las de frases y esto puede influir en el rendimiento del clasificador. A su vez los dos conjunto de datos se han dividido en entrenamiento, validación y test. El conjunto de datos de test sirve para validar la calidad del modelo con datos que no se han utilizado para el entrenamiento, se ha elegido de manera aleatoria a los hablantes con las etiquetas F10 y M02. El conjunto de validación es utilizado en este trabajo para realizar el llamado ajuste de hiperparámetros, los hiperparámetros son opciones de configuración en un modelo que debe elegir el creador, por ejemplo el número de neuronas, el tipo de capa o la función de activación en una red neuronal o la topología de un Modelo Oculto de Markov, se utiliza un conjunto de validación distinto al de test para que la elección de estos hiperparámetros no afecte al medir la calidad del modelo en el conjunto de datos de test, para el conjunto de validación se ha elegido aleatoriamente al hablante con la etiqueta F11. En los apartados 4 y 5 se explicará la implementación en los dos conjuntos de datos, sin embargo, se hará referencia solo al conjunto de datos de palabras ya que se han aplicado las mismas transformaciones a ambos conjuntos de datos y esto facilitará la comunicación al evitar repetir que se ha realizado lo mismo en los dos conjuntos, pero se comunicarán los resultados obtenidos para ambos conjuntos de datos.

ID	Palabras	ID	Frases
1	Begin	1	Stop navigation
2	Choose	2	Excuse me
3	Connection	3	I am sorry
4	Navigation	4	Thank you
5	Next	5	Good bye
6	Previous	6	I love this game
7	Start	7	Nice to meet you
8	Stop	8	You are welcome
9	Hello	9	How are you?
10	Web	10	Have a good time

Cuadro 1: Variable dependiente

4. Sistemas Tradicionales

En este apartado se va a proceder a explicar los bloques fundamentales que constituyen un sistema automático de lectura de labios, se explicarán los algoritmos que se utilizan en estos bloques y los resultados que se han obtenido utilizando el lenguaje de programación Python. El código se puede consultar en el repositorio de GitHub del proyecto (Saavedra, 2022).

4.1. Detección de los labios

La primera acción que se debe tomar en este sistema es la detección del área de los labios, al contrario que el resto de bloques, este bloque será explicado de manera superficial, ya que la detección de caras y puntos de interés faciales es un tema complejo que podría dar para otro Trabajo Final de Grado y no es el tema de estudio en este trabajo. Para la detección del área de los labios se ha hecho uso de dos librerías de Python: *OpenCV* y *dlib*. El procesado que se va a explicar en este apartado se aplica a cada uno de los fotogramas que componen el conjunto de datos, el objetivo es procesar los datos de tal manera que en las observaciones solo esté presente la región de los labios, esto se denomina la Región de Interés. A continuación, se puede ver la imagen de la izquierda de la figura 11, la cual es un ejemplo de la imagen de la que se quiere extraer su región de interés.



Figura 10: Fotograma inicial a la izquierda y a la derecha región de la cara detectada

En primer lugar, se necesita detectar la región de la cara, como se ha hecho en la imagen de la derecha, la librería **dlib** escogida incluye un modelo pre-entrenado, que implementa una técnica conocida como Histogram of Gradients (HOG) combinada con un clasificador que en este caso es un Maquina de Soporte Vectorial (Dalal y Triggs, 2005), aunque este haya sido el método utilizado es importante recalcar que dentro de la librería usada existen otras alternativas como el uso de arquitecturas Deep Learning, se ha escogido la opción de HOG + Máquina de Soporte Vectorial porque funciona correctamente y es más eficiente en cuanto a recursos que la opción con redes neuronales.

Una vez se ha detectado la cara, se procede a localizar los landmarks faciales o puntos de interés, en la Figura 12 podemos ver como se han calculado los landmarks faciales para la imagen sugerida. Para la localización de los landmarks de nuevo se ha utilizado la librería *dlib*, el modelo que se ha usado ha sido un modelo pre-entrenado de la librería.



Figura 11: Puntos de interés o landmarks faciales

A continuación, se busca dibujar un rectángulo alrededor de la zona de la boca, gracias a la documentación se sabe que los puntos del 48 al 67 son los que están en la zona de los labios, se guardan con estos puntos y manipulándolos se consiguen construir un rectángulo alrededor de la zona de los labios, para mayor detalle consultar el repositorio de GitHub del proyecto (Saavedra, 2022).



Figura 12: Área de los labios

Finalmente, se recorta para cada una de las imágenes del conjunto de datos la región de los labios. Distintos fonemas producen distintas formas de la boca, como se ve en la Figura 14, por este motivo algunas imágenes tendrán un tamaño más grande que otras, por ello se ha decidido re-escalar todas las imágenes al mismo tamaño, todas tienen unas dimensiones de 27 píxeles de ancho y 13 píxeles de alto, se ha optado por estas dimensiones porque son el tamaño medio alto y ancho de las imágenes en los datos.

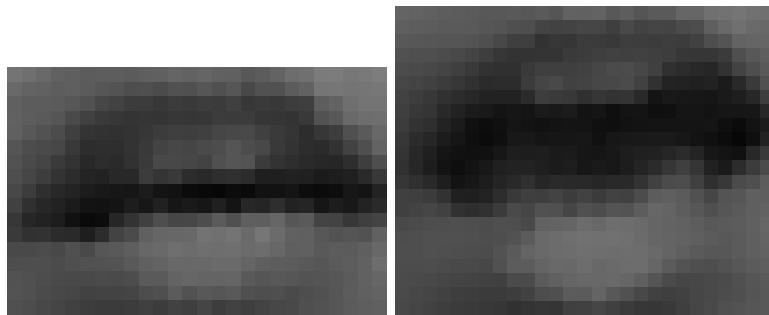


Figura 13: Recorte de la región de la boca

4.2. Extracción de las características visuales

Si se estuviese trabajando con otro tipo de datos, por ejemplo, para predecir el precio de una casa o calcular la probabilidad de que una persona tenga diabetes, una característica podría ser la superficie de la casa o el nivel de actividad física de la persona. Una característica es una de las variables independientes con la que se intenta modelar el comportamiento de una variable Y . En el campo de la Visión Artificial una característica es una pieza de información que distingue a una observación de otra, en este caso las observaciones son cada uno de los fotogramas que componen un vídeo (al que le corresponde una palabra), la extracción de características es una parte importante en Visión Artificial, el objetivo es conseguir una representación de la imagen original que no recoja la información irrelevante y se quede con las partes relevantes de la imagen (learnopencv, 2016). Se debe recordar que cada imagen se había re-escalado para tener unas dimensiones de 27 píxeles de ancho y 13 de alto, con la extracción de características se pretende conseguir para cada fotograma un vector de características de tamaño n . Otra de las razones por las que se utiliza esta técnica es porque mejora el rendimiento del clasificador.

4.2.1. Eigenlips - Análisis de Componentes Principales

El algoritmo que se va a utilizar es el de Análisis de Componentes Principales, en la literatura se ha bautizado a esta técnica como eigenlips (Delac, Grgic, y Liatsis, 2005) debido al nombre en inglés de los autovalores y autovectores (eigenvalues y eigenvectors). Análisis de Componentes Principales calcula una representación de los datos de menor dimensión, dado los datos originales $x_1, x_2, x_3, \dots, x_p$ se busca encontrar un nuevo conjunto de variables no correlacionadas $z_1, z_2, z_3, \dots, z_q$ donde $q < p$ y cada z_j es una combinación lineal de las variables x_j . Estas nuevas variables son conocidas como los componentes principales, para calcular los componentes principales se procede de la siguiente manera, dada la matriz de covarianza muestral de media 0 y con desviación típica 1:

$$S = \frac{1}{n-1} X^T X$$

se calcula la matriz de autovectores A de la matriz S , entonces:

$$Z = XA$$

Además, los autovalores correspondientes a los autovectores de A representa la cantidad de varianza que retiene cada componente, de esta manera gracias a los autovalores se puede calcular el porcentaje de variabilidad de los datos originales que retienen los componentes principales elegidos. Para aplicar el algoritmo se necesita una matriz, pero los datos que se

han utilizado no están en este formato, sino que se tiene una matriz para cada imagen, y n matrices (imágenes) para cada palabra, porque como se había visto antes, no todas las palabras estaban compuestas por el mismo número de fotogramas. Para poder aplicar el algoritmo, cada matriz se transforma a vector, es decir, las matrices de 27×13 se convierten a un vector de 351 elementos, y se apilan todas las imágenes del conjunto de datos por filas, se guardan el orden de fotograma y la palabra correspondiente a la fila para una vez se aplique el algoritmo y se obtenga la representación en menor dimensión, se pueda recuperar la naturaleza secuencial de los datos. Una vez aplicado el algoritmo y eligiendo que se retenga un 90 % de la varianza aproximadamente, se escogen 10 componentes principales, en la figura 8 se puede ver que a partir de 10 componentes no se obtiene una mejora significativa al aumentar el número de componentes.

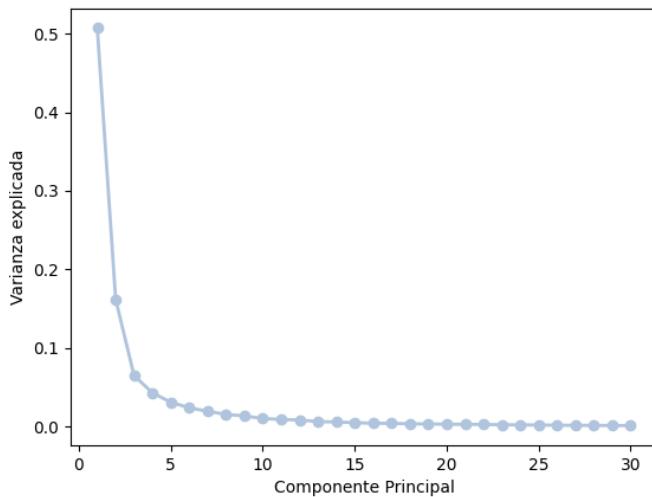


Figura 14: Gráfico de sedimentación

A continuación, en la Figura 9 se pueden observar los autovectores resultantes de aplicar Análisis de Componentes Principales, al ajustar el algoritmo a los datos se ha obtenido una matriz de 10 componentes, cada componente es un vector de 351 elementos, se han modificado sus dimensiones para poder representarlo como una imagen y ahora tiene unas dimensiones de 27×13 (matriz), esta operación ha sido aplicada a todos los componentes y como resultado se ha obtenido la Figura 9. Esta es la razón por la que este método ha sido bautizado como eigenlips, porque cuando se representan los autovectores se obtienen figuras con forma de labios, distintos componentes atrapan distintas formas que están presentes en el conjunto de datos.



Figura 15: Autovectores - eigenlips

Posteriormente, se aplica la siguiente transformación a los datos originales, para conseguir la matriz de componentes principales:

$$Z_1 = X_{entrenamiento}A, Z_2 = X_{validación}A, Z_3 = X_{test}A$$

Recalcular que la transformación se realiza tanto a los datos de entrenamiento, como en los de validación y los de test, pero la matriz A ha sido conseguido con los datos de entrenamiento, esto se hace para que a la hora de evaluar la calidad del modelo los datos de test no influyan en la calidad del modelo.

4.3. Clasificación

Una vez se ha conseguido un vector con las características para cada fotograma, se procede a la parte de la clasificación, el objetivo es clasificar una secuencia de fotogramas de longitud variable en una de las 10 palabras o frases que componen las clases del conjunto de datos (Cuadro 1), la complejidad reside en que para conseguir mejores resultados se debería aprovechar la naturaleza secuencial de los datos, por este motivo la mayoría de modelos de clasificación no son la mejor opción para resolver este problema, cómo podrían ser árboles de decisión o regresión logística. En los siguientes tres apartados se procederá a probar el rendimiento de dos modelos:

4.3.1. Máquina de Soporte Vectorial

Como primera aproximación para clasificar el conjunto de datos se va a utilizar uno de los algoritmos que han sido más populares (en este campo) antes del uso de Deep Learning (Fernandez-Lopez y Sukno, 2018), la Máquina de Soporte Vectorial. La intuición detrás de este algoritmo es intentar encontrar un hiperplano que esté lo más alejado posible de las dos clases como se puede ver en la imagen izquierda de la Figura 17, se llama de Soporte Vectorial porque los vectores que están señalados con flechas son los vectores de soporte que determinan el hiperplano (esto se conoce como Clasificador de Máximo Margen), sin embargo, encontrar este hiperplano para la mayoría de datos, ya que es muy complicado separar completamente dos clases, por ello se propuso el Clasificador de Soporte Vectorial que se puede ver en la imagen de la derecha, esta técnica introduce un parámetro C que representa el número máximo de observaciones que pueden estar en el lado del hiperplano incorrecto, es decir mal clasificadas, esto introduce flexibilidad en el modelo, el ajuste de estos algoritmos a los datos se hace a través de la optimización de un problema de programación cuadrática (Geron, 2019).

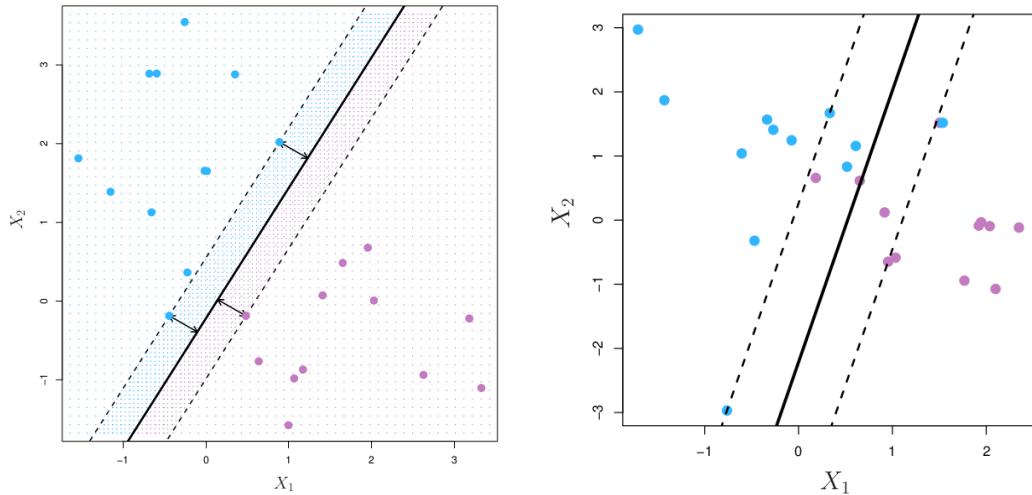


Figura 16: Clasificación con Máquina de Soporte Vectorial

Dada la limitación del Clasificador de Soporte Vectorial de solo poder ajustar datos linealmente separables, se propuso la Máquina de Soporte Vectorial que permite ajustar datos no lineales gracias a la utilización del llamado truco del kernel, que consiste en añadir una nueva dimensión a los datos en la que podamos encontrar un hiperplano que separe ambas clases. Se ha explicado una Máquina de Soporte Vectorial para dos clases para no complicar el trabajo, sin embargo, habría que modificar el problema de optimización para que pueda ser usada con más de dos clases.

Dado que este algoritmo no permite incorporar la naturaleza secuencial de los datos, estos se tienen que procesar de tal manera que se tenga un vector X_i (que representa una secuencia de fotogramas) de dimensión n y una etiqueta y (con la clase correspondiente). Esto se ha conseguido concatenando los vectores obtenidos en la extracción de características para una secuencia de fotogramas en un solo vector (esto se hace para todas las secuencias del conjunto de datos). Para que todos los vectores tengan la misma longitud se ha asignado a todos la longitud del más grande y se ha añadido ceros donde fuese necesario, la implementación de este modelo se ha realizado con la librería **scikit-learn**. A continuación, en el Cuadro 2 se puede ver el WRR para ambos conjuntos de datos y las matrices de confusión (Cuadros 3 y 4).

	Palabras	Frases
Entrenamiento	99.58 %	100 %
Validación	61 %	58 %
Test	34 %	41 %

Cuadro 2: WRR de la Máquina de Soporte Vectorial

	Palabra Predicha										
	Begin	Choose	Connection	Navigation	Next	Previous	Start	Stop	Hello	Web	
Palabra Correcta	Begin	14	4	0	0	1	0	1	0	0	0
	Choose	1	6	5	1	0	0	1	0	5	1
	Connection	0	3	13	3	0	0	0	0	0	1
	Navigation	0	9	1	10	0	0	0	0	0	0
	Next	0	1	0	7	0	0	8	4	0	0
	Previous	12	3	1	0	1	2	0	0	0	1
	Start	0	3	2	2	0	0	12	0	1	0
	Stop	1	1	0	4	1	0	3	4	4	2
	Hello	0	6	6	4	0	0	0	0	2	2
	Web	0	2	3	0	7	0	2	0	1	5

Cuadro 3: Matriz de Confusión para palabras, Máquina de Soporte Vectorial, conjunto de test

	Frases Predichas										
	Stop Navigation	Excuse me	I am sorry	Thank you	Good bye	I love this game	Nice to meet you	You are welcome	How are you?	Have a good time	
Frase Correcta	Stop Navigation	9	0	1	4	1	1	0	2	0	2
	Excuse me	1	2	0	9	4	2	0	1	0	1
	I am sorry	0	6	5	4	1	0	0	0	0	4
	Thank you	0	1	0	10	0	0	9	0	0	0
	Good bye	3	3	6	0	6	0	0	1	0	1
	I love this game	1	0	3	0	2	3	0	4	1	6
	Nice to meet you	0	0	1	2	0	0	7	0	10	0
	You are welcome	1	0	0	0	5	0	0	13	1	0
	How are you?	0	0	0	0	0	0	0	0	20	0
	Have a good time	0	6	0	0	2	0	0	2	4	6

Cuadro 4: Matriz de Confusión pra frases, Máquina de Soporte Vectorial, conjunto de test

Al observar el Cuadro 2 llama la atención el WRR obtenido en los datos de entrenamiento, parece indicar un claro sobreajuste, sin embargo, al aplicar regularización al modelo se reducía también el WRR en el conjunto de validación, por ello se ha optado por usar como modelo definitivo el modelo que se ajusta de manera perfecta a los datos de entrenamiento. Aunque el WRR sea considerablemente menor en los conjuntos de validación y test, no se consideran malos resultados debido a la complejidad de la tarea. Antes de analizar las matrices de confusión se debe recordar que el conjunto de test está formado por 200 observaciones, 20 para cada palabra. En la matriz de confusión de palabras se observa que no hay ninguna clase que haya sido predicha con mayor frecuencia que las demás, aunque hay algunas palabras que el modelo predice de manera menos frecuente como *Previous*, también llama la atención que la palabra *Begin* haya sido predicha como la palabra *Previous* en 12 ocasiones. Con respecto a la matriz de confusión de frases, llama la atención que la frase *How are you?* ha sido predicha bien siempre, sin embargo, también se ve que ha sido la frase que el modelo ha predicho con mayor frecuencia, de nuevo, como en el modelo entrenado con el conjunto de datos de palabras, no hay ninguna clase que haya sido predicha considerablemente más o menos veces que las demás.

4.3.2. Modelo Oculto de Markov

En los sistemas tradicionales el Modelo Oculto de Markov ha sido la técnica de clasificación más utilizada (Fernandez-Lopez y Sukno, 2018), en este trabajo se ha entrenado un Modelo Oculto de Markov para cada una de las palabras utilizando los datos de entrenamiento, una vez se tienen todos los modelos posibles se procede a calcular cuál de ellos tiene la mayor probabilidad de haber generado una secuencia de fotogramas dada (aunque se debe recordar que cada fotograma se transformó a vector en la sección 4.2), el modelo que maximice la probabilidad será el asignado en la clasificación, el objetivo es calcular $P(X|\lambda)$, donde X es una secuencia de fotogramas y $\lambda = (A, B, \pi)$ es una Modelo Oculto

de Markov con probabilidades de transición A, probabilidades de emisión B, y vector estacionario π , a continuación se explicarán los componentes de un Modelo Oculto de Markov. El Modelo Oculto de Markov utiliza la idea de las Cadenas de Markov donde tenemos un grafo con los distintos estados, una matriz de transición A donde cada elemento A_{ij} representa la probabilidad de transición del estado i al j y un vector con la distribución estacionaria que es la probabilidad de que el primer estado sea el elemento i del vector.

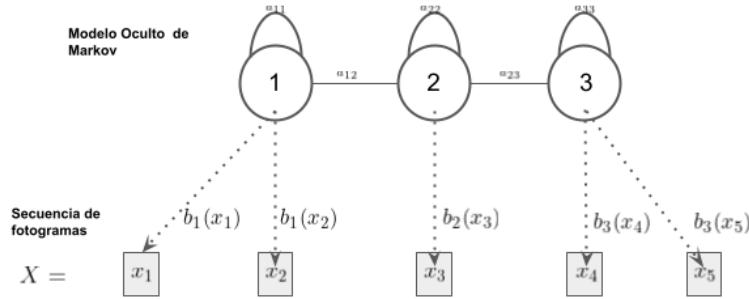


Figura 17: Diagrama de un Modelo Oculto de Markov con 5 observaciones y 5 estados ocultos

En la Figura 18 se puede ver un diagrama de un Modelo Oculto de Markov. En el gráfico las a_{ij} son las probabilidades de transición de un estado a otro, estas probabilidades están agrupadas en la matriz A, los estados son los encargados de modelar la naturaleza secuencial de los datos. $b_j(x_i)$ es la probabilidad de que una observación x_i sea generada por un estado i, esta probabilidad está definida por una distribución de probabilidad que en el caso del Reconocimiento del Lenguaje es habitual usar Modelos de Mixturas de Gaussianas, a la hora de definir un Modelo Oculto de Markov también se deben seleccionar el número de estados y la dimensión de la función de probabilidad que define las emisiones. Una vez se han especificado los parámetros de este modelo es necesario comentar que se establecen las siguientes asunciones:

- Los estados solo dependen del estado anterior.
- Las observaciones son condicionalmente independientes de las otras observaciones dado el estado que las ha generado.

Para poder utilizar este modelo se tienen que resolver dos problemas, en primer lugar, se tiene que calcular $P(X|\lambda)$, es decir, cuál es la probabilidad de que un Modelo Oculto de Markov genere una secuencia de características X, para resolver este problema se tiene calcular $P(X|q, \lambda)$ donde q son los estados del modelo, es decir se tendría que calcular la probabilidad de que un Modelo Oculto de Markov con unos estados q genere una secuencia X , esto se calcula de la siguiente manera:

$$P(X|q, \lambda) = \prod_{i=1}^T P(x_t|q_t, \lambda)$$

$$P(q|\lambda) = \pi_{q_1} a_{q_1 q_2} a_{q_2 q_3} \dots a_{q_{T-1} q_T}$$

$$P(X|\lambda) = \sum_q P(X|q, \lambda) P(q|\lambda)$$

Sin embargo, dada la complejidad algorítmica de la operación, en la práctica no es posible realizar ese cálculo, por ello para resolver este problema se hace uso del llamado *The Forward Algorithm* que hace uso de técnicas de programación dinámica y consiste en reutilizar probabilidades ya calculadas de manera recursiva. Por último, para entrenar el modelo, es decir, estimar las matrices A y B, se utiliza el algoritmo Baum-Welch (Jurafsky y Martin, 2009), no sé profundizará en él debido a su complejidad.

Para la implementación de este modelo se ha utilizado la librería **sequentia** de Python. Después de aplicar Análisis de Componentes Principales se obtuvo una matriz de dimensiones (12701×10) (para los datos de entrenamiento) 120701 son el número de fotogramas que componen los datos de entrenamiento, pero esta matriz no tiene en cuenta la secuencialidad de los datos, por lo tanto, se ha transformado a una lista de longitud 1200, cada lista tiene en su interior una matriz de dimensiones $(\#Fotogramas \times 10)$ (10 son los componentes extraídos para cada fotograma). Además, se ha estandarizado el tamaño del número de fotogramas al tamaño de la secuencia más larga, de esta manera cada lista tiene en su interior una matriz de dimensiones 22×10 . Una vez se tiene está estructura de datos, se ajuste un Modelo Oculto de Markov, con 3 estados y con emisiones definidas por Modelos de Mixturas de Gaussianas con 7 componentes, para cada una de las clases. A continuación, en el Cuadro 3 se puede ver el WRR del modelo y las matrices de confusión para ambos conjuntos de datos obtenidas tras aplicar el procedimiento descrito.

	Palabras	Frases
Entrenamiento	41 %	34 %
Validación	20 %	10 %
Test	16 %	22 %

Cuadro 5: WRR de los Modelos Ocultos de Markov

Palabra Correcta	Palabra Predicha										
	Begin	Choose	Connection	Navigation	Next	Previous	Start	Stop	Hello	Web	
Begin	1	5	0	0	2	0	0	0	1	11	
Choose	0	0	0	0	0	0	0	0	1	19	
Connection	0	0	1	0	0	5	0	1	0	13	
Navigation	0	2	0	0	1	6	0	0	9	2	
Next	6	0	0	0	1	0	0	0	3	10	
Previous	5	0	0	0	0	0	0	0	5	10	
Start	0	0	0	0	0	0	7	0	3	10	
Stop	0	0	0	0	0	0	0	0	10	10	
Hello	0	1	0	0	0	0	0	0	9	10	
Web	2	4	0	0	0	0	0	0	2	12	

Cuadro 6: Matriz de Confusión palabras, Modelo Oculto de Markov

Frase Correcta	Frase Predicha										
	Stop Navigation	Excuse me	I am sorry	Thank you	Good bye	I love this game	Nice to meet you	You are welcome	How are you?	Have a good time	
Stop Navigation	0	0	2	9	1	0	1	0	7	0	
Excuse me	0	7	7	6	0	0	0	0	0	0	
I am sorry	0	3	3	10	1	0	3	0	0	0	
Thank you	0	0	6	4	0	0	7	0	3	0	
Good bye	0	0	0	10	3	0	4	0	3	0	
I love this game	0	1	1	0	10	0	2	0	6	0	
Nice to meet you	0	1	0	0	10	0	7	0	2	0	
You are welcome	0	1	0	0	6	0	0	5	8	0	
How are you?	0	0	0	0	8	0	1	2	9	0	
Have a good time	0	0	0	0	11	0	0	2	7	0	

Cuadro 7: Matriz de Confusión frases, Modelo Oculto de Markov

Dados los resultados observados se puede decir que el modelo no se ha ajustado bien a

los datos, en la matriz de confusión para palabras se observa que el modelo predice en un gran número de ocasiones la palabra *web*, esto puede ser debido a que es la palabra más corta, lo que implica que es una secuencia más corta que las demás, cabe la posibilidad que esto haya sido lo que ha ocasionado que esta clase haya sido predicha con mayor frecuencia que las demás. También se observa que las palabras más largas *Connection* y *Navigation* no son predichas en ninguna ocasión, esto puede servir como evidencia de que el modelo tiene problemas con las secuencias de distinta longitud, porque aunque todas las secuencias tienen la misma longitud, la tienen porque se ha llenado con ceros donde ha sido necesario, por ello las secuencias correspondientes a *web* tendrán más ceros que las correspondientes a *Connection*. Con respecto a la matriz de confusión para frases, de nuevo se ve este patrón donde el modelo no ha predicho en ninguna ocasión algunas frases. Según se ha podido ver en los materiales que se ha utilizado para esta sección, como soporte, la manera adecuada de trabajar con el Modelo Oculto de Markov es con fonemas y no con palabras, ya que los fonemas no presentan esta diferencia de longitud tan reseñable, este ha podido ser el motivo por el cual el modelo presenta un rendimiento mediocre.

La información de este apartado ha sido recogida del libro *Applications of Hidden Markov Models in Speech Recognition* (Gales y Young, 2008) y el curso online *Introduction to Computer Vision* (Georgia Tech, 2014)

5. Deep Learning

En las últimas dos décadas la disponibilidad de grandes volúmenes de información y la capacidad para entrenar modelos que usen estos datos gracias a algoritmos más eficientes (Hinton, Osindero, y Teh, 2006) y mejor hardware ha posibilitado la resolución de problemas utilizando redes neuronales, esto es debido a que para entrenar estos modelos es necesario contar con muchos datos y un gran poder computacional. Como se ha podido ver en el apartado de sistemas tradicionales, la tarea se divide en distintos bloques, En esta sección se procede a mostrar los resultados obtenidos con una red neuronal end-to-end, esta red también estará constituida por dos bloques, el primero será una Red Neuronal Convolutacional que realiza una serie de operaciones y devuelve una representación de menor dimensión, el segundo bloque estará compuesto por una Red Neuronal Recurrente que permita modelar la naturaleza secuencial de los datos.

5.1. Redes Neuronales

En primer lugar, se explicará una de las redes neuronales más simples que servirá para entender otras arquitecturas más complejas utilizadas , este tipo de redes en inglés se llaman *Feedforward Neuronal Networks* o Redes Neuronales con propagación hacia delante, esto es porque dada una entrada propagan la información hacia delante, además el tipo de capa recibe el nombre de totalmente conectada para diferenciarla de otras, el material de esta sección se ha elaborado a partir del curso online de Andrew NG sobre Machine Learning (2012).

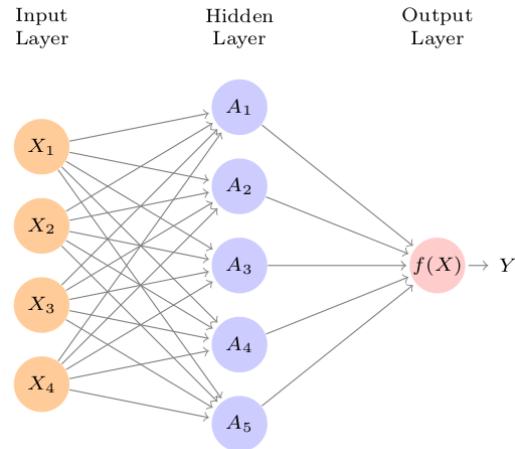


Figura 18: Diagrama de una red neuronal con 4 entradas, 1 capa oculta y 1 valor como salida (James y cols., 2013)

Se explicará siguiendo el ejemplo de la Figura 19, cada uno de los nodos son llamados neuronas, y cada grupo de neuronas apiladas verticalmente constituyen las capas de la red. La primera capa está compuesta por 4 neuronas que corresponden a las entradas que se introducen al modelo, por ejemplo podrían ser 4 características correspondientes a una imagen o características de un casa. Una vez definida la primera capa se procede a calcular cada una de las neuronas de la segunda capa de la siguiente manera:

$$A_j = g(X_1W_{1j} + X_2W_{2j} + X_3W_{3j} + X_4W_{4j})$$

donde a $g(x)$ se le suele denominar función de activación y puede ser cualquier función aunque en la práctica suele haber un grupo de funciones que son más populares debido a su rendimiento, las funciones de activaciones son clave, ya que permiten tener múltiples capas además de permitir que la red se pueda entrenar, la siguiente recibe el nombre de sigmoid (en la comunidad de Deep Learning):

$$g(x) = \frac{1}{1 + e^{-x}}$$

W_{ij} representa las flechas que conectan una neurona a otra, por ejemplo W_{12} representa la flecha que une la neurona 1 de la capa 1 con la neurona 2 de la capa 2, a su vez las fechas representan un número real que se denomina peso, estos pesos son aprendidos por la red en el proceso de entrenamiento. Una vez calculadas las neuronas de la capa intermedia se procede a calcular las neuronas de la capa de salida, B son los pesos que conectan la segunda capa con la capa de salida, se tendría:

$$f(x) = A_1B_{11} + A_2B_{21} + A_3B_{31} + A_4B_{41} + A_5B_{51}$$

Como se puede ver en este caso no se utiliza función de activación, el utilizar una función de activación o no dependerá del tipo de salida que se desee, por ejemplo si se quiere predecir un número real es probable que no se utilice función de activación, si se quiere clasificar entre dos clases se deberá usar la función $g(x)$ definida anteriormente, en el caso de la clasificación multiclas que ocupa este trabajo se debería usar la siguiente función de activación, con el objetivo de que la suma de todas las neuronas de la última capa sume 1 y poder elegir entre una de las 10 clases como la predicción:

$$\text{softmax}(x_i) = \frac{e^{x_i}}{\sum x_j}$$

Una vez se ha visto como se propaga la información hacia delante, hay que revisar la parte de entrenamiento (James y cols., 2013). La intención es la de minimizar la siguiente función:

$$\frac{1}{2} \sum (y_i - f(x_i))^2$$

para hacerlo se utiliza el algoritmo de Descenso de Gradiente, el cual es un método numérico que permite calcular el gradiente de una función. Este gradiente es utilizado en el algoritmo de Propagación Hacía Detrás, en el cual se utiliza la regla de la cadena para conseguir la derivada con respecto a los distintos pesos, de esta manera se puede minimizar la función anterior cambiando los pesos W y B .

5.2. Redes Neuronales Convolucionales

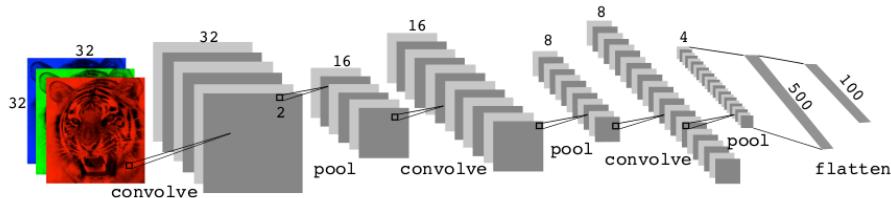


Figura 19: Red Neuronal Convolucional (James y cols., 2013)

Las Redes Neuronales Convolucionales han obtenido buenos resultados trabajando con imágenes y vídeos, cuentan con dos capas que no se habían visto en las redes del apartado

anterior, las capas de convolución y las de *pooling*. En las capas de convolución se utiliza una operación matemática llamada convolución, se puede ver el siguiente ejemplo.

$$\text{Imagen Original} = \begin{bmatrix} a & b & c \\ d & e & f \\ g & h & i \\ j & k & l \end{bmatrix}$$

$$\text{Filtro } 2 \times 2 = \begin{bmatrix} \alpha & \beta \\ \gamma & \delta \end{bmatrix}$$

$$\text{Imagen Convolucionada} = \begin{bmatrix} aa + b\beta + d\gamma + e\delta & ba + c\beta + e\gamma + f\delta \\ da + e\beta + g\gamma + h\delta & ea + f\beta + h\gamma + i\delta \\ ga + h\beta + j\gamma + k\delta & ha + i\beta + k\gamma + l\delta \end{bmatrix}$$

El filtro va recorriendo la imagen original en ventanas de 2x2 (el tamaño de la ventana se puede elegir por el creador del modelo), entonces se multiplica cada elemento del filtro con el elemento respectivo de la imagen, y se suman los resultados. Por ejemplo, para obtener el primer elemento de la imagen convolucionada se han multiplicado los elementos 1, 1, 1, 2, 2, 1 y 2, 2, de la imagen original por el filtro, y se han sumado. En la Figura 21 se puede ver un ejemplo de como modifican una imagen los filtros, se puede ver que el primer filtro captura mejor los bordes verticales y el segundo filtro captura mejor los bordes horizontales. Los valores que hay en el filtro son aprendidos durante el proceso de aprendizaje. El número de filtros que se aplican a una imagen en una capa son elegidos a la hora de crear el modelo.

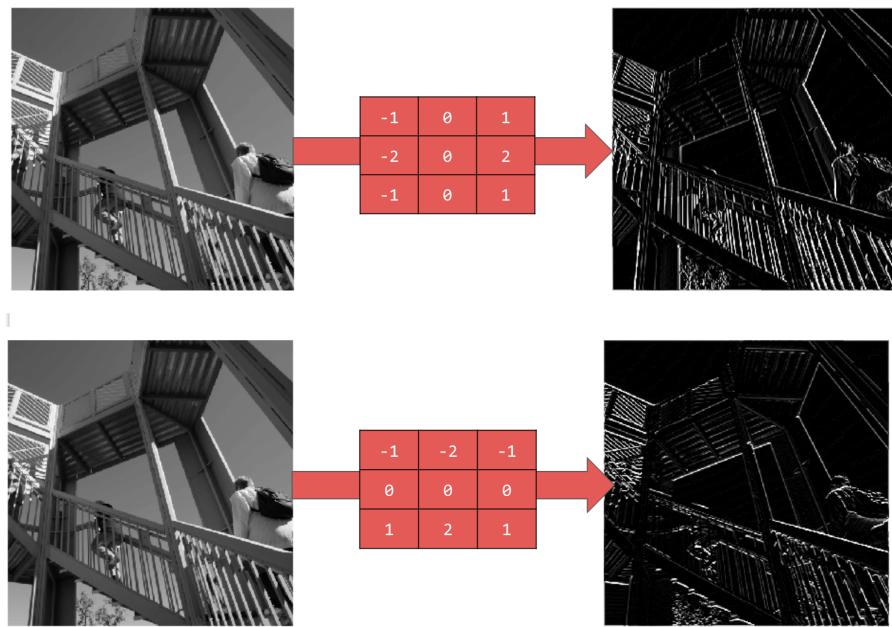


Figura 20: Demostración de dos filtros aplicados a una misma imagen (Geron, 2019)

Una vez se conoce la capa de convolución, queda conocer en que consiste una capa de *pooling*, estas capas intentan condensar imágenes grandes a más pequeñas, hay distintos algoritmos aplicables en esta capa, se verá el del *max pooling*, este algoritmo coge ventanas

de 2×2 (el tamaño de las ventanas se puede elegir) y selecciona el elemento máximo de esas ventanas, una vez ha recorrido todas las ventanas genera una nueva matriz con los elementos máximos escogidos para cada una de las ventanas.

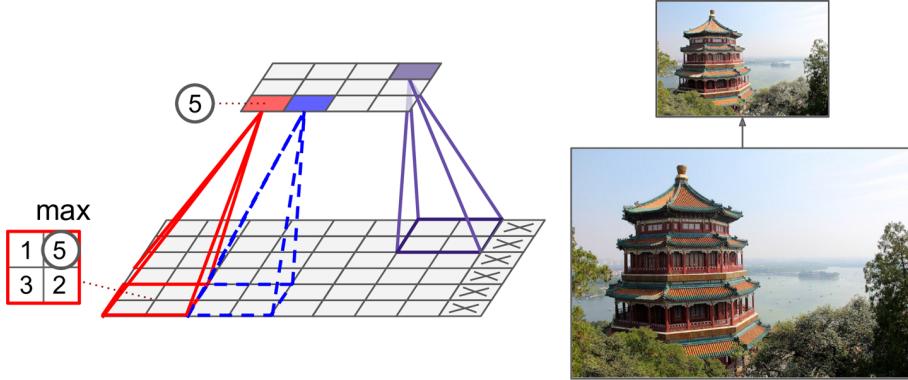


Figura 21: Demostración de una capa de pooling aplicada a una imagen (Geron, 2019)

$$\begin{pmatrix} 1 & 2 & 5 & 3 \\ 3 & 0 & 1 & 2 \\ 2 & 1 & 3 & 4 \\ 1 & 1 & 2 & 0 \end{pmatrix} \rightarrow \begin{pmatrix} 3 & 5 \\ 2 & 4 \end{pmatrix}$$

Finalmente, destacar que aunque se hayan visto estas dos capas de manera aislada a la hora de crear el modelo, se debe experimentar con el número de capas que se incluyen, teniendo que decidir cuántas capas convoluciones, cuántas de *pooling*, las dimensiones de los filtros, etc.

5.3. Redes Neuronales Recurrentes

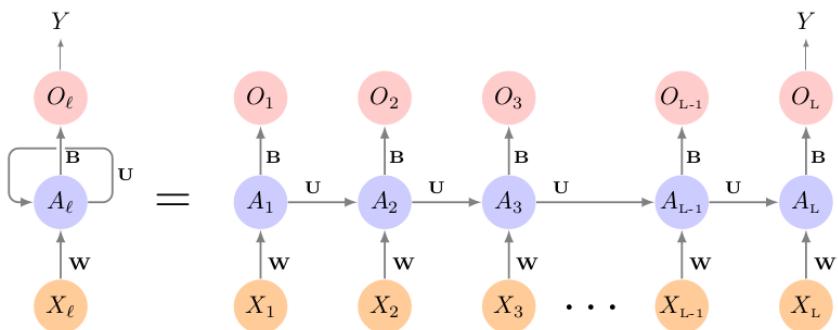


Figura 22: Red Neuronal Recurrente (James y cols., 2013)

Como se puede ver en la Figura 22, este tipo de red introduce como entrada una secuencia de vectores $\{X_l\}_1^L$, con una variable dependiente Y , cada X_l se introduce en la red de manera secuencial, cada A_l recibe como entrada el X_l correspondiente, además de A_{l-1} . W es una matriz de dimensiones $K \times (p + 1)$ donde K es el número de neuronas en la capa intermedia y $p + 1$ es la longitud del vector de entrada más uno, se le suma uno porque se le añade el intercepto o sesgo (como se le llama en la comunidad de Deep

Learning), U es una matriz de dimensiones $K \times K$, B es un vector de dimensiones $K + 1$ que se usan para calcular O_t .

$$A_{lk} = g(w_{k0} + \sum_{j=1}^p w_{kj} X_{lj} + \sum_{s=1}^K u_{ks} A_{l-1,s})$$

$$O_l = \beta_0 + \sum_{k=1}^K \beta_k A_{lk}$$

En las dos ecuaciones anteriores se puede ver como se calculan la capa intermedia A_{lk} (la k representa el número de neuronas en la capa l) y la capa de salida O_l , a la hora de crear el modelo se puede elegir cuantas capas intermedias se quieren, también es posible combinar estas capas recurrentes con las capas feedforward del apartado **5.1** o con las capas convolucionales del apartado anterior. El proceso de entrenamiento de estas redes se realiza con un algoritmo llamado **Propagación hacía detrás a través del tiempo**.

5.4. Modelo basado en Redes Neuronales

En este apartado se mostrará que Red Neuronal se ha implementado para resolver este problema, la implementación ha sido realizada con la librería **Keras**, como se ha comentado anteriormente en la era del Deep Learning los modelos con mayor popularidad han sido las llamadas *redes neuronales end-to-end*, se llaman así porque resuelven un problema de principio a fin, en el caso del conjunto de datos que se ha usado un ejemplo sería una red neuronal a la que se le introdujesen las imágenes sin ningún preprocesamiento (aunque indicándole a la red la estructura temporal a la hora crear el modelo) y como salida predijese una clase para un secuencia de fotogramas. La red neuronal que se ha implementado no es completamente *end-to-end*, sino que se le introducen las imágenes de la zona de los labios recortados como datos de entrenamiento y la red se encarga de extraer las características y clasificar (teniendo en cuenta la naturaleza secuencial), esto se ha hecho porque el tamaño del conjunto de datos no era suficientemente grande, darle más tareas a la red implica que más datos son necesarios para que sea capaz de atrapar los patrones en los datos, es decir, si se introdujese la imagen sin procesamiento, sería la red neuronal la que tendría que a través de intentar reducir la función de pérdida “llegar a la conclusión” que la información relevante se encuentra en la zona de los labios.

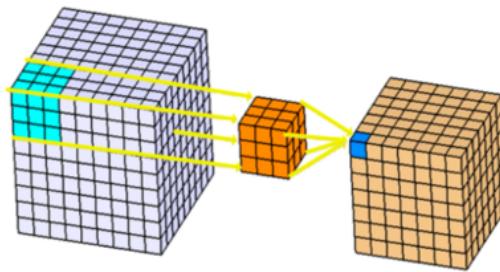


Figura 23: Convolución en 3 dimensiones

La arquitectura elegida hace uso de Redes Convolucionales 3D (en la Figura 23 se puede observar una convolución en 3 dimensiones), el motivo por el que su usa este tipo de neurona es para recoger patrones temporales en los datos, antes de explicar por qué estas redes neuronales permiten extraer patrones espacio-temporales, se comentará la estructura

de datos que se introduce en el modelo. Cada observación tiene 3 dimensiones, la primera son el número de fotogramas que forman una palabra, la segunda el ancho del fotograma y la tercera el largo (si se trabajase con imágenes en color habría una cuarta dimensión para los componentes RGB, pero se podría seguir aplicando convoluciones 3D porque el algoritmo aplica una convolución por cada canal de color). En el caso de este trabajo se ha elegido un *array* de 5 dimensiones donde la primera son el número de observaciones (1200 para entrenamiento, 100 validación y 200 para test), la segunda son el número de fotogramas (normalizado a 22), la tercera el largo del fotograma (27), la cuarta el ancho (13) y la quinta los canales de color (en este caso 1 porque las imágenes son en blanco y negro), esto forma unos *arrays* de dimensiones (1100, 22, 27, 13, 1), (100, 22, 27, 13, 1) y (200, 22, 27, 13, 1) para entrenamiento, validación y test. Como se ve en la Figura 23 las convoluciones se aplican con un filtro de dimensiones (3, 3, 3) esto es lo que permite extraer patrones espacio-temporales de los datos, porque un solo filtro está recogiendo información de todas las imágenes que forman la secuencia.

El otro tipo de capa utilizada es la llamada *LSTM*, es un tipo de red recurrente que soluciona un problema de las redes recurrentes del apartado **5.3**, el problema de memoria a corto plazo y es que estas redes van perdiendo la información del inicio de la secuencia cuando avanzando en ella, para solucionar este problema las *LSTM* añaden unas celdas de memoria a largo plazo.

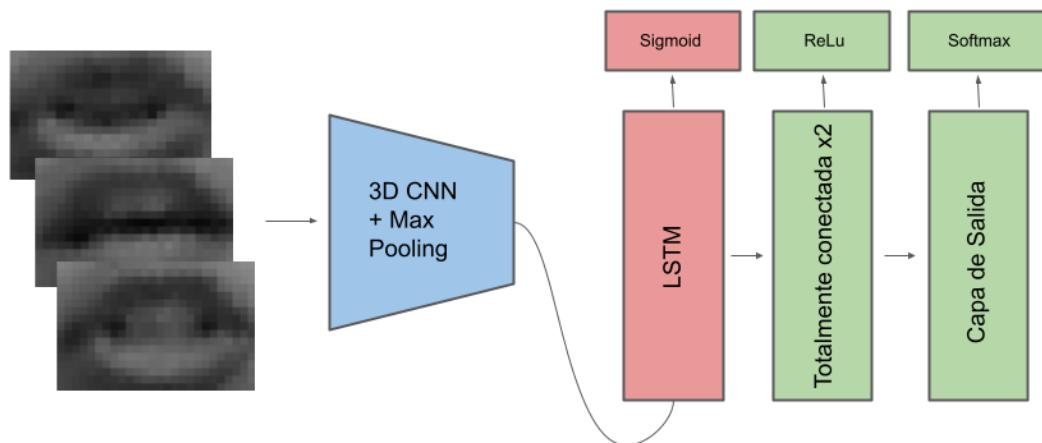


Figura 24: Arquitectura usada

A continuación se define la arquitectura utilizada (Figura 24), la red neuronal está compuesta por una capa Convolutional 3D, con 16 filtros, de dimensiones (3, 3, 3), se ha aplicado regularización sobre los pesos para evitar el sobreajuste, después se aplica una capa de *MaxPooling3D*. Esta sería la parte convolucional, la salida de la capa de *MaxPooling* se introduce en una capa *LSTM* con 32 neuronas, a continuación se encuentran dos capas totalmente conectadas con 2048 y 1024 neuronas y la capa de salida con 10 neuronas cada una representando una palabra. En total el modelo tiene 3 240 522 parámetros. El WRR para ambos conjuntos de datos y las matrices de confusión se encuentran a continuación.

	Palabras	Frases
Entrenamiento	86 %	71 %
Validación	40 %	42 %
Test	45 %	30 %

Cuadro 8: WRR de una Red Neuronal

Palabra Correcta	Palabra Predicha									
	Begin	Choose	Connection	Navigation	Next	Previous	Start	Stop	Hello	Web
Begin	2	0	0	0	9	1	3	0	0	5
Choose	0	15	2	0	0	0	0	0	0	3
Connection	0	1	13	0	0	0	0	0	0	6
Navigation	0	6	2	0	0	7	1	3	1	0
Next	1	0	0	0	9	2	1	0	0	7
Previous	0	0	0	1	0	12	2	2	3	0
Start	0	0	1	1	0	0	13	0	1	4
Stop	0	2	6	0	0	0	0	3	0	9
Hello	0	0	0	0	0	1	8	2	9	0
Web	0	3	3	0	0	0	0	0	0	14

Cuadro 9: Matriz de Confusión para palabras, Red Neuronal, conjunto test

Frases Correcta	Frase Predicha									
	Stop Navigation	Excuse me	I am sorry	Thank you	Good bye	I love this game	Nice to meet you	You are welcome	How are you?	Have a good time
Stop Navigation	6	0	0	0	0	0	11	1	2	
Excuse me	6	1	0	1	0	2	1	1	5	3
I am sorry	2	0	1	7	0	0	1	1	3	5
Thank you	0	0	0	7	0	0	6	0	2	5
Good bye	0	0	0	0	0	0	0	12	8	0
I love this game	5	1	0	0	0	1	0	3	10	0
Nice to meet you	0	0	0	0	0	0	10	0	10	0
You are welcome	0	0	0	0	0	0	0	14	6	0
How are you?	0	0	0	0	0	0	0	0	20	0
Have a good time	0	0	0	0	0	0	0	4	16	0

Cuadro 10: Matriz de Confusión para frases, Red Neuronal, conjunto test

Mirando solamente los resultados del Cuadro 8 se ve que los modelos se han ajustado bien a los datos de entrenamiento en ambos casos, sin embargo, se observa una disminución del rendimiento cuando se miran los resultados en validación y en test, pero no son cifras malas dada la complejidad de la tarea. Mirando la matriz de confusión para palabras, se observa que como ocurría con el Modelo Oculto de Markov un número considerable de observaciones se han predicho como la palabra *web*, aunque en menor medida. Mirando la matriz de confusión para frases, se puede ver que las frases *You are welcome* y *How are you?* han sido predichas en mayor proporción que el resto. Finalmente, se puede decir que el modelo ha tenido un rendimiento aceptable, el entrenado con palabras mejor que el de frases.

6. Conclusiones

Este trabajo empezó con el objetivo de entrenar un modelo basado en redes neuronales en el conjunto de datos *MIRACL – VC₁*, cuando se fueron revisando las técnicas que se utilizaban en el campo del Reconocimiento del Lenguaje se observó que este problema estaba planteado desde hace 3 décadas y había un gran número de aproximaciones posibles, por ello se optó por implementar 2 modelos siguiendo una estructura tradicional y un modelo basado en redes neuronales, este apartado pretende reflejar las conclusiones que han sido extraídas tras la elaboración del trabajo. En primer lugar, se evaluarán cada uno de los bloques que forma la tarea de lectura de labios y el rendimiento de los clasificadores, antes de hacerlo un punto importante a tener en cuenta es que si uno de estos bloques no funciona correctamente repercutirá en los bloques siguientes, por ejemplo, si la detección y recorte de la zona de los labios no se realiza de manera correcta la extracción de características no será representativa de las imágenes originales y, por lo tanto, la clasificación no será precisa.

El bloque de localización y recorte de los labios no presentaba el tema de estudio del trabajo, sin embargo, se cree que si se hubiese mejorado el rendimiento de este bloque, mejorando el código utilizado, se hubiese conseguido una extracción de características de mayor calidad, lamentablemente no se ha podido invertir más tiempo en este apartado, en la Figura 14 se puede ver como los recortes de la zona de los labios cuentan con poca resolución, otra opción podría haber sido buscar un conjunto de datos con imágenes de mayor calidad.

Con respecto al bloque de extracción de características, destacar que solo un algoritmo ha sido utilizado, se escogió el Análisis de Componentes Principales porque es una técnica familiar para el autor y ha sido utilizado en la literatura a lo largo de los años en el campo de la Visión Artificial, sin embargo, no es el extractor de características más usado y se podrían haber mejorado los resultados experimentando con otros extractores de características como los basados en la geometría de la boca u otras arquitecturas Deep Learning como los *Autoencoders*.

Finalmente, con respecto al apartado de la clasificación, los modelos utilizados han sido los adecuados de acuerdo con la literatura, aunque se cree que se podría haber obtenido un mejor resultado si se hubiese hecho un mejor trabajo en los bloques de recorte de los labios y en la extracción de características. Con respecto al modelo basado en redes neuronales, se le podría haber sacado mayor partido si se hubiese tenido mayor conocimiento sobre el ajuste de hiperparámetros, ya que es una parte importante a la hora de definir una red neuronal. Por último si se hubiese dispuesto de una conjunto de datos de mayor tamaño, es probable que se hubiese obtenido un mejor resultado, ya que las redes neuronales funcionan mejor con una gran cantidad de datos.

Los resultados obtenidos con respecto al WRR, no han sido los esperados antes de iniciar el trabajo, no solo porque el sistema tiene un rendimiento por debajo de lo esperado, sino porque las técnicas han funcionado de manera diferente a la esperada, por ejemplo se esperaba que el Modelo Oculto de Markov funcionase mejor que la Máquina de Soporte Vectorial, ya que, el segundo modelo no tiene en cuenta la naturaleza secuencial de los datos. Esto puede ser debido a que en la literatura se ha trabajado generalmente con Modelos Ocultos de Markov y fonemas, en este trabajo se ha trabajado con palabras. Con respecto al rendimiento esperado de las redes neuronales, dado que no había una gran cantidad de datos no se sabía como iba a funcionar este modelo debido al tamaño limitado de los datos, sin embargo, se puede decir que ha tenido un comportamiento correcto dada la complejidad de la tarea. De manera general, el modelo creado no ha cumplido las

expectativas iniciales, sin embargo, se ha visto que los modelos entrenados tiene un poder discriminativo significante. Si hubiese que escoger uno, se escogería el modelo basado en redes neuronales, ya que parece que tiene menos sobreajuste que la Máquina de Soporte Vectorial y se cree que se puede mejorar su rendimiento aumentando la cantidad de datos y mejorando la arquitectura usada.

6.1. Trabajo Futuro

A lo largo de este trabajo se ha comprobado que el reconocimiento del habla a partir de vídeo es un campo extenso y complejo, con un gran abanico de técnicas posibles que se pueden aplicar a cada uno de los distintos bloques que forman un sistema de reconocimiento del habla, por ello se han recogido las siguientes implementaciones que harían tener un conocimiento más extenso sobre este campo:

- Aprovechar datos de profundidad: el conjunto de datos MIRACL-VC1 utilizado (Figura 26) cuenta con datos de profundidad, además de las imágenes RGB que se han utilizado en este trabajo(Rekik y cols., 2014), este conjunto de datos fue creado con el objetivo de examinar como se pueden usar este tipo de datos con el objetivo de mejorar el rendimiento de los sistemas automáticos de lecturas de labios, el añadir la profundidad permite realizar una reconstrucción en tres dimensiones del escenario.
- Predicción a nivel de fonema o visema: en este trabajo se han utilizado como las clases a predecir las palabras o frases, otra de las aproximaciones que se han visto en la literatura es la de usar fonemas o visemas y a partir de estos construir palabras con los llamados n-gramas, de esta manera la palabra 'hola' estaría formada por los fonemas /o/ /l/ /a/, una de las ventajas de trabajar con fonemas es que permite formar otras palabras que no se encontraban en los datos originales, por el contrario, si se trabaja con palabras solo se podrá predecir esas palabras.
- Trabajar con conjuntos de datos realistas: la posición en la que los hablantes han sido grabados y el tamaño del vocabulario es poco realista, podría ser más interesante experimentar con más vocabulario y un entorno más realista.
- Conjuntos de datos en español: los datos que se han utilizado estaban en inglés, podría ser interesante el usar datos en español como los de RTVE (Fernandez-Lopez y Sukno, 2018).
- Explorar otras técnicas de extracción de características: hay un amplio abanico de diferentes técnicas que no han sido utilizadas y sería interesante hacerlo como las características basadas en el movimiento, en características geométricas o una combinación de las distintas técnicas.
- Alguno de los algoritmos utilizados solo se ha consultado de manera superficial, podría ser interesante estudiarlos más a fondo desde un punto de visto teórico, esto podría ayudar a implementar mejor las soluciones en un lenguaje de programación.
- Experimentación con audio: en este trabajo se ha limitado a trabajar con vídeo, sin embargo, podría ser interesante ver como se complementan estas dos tipos de señales.

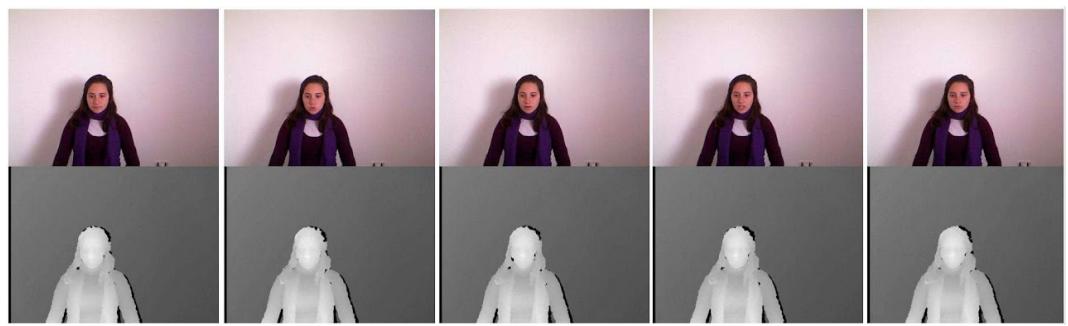


Figura 25: Demostración del conjunto de datos incluyendo profundidad (Rekik y cols., 2014)

Referencias

- Chung, J. S., y Zisserman, A. (2017, 03). Lip reading in the wild.. Descargado de https://www.robots.ox.ac.uk/~vgg/data/lip_reading/lrw1.html
- Cox, S., Harvey, R., y Lan, Y. (2008, 01). The challenge of multispeaker lip-reading..
- Dalal, N., y Triggs, B. (2005). Histograms of oriented gradients for human detection. En *2005 ieee computer society conference on computer vision and pattern recognition (cvpr'05)* (Vol. 1, p. 886-893 vol. 1).
- Delac, K., Grgic, M., y Liatsis, P. (2005, 07). Appearance-based statistical methods for face recognition. En (p. 151-158).
- Fernandez-Lopez, A., Martínez, O., y Sukno, F. M. (2017). Towards estimating the upper bound of visual-speech recognition: The visual lip-reading feasibility database. Descargado de <http://arxiv.org/abs/1704.08028>
- Fernandez-Lopez, A., y Sukno, F. M. (2018). Survey on automatic lip-reading in the era of deep learning. *Image and Vision Computing*, 78, 53-72.
- Fulton, W. (s.f.). *What's a pixel?* Descargado de <https://www.scantips.com/basics1b.html>
- Gales, M., y Young, S. (2008). *Application of hidden markov models in speech recognition*.
- Georgia Tech. (2014). *Introduction to computer vision*. Descargado de <https://www.udacity.com/course/introduction-to-computer-vision--ud810>
- Geron, A. (2019). *Hands-on machine learning with scikit-learn, keras and tensorflow*. O'Reilly Media.
- Gurjar, H. S. (2016). Descargado de https://unsplash.com/es/fotos/iSi02D_Qx_w
- Hinton, G. E., Osindero, S., y Teh, Y. W. (2006). A fast learning algorithm for deep belief nets. *Neural Computation*, 18, 1527-1554.
- Howell, D., Cox, S., y Theobald, B. (2016, 04). Visual units and confusion modelling for automatic lip-reading. *Image and Vision Computing*, 51.
- James, G., Witten, D., Hastie, T., y Tibshirani, R. (2013). *An introduction to statistical learning: with applications in r*. Springer.
- Jurafsky, D., y Martin, J. H. (2009). *Speech and language processing : an introduction to natural language processing, computational linguistics, and speech recognition*. Upper Saddle River, N.J.: Pearson Prentice Hall.
- learnopencv. (2016). *Histogram of oriented gradients*. <https://learnopencv.com/histogram-of-oriented-gradients/>.
- Liopa. (2022). *Technology - liopa*. Descargado de <https://liopa.ai/technology/> (Accessed: 2022-05-24)
- Lleida, E., Ortega, A., Miguel, A., Bazán-Gil, V., Pérez, C., Gómez, M., y de Prada, A. (2020). Rtve2020 database description. Descargado de <https://catedrartve.unizar.es/rtvedatabase.html>
- Matthews, I., Cootes, T., Bangham, J., Cox, S., y Harvey, R. (2002). Extraction of visual features for lipreading. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(2), 198-213.
- NG, A. (2012). *Machine learning*. Descargado de <https://www.coursera.org/learn/machine-learning>
- O'Halloran, B. (2021). Liopa gets a yes for its speech-recognition software. *The Irish Times*. Descargado 2021-05-24, de <https://www.irishtimes.com/business/technology/liopa-gets-a-yes-for-its-speech-recognition-software-1.4702899>

- Papandreou, G., Katsamanis, A., Pitsikalis, V., y Maragos, P. (2009). Adaptive multi-modal fusion by uncertainty compensation with application to audiovisual speech recognition. *IEEE Transactions on Audio, Speech, and Language Processing*, 17(3), 423-435.
- Prajwal, K. R., Mukhopadhyay, R., Namboodiri, V., y Jawahar, C. V. (2020). Learning individual speaking styles for accurate lip to speech synthesis. Descargado de <https://arxiv.org/abs/2005.08209>
- Rekik, A., Ben-Hamadou, A., y Mahdi, W. (2014). A new visual speech recognition approach for RGB-D cameras. En *Image analysis and recognition - 11th international conference* (pp. 21–28).
- Saavedra, R. (2022). *Lectura de labios, Trabajo Final de Grado, repositorio de GitHub*. Descargado de <https://github.com/robertosaavedr/lip-reading>
- Stigler, S. M. (1986). The history of statistics: The measurement of uncertainty before 1900. En (cap. 1). Harvard University Press.