



UNIVERSIDAD AUSTRAL

TESIS DE MAESTRÍA

**Clasificación de los dígitos escritos
en los telegramas de las elecciones
legislativas en Santa Fe mediante
técnicas de adaptación de dominio.**

Autor:
Franco LIANZA

Supervisor:
Dr. Leandro BUGNON

6 de septiembre de 2022

UNIVERSIDAD AUSTRAL

Resumen

Facultad de Ingeniería

Magister en Explotación de Datos y Gestión del Conocimiento

Clasificación de los dígitos escritos en los telegramas de las elecciones legislativas en Santa Fe mediante técnicas de adaptación de dominio.

by Franco LIANZA

The Thesis Abstract is written here (and usually kept to just this page). The page is kept centered vertically so can expand into the blank space above the title too...

Reconocimientos

The acknowledgments and the people to thank go here, don't forget to include your project advisor...

Índice general

Resumen	III
Reconocimientos	v
1. Introducción	1
1.1. Contexto	1
1.2. Motivación	2
1.3. Objetivos	3
2. Marco teórico	5
2.1. Reconocimiento de dígitos	5
2.2. Redes Neuronales	7
2.3. Adaptación de Dominio	8
3. Metodología	11
3.1. Descripción de los datos	11
3.2. Extracción de dígitos de los telegramas	11
3.2.1. Enderezado	11
3.2.2. Extracción de la grilla de votos	12
3.2.3. Detección de líneas de la grilla de votos	12
3.2.4. Segmentación de dígitos	14
3.3. Análisis del dataset	14
3.4. Diseño experimental	15
3.5. Métricas de evaluación	15
3.5.1. Accuracy	15
3.5.2. F_1	16
3.5.3. Intersección sobre unión	16
3.5.4. Distancia \mathcal{A}	17
4. Análisis de resultados	19
4.1. Comparación de espacios latentes	19
4.2. Comparación de métricas	19
4.3. Análisis de errores	19
5. Conclusiones	21
5.1. Conclusiones	21
5.2. Trabajos Futuros	21
A. Anexo: Telegramas	23
A.1. Ejemplo de telegrama	24

Índice de figuras

1.1.	Proceso eleccionario. TODO: cambiar para que se pueda leer o eliminar directament?	2
2.1.	Ejemplos del dataset MNIST	5
2.2.	Arquitectura de la red LeNet-5	6
2.3.	Estructura de pre-entrenar y fine tuning.	8
3.1.	Enderezamiento de un telegrama utilizando OpenCV.	12
3.2.	Grilla de votos extraída buscando el contorno de mayor área.	12
3.3.	Proyecciones de los ejes de la grilla. En rojo se marca el umbral de corte.	13
3.4.	Grilla detectada (en verde) utilizando el umbral por sobre las proyecciones y su posterior agrupamiento con clustering jerár- quico.	13
3.5.	Primer registro extraído de la grilla de votos.	14
3.6.	Dígitos detectados en el primer bloque de votos.	14
3.7.	algo	15
3.8.	Ejemplos de distribuciones de dominios.	18

Índice de cuadros

2.1. Precisión obtenida al entrenar una LeNet-5 con distintos datasets de dígitos.	6
3.1. Ejemplo de registros del dataset.	14

Capítulo 1

Introducción

La intro la podes pensar en 3 subsecciones: descripción de elecciones y contabilización de las actas hoy, la oportunidad que se presenta con la digitalización (y los desafíos que aparecen) y luego los objetivos. Tiene que quedar claro que problema ves y qué propones hacer a grandes rasgos. También es importante que estos objetivos se vean reflejados en los resultados y conclusiones (es decir, prometer algo que después cumplis al final del doc)

1.1. Contexto

En Argentina se celebran elecciones cada 2 años a excepción de las presidenciales que se realizan cada 4 años. Existen, principalmente, tres tipos de elecciones:

- Elecciones nacionales, para elegir a las autoridades federales del país: el Poder Ejecutivo, constituido por el Presidente y el vicepresidente y el Congreso Nacional, formado por Senadores y Diputados.
- Elecciones provinciales y de la Ciudad de Buenos Aires o locales, para elegir a las autoridades de cada provincia: los poderes ejecutivos de las provincias y sus legislaturas.
- Elecciones municipales, regidas por las leyes y procedimientos de cada provincia.

Si bien emitir el sufragio es diferente en cada una de ellas, generalmente consta de ingresar a un cuarto oscuro, elegir el candidato que se desea y depositar el voto en una urna. Al finalizar la jornada, las autoridades de mesas recuentan los votos y llenan una planilla a mano alzada donde se resume la cantidad de votos obtenidos por cada candidato o partido político. Dicha planilla es escaneada y enviada a través de un telegrama del correo argentino al centro de cómputo para su procesamiento. Una vez allí, se contabilizan en un sistema informático a partir de un grupo de personas. Este proceso cuenta con una etapa de digitalización y otra de validación (TODO: esto lo vi en el diagrama de las elecciones de cordoba. no encontre nada oficial al respecto.).



FIGURA 1.1: Proceso eleccionario. TODO: cambiar para que se pueda leer o eliminar directamente?

Para que el proceso sea lo mas rápido y eficaz posible, se requiere a una gran cantidad personas destinadas al centro de cómputo. Tal solución hace que el proceso sea altamente ineficiente en cuanto a tiempos y costos se refiere. En las elecciones legislativas del 2021 se gastaron unos \$17.000 millones de pesos de los cuales \$4.000 millones de pesos fueron destinados a sueldos para el personal¹.

1.2. Motivación

Digitalizar los telegramas de manera automática supondrá un ahorro considerable en el presupuesto de las elecciones, agilizará la obtención de los resultados y aportará transparencia al proceso en general. Es posible entrenar un modelo de clasificación de dígitos a un costo extremadamente menor al actual y utilizarlo al momento de la contabilización de los votos.

Contar con una digitalización automática permitirá bajar los costos debido a que se necesitará un grupo menor de personas en el centro de cómputo. Además, el trabajo a realizar será mas simple ya que sólo constará del proceso de validación.

La clasificación de números es un problema que, si bien parece resuelto con LeCun et al., 1998 y la creación del dataset MNIST, no debe ser tomada a la ligera. No existe una única forma de escribir y año a año cambian las personas que son los jefes de mesa encargados de completar los telegramas. Las características de los números escritos a mano difiere entre cada elección. Cuando la distribución de los datos de entrenamiento difiere a la de los datos de aplicación, se está ante un *corrimiento de dominio* (*domain shift* o *data drift* en inglés). Esto implica que un modelo que se entrene en algún dataset estático como el MNIST que fue creado en el 1998, hará malas clasificaciones los dígitos de las elecciones.

Cuando los dominios de entrenamiento (origen) y aplicación (destino) son distintos, se debe a que hay un *sesgo* en los datos. Las técnicas de entrenamiento clásicas suponen que, si bien existe el sesgo existe, éste será el mismo entre origen y destino. Cuando el supuesto no se cumple, se debe recurrir a técnicas de entrenamiento más complejas donde se intenta que el

¹Fuente: [El cronista](#)

modelo aprenda a adaptar un dominio a otro. Hoy en día no hay trabajos relacionados a la digitalización de telegramas en Argentina que compare cuál es la mejor técnica de adaptación de dominio.

1.3. Objetivos

La presente tesis enfocará en el desarrollo de un modelo que permita digitalizar los telegramas de las elecciones en Argentina utilizando las legislativas de la provincia de Santa Fe del año 2021 como comparativa. Los telegramas son públicos y se encuentran subidos en la [página oficial del estado argentino](#). En el anexo A se adjunta un ejemplo de uno de ellos.

Para poder llevarlo a cabo, se procede a:

- Extraer los números escritos a mano en los telegramas.
- Evaluar técnicas de adaptación de dominio para entrenar modelos destinados a la clasificación de números de los telegramas.
- Analizar los espacios latentes de los modelos con el fin de obtener *insights* respecto a la capacidad de generalización de los modelos.

Capítulo 2

Marco teórico

2.1. Reconocimiento de dígitos

La clasificación de dígitos es un problema resuelto desde 1998. En uno de los primeros exponentes de lo que luego se denominó *deep learning*, LeCun et al., 1998 se crea un nuevo dataset de dígitos modificando el existente NIST y se propone la red neuronal LeNet-5 para la clasificación de los mismos. El dataset MNIST consiste de 60.000 imágenes de entrenamiento y 10.000 de testing. Cada una de las imágenes es de un tamaño de 28x28 pixeles.



FIGURA 2.1: Ejemplos del dataset MNIST

La red posee capas convolucionales las cuales se encargan de extraer características o patrones de las imágenes para luego ser procesadas por capas densas que se encargan de la clasificación.

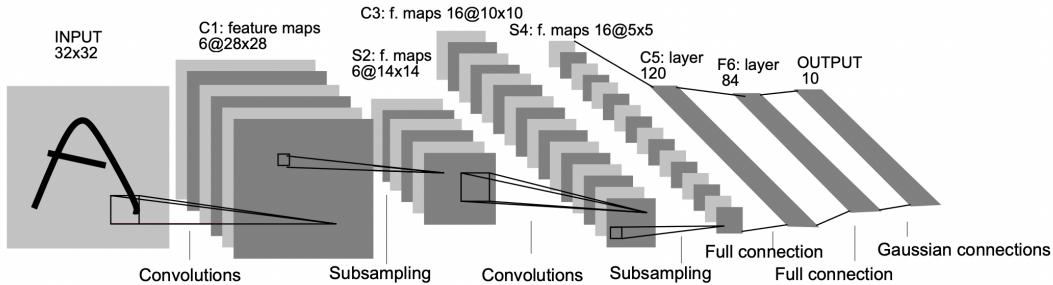


FIGURA 2.2: Arquitectura de la red LeNet-5

Sin embargo, aunque LeNet-5 (o cualquier otro modelo) presente buenas métricas de performance, no significa que pueda ser aplicado a otro dataset de dígitos. El cuadro 2.1 muestra la precisión que se obtiene al entrenar una red LeNet-5 en distintos datasets de dígitos: MNIST (LeCun et al., 1998), USPS (Hull, 1994) y SVHN (Netzer et al., 2011). La arquitectura de la red posee la capacidad de aprender los patrones de cada uno de los ellos pero no logra generalizar a otros. Si bien cada uno de ellos muestran los mismos números, lo hacen de forma diferente. Esto se debe al *sesgo* que existe en los datos. Cuando se entrena un modelo se aprende a reconocer características propias del dataset que le permite resolver el problema, incluyendo el sesgo de los mismos. Esto hace que un problema "sencillo" de clasificación de imágenes de 28x28 pixeles no sea trivial. Todos los datasets se encuentran sesgados de alguna forma y es imposible armarlos de tal forma que no presenten algún nivel de sesgo (Khosla et al., 2012). Cuando los datos con los que se quiere evaluar un modelo provienen de un dominio o distribución diferente al de entrenamiento, se está ante un *dataset shift* (Quinonero-Candela et al., 2008). Es decir, se le muestran datos al modelo con características que nunca vió, provocando predicciones incorrectas.

	MNIST	USPS	SVHN
Entrenamiento			
MNIST	99.17 %	78.08 %	31.50 %
USPS	57.10 %	95.42 %	26.94 %
SVHN	61.92 %	64.28 %	89.52 %

CUADRO 2.1: Precisión obtenida al entrenar una LeNet-5 con distintos datasets de dígitos.

El problema que abordará la presente tesis es cómo se puede entrenar un modelo que aprenda a clasificar dígitos para ser utilizados en los telegramas

de elecciones legislativas de Santa Fe del 2021 (distinto al dominio de entrenamiento) pese al sesgo existente en los datasets mencionados previamente. Para poder lograrlo, se necesita entrenar de cierta manera un modelo en alguno de los datasets públicos etiquetados de forma tal que pueda generalizar a otro similar.

2.2. Redes Neuronales

Cuando se habla de *Deep learning*, se hace referencia a una serie de algoritmos de *machine learning* que son capaces de utilizar múltiples capas de procesamiento de forma que puedan aprender representaciones de los datos con diferentes niveles de abstracción (LeCun, Bengio e Hinton, 2015). Estos algoritmos, denominados redes neuronales profundas (o DNNs por sus siglas en inglés), poseen la capacidad de encontrar variables que expliquen la naturaleza del comportamiento de los datos.

(TODO: agregar algún grafico de una red donde se muestre que las capas van extrayendo features)

Los modelos obtenidos a partir del *deep learning* han demostrado tener gran capacidad de aprendizaje para todo tipo de problemas, como ser *computer vision* (Szeliski, 2010; Redmon et al., 2016), procesamiento del lenguaje natural (Devlin et al., 2018), reconocimiento del habla (Hannun et al., 2014), juegos (Silver et al., 2016), generación de imágenes a partir de descripciones (Ramesh et al., 2022), entre otros.

Aunque la utilidad de estos modelos se encuentra demostrada y día a día son utilizados en diferentes ámbitos de la vida, presentan un gran problema: la enorme cantidad de datos etiquetados que requieren para su entrenamiento. La mayoría de los modelos que mejores métricas de performance presentan, necesitan millones de datos en sus datasets de entrenamiento. Esto implica que, para que los mismos sean de utilidad, resultan de suma importancia los procesos de recolección y etiquetado de los datos. La eficacia de los modelos queda altamente relacionada con la calidad de los datos que se posean o se logren conseguir. El etiquetado de los datos es una tarea costosa, ineficiente y hasta a veces resulta inviable de realizar (Reis, 2022). Año a año los telegramas de las elecciones son completados a mano por distintas personas, por lo que resulta imposible contar con un dataset lo suficientemente general como para ser utilizado en la clasificación de los dígitos.

Una posible solución a este problema consiste en emular la capacidad que tienen los humanos de adquirir conocimiento relevante en un área y aplicarlo en otra similar (Thrun y Pratt, 1998). Es decir, poder *transferir* lo aprendido. En el caso del *deep learning*, lo que se busca es que la red aprenda representaciones lo suficientemente generales para que puedan ser utilizados en el entrenamiento de una tarea similar. Esto implica que se puede contar con una red pre-entrenada y luego, continuar entrenándola con los datos propios del problema particular que se quiere resolver. A esto último se lo denomina *fine tuning*. Lo que se busca es acortar tiempos de entrenamiento, reducir la cantidad de datos necesarios y construir modelos más robustos.

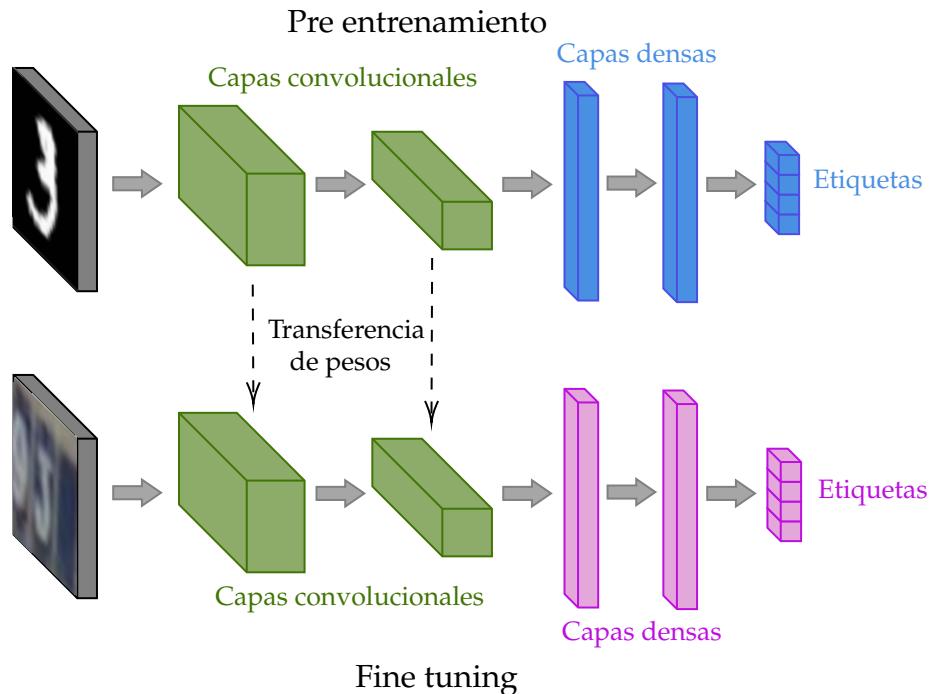


FIGURA 2.3: Estructura de pre-entrenar y fine tuning.

El proceso de pre-entrenar y *fine tuning* ha mejorado considerablemente los resultados obtenidos en diversos problemas del estado del arte, incluso las redes pre-entrenadas pueden ser fácilmente adaptadas a otras tareas con pocos datos etiquetados. No obstante, en muchos escenarios no se cuenta con datos etiquetados, imposibilitando el *fine tuning*. De aquí es que surje la necesidad de transferir el aprendizaje obtenido en un dominio de origen con datos etiquetados a otro de destino donde no se poseen etiquetas (Ben-David et al., 2006). Los modelos de aprendizaje profundo se ven afectados negativamente cuando existe un *dataset shift* como se mostró previamente en el cuadro 2.1. Por lo tanto, la *adaptación de dominio* aparece como una solución cuando la distribución de los datos de entrenamiento difiere de los datos de testing. La idea que persigue es reducir la diferencia que existe entre las distribuciones de origen y destino.

La tesis abordará el uso de diferentes técnicas de *adaptación de dominio* para entrenar un modelo capaz de transferir el conocimiento del dominio del MNIST a los dígitos escritos en los telegramas de las elecciones.

2.3. Adaptación de Dominio

Loreum ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetuer id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis

in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.

Capítulo 3

Metodología

En este capítulo se detallarán los procesos abordados para la extracción y clasificación de los dígitos escritos en los telegramas de las elecciones. Primero se describirá el proceso de obtención, limpieza y transformación de los datos a fin de obtener un dataset con el cual entrenar. Luego, se describirán los procesos a ser llevados a cabo para el diseño experimental de los modelos.

3.1. Descripción de los datos

Los telegramas fueron descargados desde la [página oficial del estado argentino](#). Presentan un formato estándar en forma de grilla donde en cada renglón se encuentra el partido político y los votos obtenidos para diputados y senadores. En la página también se puede descargar un CSV donde se encuentra el *id* de cada telegrama y los valores digitalizados oficialmente.

3.2. Extracción de dígitos de los telegramas

Los telegramas de las elecciones de Santa Fe presentan un formato tabular, donde cada fila representa un partido político y los votos obtenidos abiertos por senadores y diputados. Se debieron ejecutar múltiples pasos de extracción, transformación y carga (ETL por sus siglas) de los mismos antes de poder armar un dataset con el cual entrenar los modelos. Se utilizó la librería OpenCV (Bradski, 2000) para manipular las imágenes y poder llevar a cabo el proceso de ETL.

3.2.1. Enderezado

Como los telegramas son escaneados a mano, el primer paso consiste en enderezarlos. Este proceso puede realizarse buscando el rectángulo de mayor área, calculando el ángulo de rotación y rotando la imagen completa con la función `getRotationMatrix2D` de OpenCV.

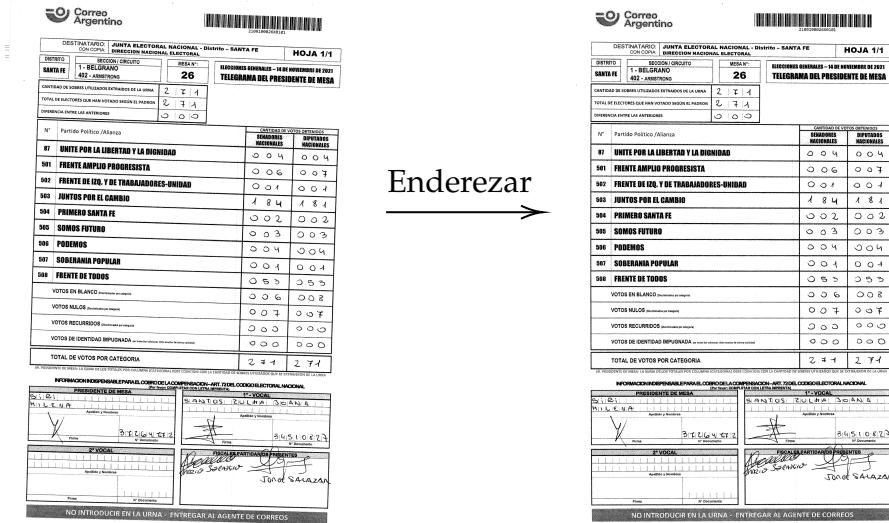


FIGURA 3.1: Enderezamiento de un telegrama utilizando OpenCV.

3.2.2. Extracción de la grilla de votos

El siguiente paso consiste en poder seleccionar la grilla de los votos y poder extraerla para seguir trabajando en ella. Esto puede realizarse utilizando la función `getContours` de OpenCV y quedándose con el contorno de mayor área.

Nº	Partido Político /Alianza	CANTIDAD DE VOTOS OBTENIDOS	
		SOCIALES NACIONALES	MIGRANTES NACIONALES
07	UNITE POR LA LIBERTAD Y LA DIGNIDAD	0 0 4	0 0 4
501	FRENTE AMPLIO PROGRESISTA	0 0 6	0 0 7
502	FRENTE DE IZQ. Y DE TRABAJADORES-UNIDAD	0 0 1	0 0 1
503	JUNTOS POR EL CAMBIO	1 8 4	1 8 4
504	PRIMERO SANTA FE	0 0 2	0 0 2
505	SOMOS FUTURO	0 0 3	0 0 3
506	PODEMOS	0 0 4	0 0 4
507	SOBERANIA POPULAR	0 0 1	0 0 1
508	FRENTE DE TODOS	0 5 5	0 5 5
VOTOS EN BLANCO (descartados por impugnación)		0 0 6	0 0 2
VOTOS NULOS (descartados por impugnación)		0 0 7	0 0 7
VOTOS RECURSIDOS (descartados por impugnación)		0 0 0	0 0 0
VOTOS DE IDENTIDAD IMPUGNADA (se votó por el voto que no aparece en la lista actual)		0 0 0	0 0 0
TOTAL DE VOTOS POR CATEGORÍA		2 4 1	2 7 4

FIGURA 3.2: Grilla de votos extraída buscando el contorno de mayor área.

3.2.3. Detección de líneas de la grilla de votos

Teniendo la grilla de votos separada del telegrama, se procede a detectar las líneas para poder extraer cada registro de la misma. Si bien OpenCV posee funciones para la detección de líneas, se optó por utilizar un enfoque simplificado. El mismo es similar al utilizado en (Lamagna, 2016). Debido a que la grilla se encuentra enderezada, se puede utilizar proyecciones de los colores sobre los ejes x e y para detectarlas. Al sumar todos los colores por eje, se pueden encontrar los picos donde se encuentran las líneas horizontales y verticales. Posteriormente, se selecciona un umbral de corte por cada

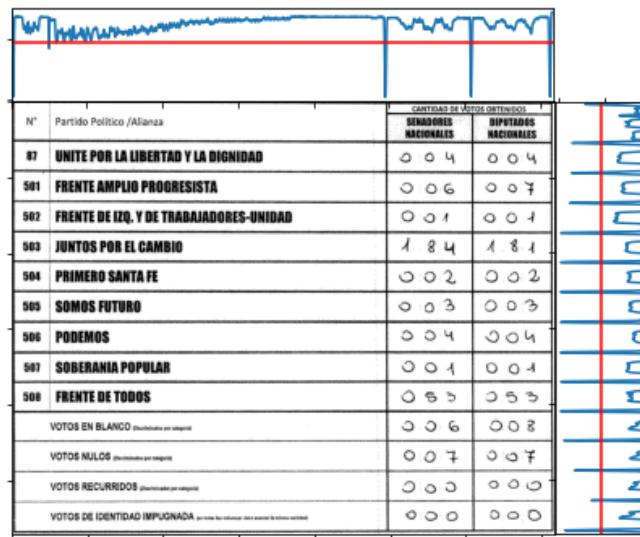


FIGURA 3.3: Proyecciones de los ejes de la grilla. En rojo se marca el umbral de corte.

eje, siendo el promedio menos dos desvíos estándar. La figura 3.3 muestra las proyecciones obtenidas con los umbrales por cada eje.

Sin embargo, aunque se seleccione aquellos píxeles que se encuentren por debajo del umbral, por cada pico existe mas de un pixel. Siguiendo con el ejemplo, en el eje x se obtienen los siguientes píxeles que se encuentran en los "picos":

```
array([    3,     4,     5,     6,     7,   100,   119,   120,   988,
       989,   990,   991,   992,  1215,  1216,  1217,  1218,  1219,
      1423,  1424,  1425,  1426,  1427])
```

Con la estrategia del umbral no se obtienen las 4 líneas que se desean detectar sobre el eje x . No obstante, se puede aplicar un clustering jerárquico sobre los índices obtenidos y luego calcular el índice promedio de cada cluster para tomarlo como único representante. El proceso se repite de la misma manera sobre el eje y . Esta última modificación sobre el trabajo realizado en (Lamagna, 2016) asegura tener un único pixel por línea.

FIGURA 3.4: Grilla detectada (en verde) utilizando el umbral por sobre las proyecciones y su posterior agrupamiento con clustering jerárquico.

Nº	Partido Político /Alianza	CANTIDAD DE VOTOS DETERMINADOS	
		SEÑADORES NACIONALES	DIPUTADOS NACIONALES
07	UNITE POR LA LIBERTAD Y LA DIGNIDAD	0 0 4	0 0 4
501	FRENTE AMPLIO PROGRESISTA	0 0 6	0 0 7
502	FRENTE DE IZQ. Y DE TRABAJADORES-UNIDAD	0 0 1	0 0 1
503	JUNTOS POR EL CAMBIO	1 8 4	1 8 4
504	PRIMERO SANTA FE	0 0 2	0 0 2
505	SOMOS FUTURO	0 0 3	0 0 3
506	PODEMOS	0 0 4	0 0 4
507	SOBERANIA POPULAR	0 0 1	0 0 4
508	FRENTE DE TODOS	0 5 5	0 5 5
	VOTOS EN BLANCO (descartados por voto nulo)	0 0 6	0 0 8
	VOTOS NULOS (descartados por voto nulo)	0 0 7	0 0 7
	VOTOS RECURRIDOS (descartados por voto nulo)	0 0 0	0 0 0
	VOTOS DE IDENTIDAD IMPUGNADA (en votos que no cumplieron con las normas establecidas)	0 0 0	0 0 0

FIGURA 3.4: Grilla detectada (en verde) utilizando el umbral por sobre las proyecciones y su posterior agrupamiento con clustering jerárquico.

3.2.4. Segmentación de dígitos

Una vez obtenida la grilla, se itera por cada registro obteniendo los rectángulos que contienen los votos, ignorando el primer renglón que contiene los títulos de la grilla.



FIGURA 3.5: Primer registro extraído de la grilla de votos.

Para separar cada dígito en una única imagen, se utiliza un análisis de componentes conectados (`connectedComponents` de OpenCV) y se recortan aquellos contornos que posean de área entre el 5 % y el 70 % de los pixeles totales de la imagen. Los valores fueron obtenidos mediante experimentación y sirven para descartar ruido que pueda detectar el análisis de componentes conectados.



FIGURA 3.6: Dígitos detectados en el primer bloque de votos.

Por último, los dígitos se guardan de forma que sean cuadrados junto a su etiqueta tomada de la digitalización oficial. El dataset será denominado como *TDS* (dataset de telegramas) a lo largo de la presente tesis.

3.3. Análisis del dataset

Con la extracción descripta en la sección anterior, se procede a hacer un análisis exploratorio de los datos en búsqueda de posibles errores. La tabla 3.1 muestra algunos registros del dataset que vincula los dígitos extraídos de los telegramas con los partidos políticos.

Telegrama	Partido	Tipo	Dígitos	Cantidad Dígitos
2100100026X	Unite	Diputados	0 0 4	3
2100100026X	Unite	Senadores	0 0 4	4

CUADRO 3.1: Ejemplo de registros del dataset.

El lector podrá darse cuenta que existen errores en el cuadro anterior. En la columna de dígitos donde se debería encontrar sólamente números existe uno incorrecto para senadores. Para limpiar estas extracciones incorrectas, se calculan dos columnas las cuales contienen la proporción mínima y máxima de pixeles blancos en las imágenes. La figura 3.7 muestra los histogramas generales de estas dos variables.

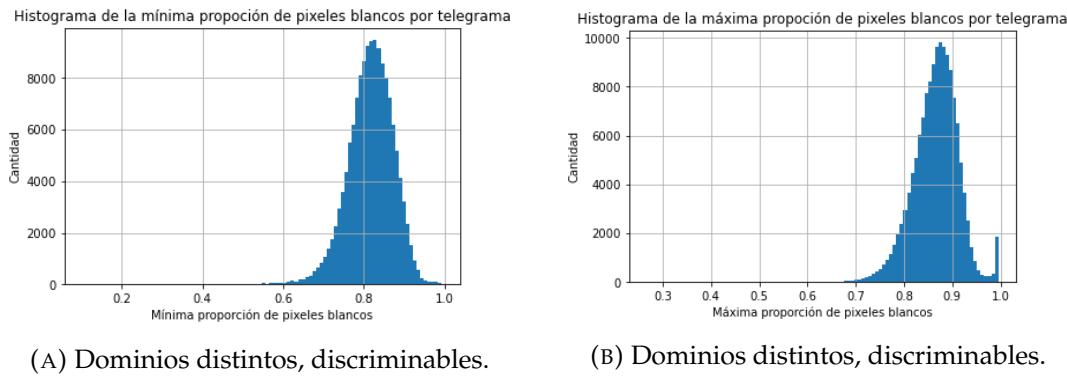


FIGURA 3.7: algo

3.4. Diseño experimental

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetuer id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.

3.5. Métricas de evaluación

Los modelos entrenados serán evaluados con distintas métricas, descriptas en las siguientes sub-secciones del capítulo. Las mismas pretenden evaluar qué tan buenos son los modelos y qué capacidad de adaptación de dominio poseen.

3.5.1. Accuracy

La métrica de *accuracy* permite identificar qué tan cerca o lejos un conjunto de observaciones se encuentra respecto a los valores reales. Es el ratio de predicciones correctas sobre las totales.

$$\text{Accuracy}(y, \hat{y}) = \frac{1}{n} \sum_{i=0}^{n-1} 1(\hat{y}_i = y_i) \quad (3.1)$$

Donde:

- n : es la cantidad de observaciones totales.

- y_i : es el valor de la i -ésima observación correspondiente al real.
- \hat{y}_i : es el valor predicho para la i -ésima observación.
- $1(x)$: es la función indicador.

3.5.2. F_1

La métrica F_1 es una media armónica de otras dos: *precisión* y *recall*. De manera simplificada, la primera muestra la capacidad del modelo de no etiquetar como positivo una obsevación que es negativa y la segunda muestra la capacidad del modelo de encontrar todas las observaciones positivas.

La *precisión* consiste en calcular el ratio de predicciones positivas correctas de la clase respecto al total de predicciones positivas de la clase.

$$\text{Precision}(y, \hat{y}) = \frac{TP}{TP + FP} \quad (3.2)$$

Donde:

- TP : es la cantidad de verdaderos positivos.
- FP : es la cantidad de falsos positivos.

Por otro lado, el *recall* consiste en calcular el ratio de predicciones positivas correctas de la clase respecto al total de predicciones correctas de la clase.

$$\text{Recall}(y, \hat{y}) = \frac{TP}{TP + FN} \quad (3.3)$$

Donde:

- TP : es la cantidad de verdaderos positivos.
- FN : es la cantidad de falsos negativos.

Las dos métricas mencionada anteriormente pueden ser combinadas en una sola denominada F_β . El valor de β permite asignar un peso distinto a la precisión o al recall dentro del promedio armónico. Cuando $\beta = 1$, ambas poseen el mismo peso.

$$F_\beta(y, \hat{y}) = (1 + \beta^2) \times \frac{\text{precision}(y, \hat{y}) \times \text{recall}(y, \hat{y})}{\beta^2 \times \text{precision}(y, \hat{y}) + \text{recall}(y, \hat{y})} \quad (3.4)$$

El rango de valores es de $[0, 1]$, donde 1 corresponde a un clasificador que funciona sin errores.

3.5.3. Intersección sobre unión

Otra forma de medir el clasificador es mediante la *intersección sobre unión* (IoU por sus siglas en inglés). Consta de calcular la cantidad de dígitos únicos predichos por sobre la cantidad de dígitos únicos reales. Por ejemplo, para un

telegrama y un partido el clasificador predice 189 votos cuando lo real es 180. Entonces, el IoU viene dado por:

$$\begin{aligned}
 IoU(\{1, 8, 0\}, \{1, 8, 9\}) &= \frac{|\{1, 8, 0\} \cap \{1, 8, 9\}|}{|\{1, 8, 0\} \cup \{1, 8, 9\}|} \\
 &= \frac{|\{1, 8\}|}{|\{1, 8, 0, 9\}|} \\
 &= \frac{2}{4} \\
 &= 0.5
 \end{aligned} \tag{3.5}$$

De forma general: sea y_i el dígito i -ésimo del voto real y de longitud n , \hat{y}_j el dígito j -ésimo del voto predicho por el modelo \hat{y} de longitud m , entonces la métrica viene dada por:

$$IoU(y, \hat{y}) = \frac{|\{y_i\} \cap \{\hat{y}_j\}|}{|\{y_i\} \cup \{\hat{y}_j\}|} \forall i \in \{1, \dots, n\}, \forall j \in \{1, \dots, m\} \tag{3.6}$$

3.5.4. Distancia \mathcal{A}

La distancia \mathcal{A} mide la similaridad entre dos distribuciones. Puede ser utilizada para analizar los espacios latentes de clasificadores en problemas de adaptación de dominio (Ben-David et al., 2006). La métrica viene dada por:

$$dist_{\mathcal{A}} = 2(1 - 2\epsilon) \tag{3.7}$$

Donde ϵ es el error en test de un clasificador entrenado para discriminar el dominio de origen del de destino. Cuando el error de clasificación es bajo, significa que hay diferencias significativas entre las dos distribuciones, haciendo que $dist_{\mathcal{A}}$ sea grande y vice versa.

En la figura 3.8 se muestran posibles casos de distribuciones de dominios. En la sub-figura 3.8a las distribuciones de dominios son significativamente diferentes entre si, por lo tanto $dist_{\mathcal{A}} \approx 2$. Por el contrario, en la sub-figura 3.8b las distribuciones de dominios son similares entre si, por lo que se espera que $dist_{\mathcal{A}} \approx 0$.

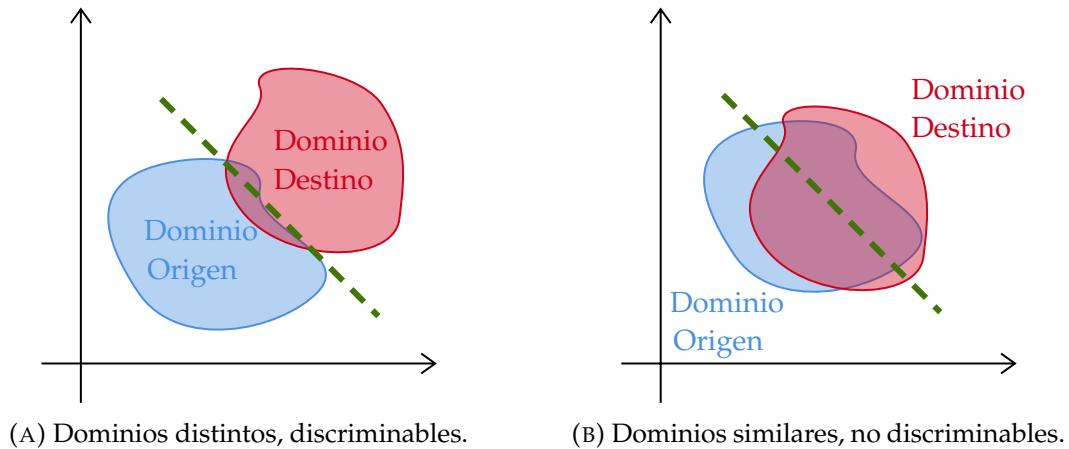


FIGURA 3.8: Ejemplos de distribuciones de dominios.

Durante el proyecto se utiliza como clasificador una red de una capa densa con una función de activación sigmoidea, es decir, una regresión logística.

Capítulo 4

Análisis de resultados

4.1. Comparación de espacios latentes

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetuer id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.

4.2. Comparación de métricas

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetuer id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.

4.3. Análisis de errores

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida

mauris. Nam arcu libero, nonummy eget, consectetuer id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.

Capítulo 5

Conclusiones

5.1. Conclusiones

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetuer id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.

5.2. Trabajos Futuros

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetuer id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.

Apéndice A

Anexo: Telegramas

A.1. Ejemplo de telegrama

DESTINATARIO: CON COPIA:			JUNTA ELECTORAL NACIONAL - Distrito - SANTA FE DIRECCION NACIONAL ELECTORAL	HOJA 1/1	
DISTRITO SANTA FE	SECCION / CIRCUITO 1 - BELGRANO 402 - ARMSTRONG	MESA N°: 1	ELECCIONES GENERALES – 14 DE NOVIEMBRE DE 2021 TELEGRAMA DEL PRESIDENTE DE MESA		
CANTIDAD DE SOBRES UTILIZADOS EXTRAIDOS DE LA URNA 2 8 3					
TOTAL DE ELECTORES QUE HAN VOTADO SEGÚN EL PADRON 2 8 3					
DIFERENCIA ENTRE LAS ANTERIORES 0 0 0					
Nº	Partido Político /Alianza	CANTIDAD DE VOTOS OBTENIDOS			
		SENADORES NACIONALES		DIPUTADOS NACIONALES	
		87	UNITE POR LA LIBERTAD Y LA DIGNIDAD		4 5
		501	FRENTE AMPLIO PROGRESISTA		15 16
		502	FRENTE DE IZQ. Y DE TRABAJADORES-UNIDAD		8 8
		503	JUNTOS POR EL CAMBIO		135 135
		504	PRIMERO SANTA FE		2 2
		505	SOMOS FUTURO		7 7
		506	PODEMOS		1 1
507	SOBERANIA POPULAR		6 6		
508	FRENTE DE TODOS		94 92		
VOTOS EN BLANCO (Discriminados por categoría)			5 5		
VOTOS NULOS (Discriminados por categoría)			6 6		
VOTOS RECURRIDOS (Discriminados por categoría)			— —		
VOTOS DE IDENTIDAD IMPUGNADA (en todas las columnas debe asentir la misma cantidad)			— —		
TOTAL DE VOTOS POR CATEGORIA			283 283		
SR. PRESIDENTE DE MESA: LA SUMA DE LOS TOTALES POR COLUMNA (CATEGORIA) DEBE COINCIDIR CON LA CANTIDAD DE SOBRES UTILIZADOS QUE SE EXTRAJERON DE LA URNA					
INFORMACION INDISPENSABLE PARA EL COBRO DE LA COMPENSACION-ART. 72 DEL CODIGO ELECTORAL NACIONAL (Por favor: COMPLETAR CON LETRA IMPRESA)					
PRESIDENTE DE MESA					
B. Rebeca Livicra		1 ^o - VOCAL			
Apellido y Nombres		Apellido y Nombres			
Firma		Nº Documento			
38087468		37228128			
2 ^o VOCAL					
Apellido y Nombres		Apellido y Nombres			
Firma		Nº Documento			
40250485		42328417			
FISCALES PARTIDARIOS PRESENTES					
NO INTRODUCIR EN LA URNA - ENTREGAR AL AGENTE DE CORREOS					

Bibliografía

- Ben-David, Shai et al. (2006). «Analysis of Representations for Domain Adaptation». En: *Advances in Neural Information Processing Systems*. Ed. por B. Schölkopf, J. Platt y T. Hoffman. Vol. 19. MIT Press. URL: <https://proceedings.neurips.cc/paper/2006/file/b1b0432ceafb0ce714426e9114852ac7-Paper.pdf>.
- Bradski, G. (2000). «The OpenCV Library». En: *Dr. Dobb's Journal of Software Tools*.
- Devlin, Jacob et al. (2018). «Bert: Pre-training of deep bidirectional transformers for language understanding». En: *arXiv preprint arXiv:1810.04805*.
- Hannun, Awni et al. (2014). «Deep speech: Scaling up end-to-end speech recognition». En: *arXiv preprint arXiv:1412.5567*.
- Hull, Jonathan J. (1994). «A database for handwritten text recognition research». En: *IEEE Transactions on pattern analysis and machine intelligence* 16.5, págs. 550-554.
- Khosla, Aditya et al. (2012). «Undoing the damage of dataset bias». En: *European Conference on Computer Vision*. Springer, págs. 158-171.
- Lamagna, Walter Marcelo (2016). «Lectura artificial de números manuscritos en datos abiertos de elecciones legislativas en la Ciudad de Buenos Aires». Tesis doct. Universidad de Buenos Aires. Facultad de Ciencias Exactas y Naturales.
- LeCun, Yann, Yoshua Bengio y Geoffrey Hinton (2015). «Deep learning». En: *nature* 521.7553, págs. 436-444.
- LeCun, Yann et al. (1998). «Gradient-based learning applied to document recognition». En: *Proceedings of the IEEE* 86.11, págs. 2278-2324.
- Netzer, Yuval et al. (2011). «Reading digits in natural images with unsupervised feature learning». En.
- Quinonero-Candela, Joaquin et al. (2008). *Dataset shift in machine learning*. Mit Press.
- Ramesh, Aditya et al. (2022). «Hierarchical text-conditional image generation with clip latents». En: *arXiv preprint arXiv:2204.06125*.
- Redmon, Joseph et al. (2016). «You only look once: Unified, real-time object detection». En: *Proceedings of the IEEE conference on computer vision and pattern recognition*, págs. 779-788.
- Reis, Pedro Miguel Lima de Sousa (2022). «Data Labeling tools for Computer Vision: a Review». En.
- Silver, David et al. (2016). «Mastering the game of Go with deep neural networks and tree search». En: *nature* 529.7587, págs. 484-489.
- Szeliski, Richard (2010). *Computer vision: algorithms and applications*. Springer Science & Business Media.
- Thrun, Sebastian y Lorien Pratt (1998). *Learning to learn*.