Universidad de Buenos Aires

Facultad de Ciencias Exactas y Naturales

Segmentación y reconocimiento de textos manuscritos utilizando redes profundas

Tesis presentada para optar al título de Magíster de la Universidad de Buenos Aires en Explotación de Datos y Descubrimiento del Conocimiento

Candidato: Juan Pablo Pilorget

Directora: María Elena Buemi

31 de mayo de 2023

Índice general

Resumen

El reconocimiento de caracteres es un campo de investigación y desarrollo aplicado que ha tenido numerosos avances en los últimos años. Las técnicas tradicionales de reconocimiento óptico de caracteres, con una alta carga de trabajo manual para lograr su correcto funcionamiento, han comenzado a ser reemplazadas por abordajes inteligentes utilizando redes neuronales profundas. Los nuevos modelos permiten no sólo automatizar los procesos de reconocimiento sino mejorar la calidad de un tipo específico: el texto escrito a mano. La cantidad creciente de documentos manuscritos digitalizados presentan a la vez un desafío, por la imposibilidad de los métodos tradicionales para reconocer adecuadamente ese tipo de texto, y una oportunidad, por el impacto que podrían generar para el acervo público el desarrollo de modelos que detecten y reconozcan correctamente caracteres escritos a mano.

El presente trabajo profundiza en las arquitecturas de redes neuronales que pueden utilizarse para detectar texto y reconocer caracteres de forma inteligente, analizando cuantitativamente el desempeño de modelos de segmentación y reconocimiento. Asimismo, evalúa cualitativamente los modelos en el acervo de la Biblioteca Digital de la Facultad de Ciencias Exactas y Naturales de la Universidad de Buenos Aires, aplicando las técnicas a la correspondencia digitalizada de Mario Bunge. La información obtenida es sistematizada y puesta a disposición para futuros trabajos y líneas de investigación.

Palabras clave: reconocimiento de caracteres; redes neuronales; segmentación semántica; texto manuscrito; aprendizaje profundo.

Abstract

Title: Handwritten text segmentation and recognition with deep learning

Character recognition is an applied research field that has gone through numerous advances in recent years. Traditional optical character recognition techniques, requiring several manual steps in order to work properly, are being gradually replaced by new intelligent approaches leveraging deep neural networks. These new models allow not only to automate the processes but also enable improving the quality of a specific type of document: handwritten text. The increasing volume of handwritten documents that are being digitized pose both a challenge, given the struggle of traditional methods on correctly identifying text, and an opportunity, for the impact the development of detection and recognition models might have in the public archive.

The present work dives deep in the neural network architectures that can be used for detecting text and intelligently recognizing characters, performing a quantitative analysis of performance on the segmentation and recognition models. It also evaluates qualitatively the models by applying them to the digital assets of the Library of the Faculty of Exact and Natural Sciences of the University of Buenos Aires, using the techniques on the correspondence of Mario Bunge. The resulting information is thus systematized and made available for future works and researches.

Keywords: character recognition; neural networks; semantic segmentation; handwritten text; deep learning.

Capítulo 1

Introducción

1.1. Tema de investigación

A partir del desarrollo de los primeros sistemas informáticos, uno de los primeros problemas que se buscó resolver fue el reconocimiento óptico de caracteres. Desde los desarrollos de Emanuel Goldberg para la transformación de caracteres a código telegráfico y su creación de la máquina estadística veinte años después (Schantz, 1982), la búsqueda de transformar texto de letra de molde a canales analógicos y digitales fue creciendo y sofisticándose a lo largo de todo el siglo veinte y veintiuno.

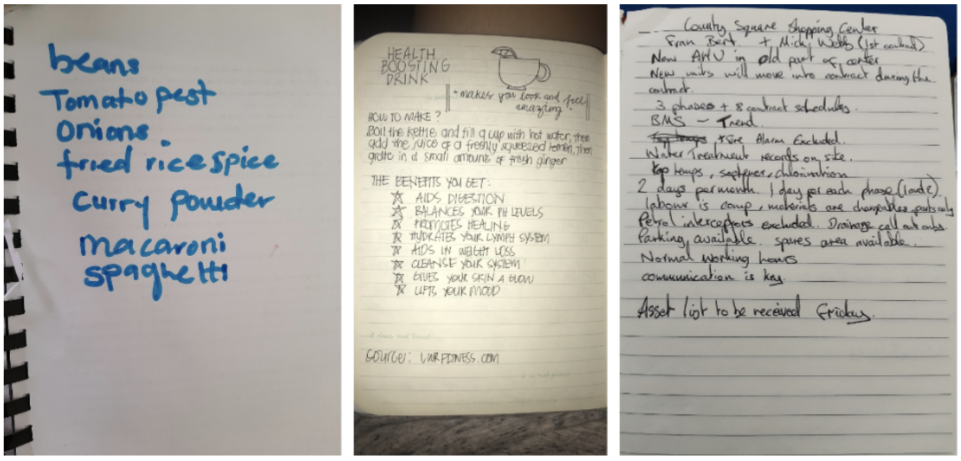
Hoy día, a pesar del enorme avance en términos de capacidad de cómputo y de democratización de los recursos informáticos, gran parte de los documentos con los que lidiamos en el día a día siguen estando escritos a mano. Además de un enorme acervo histórico de archivos de este tipo, cosas de la vida cotidiana como las recetas médicas y las notas de clases de estudiantes siguen siendo escritas mayormente a mano.

En el campo del reconocimiento óptico de caracteres en letra de máquina o de molde se han realizado enormes avances que facilitaron su incorporación en una gran cantidad de ámbitos: desde la detección de documentos como pasaportes en aeropuertos hasta la digitalización de libros y el reconocimiento de patentes de automóviles. La popularización de sistemas del reconocimiento óptico de caracteres (OCR, por su acrónimo en inglés) y su significativa mejora en el desempeño tanto partiendo de procesamiento de señales digitales como de redes neuronales han generado, como mencionamos anteriormente, incorporados en muchos aspectos del día a día.

En este contexto, y a pesar de enormes avances en el campo de los textos escritos en letra de molde o a máquina, el reconocimiento de texto escrito a mano ha presentado otras complejidades que no hacen tan sencilla su detección y digitalización. Por lo general, los desafíos que se identifican consisten en dos tipos: reconocer las áreas en las que se encuentra escrito el texto -tanto en segmentos como en líneas- y reconocer los caracteres, así como el modelo de lenguaje. En algunos casos -como cuando se trata de dominios específicos tal como las matemáticas o la medicina- es deseable que las palabras se adecuen a ciertas normas establecidas en distintos modelos de lenguaje (Kamalanaban y col., 2018).

Dentro del campo de OCR existen dos grandes tipos de modelos: los que reconocen letras y los que reconocen palabras. En el ámbito de este primero, conjuntos de datos como MNIST (Modified National Institute of Standards and Technology, desarrollado por LeCun y Cortes, 2010) han sido de gran ayuda. En el ámbito del segundo, el conjunto de datos IAM (Marti & Bunke, 1999) es uno de los más relevantes (Figura 1.1) en inglés, junto con el RIMES (Grosicki y col., 2006) en francés.

Figura 1.1: Conjunto de datos GNHK (Lee y col., 2021) - Ejemplos

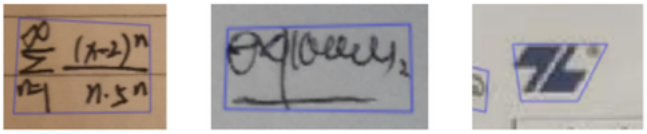


Sin embargo, a pesar de tener desempeños que rozan la perfección al detectar letras en conjuntos como el mencionado anteriormente (Kowsar y col., 2018), presentan un conjunto de problemas, como la detección de caracteres concatenados en el caso de escritura de tipo cursiva, el solapamiento parcial de caracteres y el reconocimiento de las cadenas de los mismos como un todo coherente y cohesivo.

En la actualidad, existen esfuerzos considerables para desarrollar métodos automáticos para transcribir texto escrito, intentando ir del reconocimiento óptico de caracteres al reconocimiento inteligente de caracteres (ICR). Por lo general, el flujo de datos consiste en realizar segmentación de páginas para identificar regiones del texto en un documento y luego realizar el reconocimiento del texto escrito a mano. Del mismo modo, existen dificultades asociadas a si el reconocimiento es de tipo *offline*, tomando el texto como un todo, u *online*, donde el momento del trazo, su forma e intensidad influyen en las predicciones siguientes.

Respecto del preprocesamiento de las imágenes y la extracción de atributos, surgen desafíos asociados a la nitidez de la imagen, su calidad y forma. En documentos históricos, no necesariamente digitalizados en alta resolución y cuyo papel puede presentar roturas, es importante contar con métodos robustos que cuenten con características que permitan sobreponerse.

Figura 1.2: Tipos de caracteres encontrados en GNHK (Lee y col., 2021)



(a) %math% (b) %SC% (c) %NA%

El desempeño de los modelos de aprendizaje automático y, particularmente, los de aprendizaje profundo ha crecido exponencialmente con el aumento de la capacidad de procesamiento, así como la sofisticación de las arquitecturas utilizadas para obtener resultados, lo que ha permitido una mejora significativa en el desempeño de este tipo de modelos. En este sentido, los procesos propios de este tipo de trabajos (procesamiento, segmentación, extracción de atributos, entrenamiento e inferencia) han sufrido grandes modificaciones.

El presente trabajo busca estudiar las maneras de reconocer caracteres mediante aprendizaje profundo, evaluando arquitecturas para localización de texto y extracción de atributos. Además, se indagará brevemente sobre la posibilidad de extender los modelos para el reconocimiento de caracteres no alfanuméricos, como símbolos matemáticos.

1.2. Antecedentes

Dentro de lo que Mitchell, 1997 considera como aprendizaje automático, una de las primeras maneras de intentar resolver el problema del reconocimiento óptico de caracteres fue mediante la utilización de filtros como los provenientes del procesamiento de señales digitales. Una vez confeccionados estos atributos, que poseían características estáticas, se aplicaban técnicas como las máquinas de soporte vectorial -SVM, por su acrónimo en inglés- (Géron, 2017) para su clasificación. El proceso era, como puede suponerse, secuencial. A la búsqueda de incrementar las correlaciones cruzadas de las señales filtradas le precedía un arduo trabajo de preprocesamiento y le sucedía el desarrollo de las sofisticadas tareas de entrenamiento de modelos. En estos casos, la sola construcción de filtros podía ser una tarea digna de una tesis de doctorado (Goodfellow y col., 2016).

El surgimiento de nuevas técnicas como las de aprendizaje profundo permitieron desarrollas procesos íntegros, donde se pudiera entrenar punta a punta un modelo que construya los filtros, calcule adaptativamente sus pesos, clasifique los objetos y sea robusto ante modificaciones.

Así, la secuencialidad que se mantuvo en algunos modelos (como el propuesto por Chung y Delteil, 2019) refiere a la segmentación de tipo *blob detection* primero -tanto para las páginas (Figura 1.1 como para las líneas Figura 1.1)- y su clasificación después. Sin embargo, en lo que respecta a los tipos de filtros y su adaptabilidad, los modelos desde hace ya muchos años plantearon la posibilidad de ser agnósticos del lenguaje utilizado (Graves & Schmidhuber, 2009), con las limitaciones evidentes que implica en términos de modelo de lenguaje subyacente.

La incorporación de mecanismos de atención (Vaswani y col., 2017) dio

lugar a la creación de arquitecturas recurrentes multidimensionales como las *Multi-Dimensional Long Short-Term Memory recurrent neural networks* detalladas en Bluche y col., 2017, donde la información que ingresa a la red es de tipo multidimensional. Desarrollos como los de Chung y Delteil, 2019 han señalado que la relación entre costo computacional y beneficio de las redes LSTM multidimensionales no justifican su utilización, promoviendo redes recurrentes bidireccionales.

La utilización de técnicas de *data augmentation*, explicadas en Chollet, 2017, permitieron robustecer el comportamiento de los modelos de aprendizaje profundo frente a variaciones en el trazo de los caracteres mediante lo que se denomina regularización. Así, una palabra de un texto del conjunto de datos (como la de la Figura 1.2) sufrirá una ligera modificación en su ángulo, brillo, forma o nitidez.

En lo que respecta a su aplicación en otros ámbitos, una de las técnicas más habituales para la adaptación de dominios es el aprendizaje por transferencia (en inglés *transfer learning*). El aprendizaje por transferencia se ha constituido con uno de los principales métodos que permiten apalancar el conocimiento de grandes conjuntos de datos para distintas tareas y conjuntos más acotados (Chollet, 2017). Aprovecha casi exclusivamente la estructura jerárquica de las redes neuronales convolucionales -una de cuyas características sobresalientes es la detección de patrones locales por lo acotado de sus conexiones entre capas (Goodfellow y col., 2016) para modificar las representaciones en niveles más altos de abstracción y utilizarlos en otros dominios.

Si bien el alcance de este trabajo se circunscribe a la evaluación de las arquitecturas para detección y reconocimiento de caracteres escritos a mano, este tipo de métodos de reconocimiento de caracteres pueden extenderse a

distintas aplicaciones como, por ejemplo, documentos del ámbito médico -replicando lo hecho por Kamalanaban y col., 2018. Otro ámbito en el que se puede evaluar este tipo de modelos, y que será parcialmente abordado en el apartado de análisis cualitativo del presente trabajo, es el de reconocimiento de ecuaciones matemáticas y su análisis de coherencia. Un conjunto de datos conmúnmente utilizado para este fin es el de la *Competition on Recognition of Online Handwritten Mathematical Expressions* (CROHME, por su acrónimo en inglés) (Awal y col., 2010), que se puede observar en la figura 1.3.

Figura 1.3: Conjunto de datos CROHME

1.3. Aportes al campo de estudio

Con el desarrollo de nuevas arquitecturas para segmentación de palabras y detección de texto se pretende incorporar el estado del arte en aprendizaje profundo a desarrollos como los realizados por Chung y Delteil, 2019, permitiendo extender el enfoque a múltiples párrafos por hoja a ser analizada y extendiéndola a otros dominios. En términos técnicos, compararemos las arquitecturas de detección de palabras y reconocimiento de caracteres del modelo mencionado anteriormente -consistente en redes neuronales convolucionales, incluyendo modelos de aprendizaje profundo con arquitecturas de tipo *RCNN*, para lo primero y una combinación de redes convolucionales y recurrentes para lo segundo.

Como objetivo del trabajo pretendemos aplicar las arquitecturas y modelos mencionados para identificar las palabras presentes en la correspondencia de Mario Bunge disponibles en la Biblioteca Digital de la Facultad de Ciencias Exactas y Naturales de la Universidad de Buenos Aires. Se asume que dicha información permitirá nuevas líneas de investigación tanto para catalogar el contenido de los textos como para análisis más sofisticados del tipo de modelado de tópicos, enriqueciendo el acervo público de la Facultad.

Capítulo 2

Marco teórico

2.1. El reconocimiento óptico de caracteres tradicional

Existen dos grandes tipos de reconocimiento de caracteres: *offline*, cuando se considera al texto a analizar como un todo, y *online*, donde el momento del trazo, su forma e intensidad influyen en las predicciones siguientes. En este marco teórico desarrollaremos las características y arquitecturas diseñadas para reconocimiento de tipo *offline*.

2.1.1. Procesamiento digital de señales

El reconocimiento óptico de caracteres mediante la utilización de procesamiento de señales con filtros ha sido la solución utilizada durante varias décadas para convertir documentos o imágenes a texto. En muchos casos, el estándar OCR ha permitido resolver muchas tareas asociadas al ingreso

manual de datos de documentos, con un resultado que tiene un alto grado de eficacia y reduce significativamente el tiempo respecto del ingreso manual.

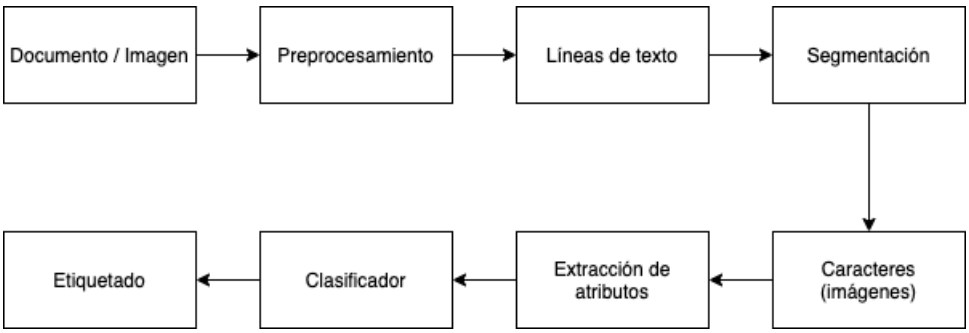
En contextos de baja variabilidad -es decir, tipografía, formato y estructura similar a lo largo de todo el documento- este procesamiento es altamente efectivo. Aún cuando puede tener un tiempo de configuración alto previo al análisis para identificar y acotar el análisis a estos contextos de baja variabilidad, el resultado es suficientemente satisfactorio como para que haya valido la pena hacerlo de esta manera durante décadas.

Así, la posibilidad de reconocer patrones en texto escrito a máquina e incluso a mano ha dado lugar a la digitalización de documentos como facturas o para distribución a alta velocidad de cartas y paquetes en el servicio postal (Nasien y col., 2010) ha generado que se desarrollen un conjunto de mecanismos para facilitar la compleja tarea de definir plantillas y reglas para cada campo a analizar, dándole al OCR la baja variabilidad necesaria.

Analizando el proceso de clasificación de los caracteres surge, como es evidente, el primer problema asociado a esta técnica de reconocimiento: cada modificación requiere de una nueva regla. Además, como también indica Nasien y col., 2010, la falta de flexibilidad en lo que respecta a modificaciones en el documento puede ocasionar errores como falsos positivos. La automatización completa en este tipo de procesos es, entonces, imposible.

Como podemos ver en la siguiente imagen (proveniente de Saba y col., 2011, el proceso de detección de texto con las técnicas tradicionales tiene múltiples pasos, muchos de ellos necesariamente manuales. Cada uno de estos pasos, de una creciente complejidad, pueden haber requerido de por sí -por ejemplo, en el caso de la extracción de atributos mediante filtros como FFT, Kernel o Gaussianos- muchos años de investigación sólo para optimizar un solo eslabón de la cadena:

Figura 2.1: Diagrama de flujo de un sistema de OCR tradicional



Ante la complejidad de este tipo de procesos -donde la técnica de Matrix Matching (Singh & Desai, 2016), consistente en medir la distancia de un caracter con un conjunto prefijado de caracteres "patrón" era uno de los métodos más avanzados- y la imposibilidad de automatizar el flujo, nuevas alternativas como el reconocimiento inteligente de caracteres cobraron relevancia. Una de las primeras opciones fue el uso de técnicas como Vecinos Más Cercanos y Máquinas de Soporte Vectorial (kNN y SVM, respectivamente, por sus acrónimos en inglés), mediante las que se buscó mejorar el desempeño de los filtros tradicionales (Nasien y col., 2010). Ambos métodos lograron una mejora significativa sobre los filtros y los clasificadores lineales tradicionales, pero aún habría un margen de reducción de error -y, además, de automatización de los procesos reduciendo las reglas intrínsecas en el OCR- de la mano de las redes neuronales.

2.2. Las redes neuronales y el reconocimiento inteligente de caracteres

Dentro del universo de la redes neuronales existen distintas arquitecturas, adaptables en mayor o menor medida a las necesidades de un dominio. En el caso del reconocimiento de caracteres, las propiedades deseables de las redes serían dos: en primer lugar, que tengan la capacidad de reconocer caracteres escritos a mano y las palabras que estos conforman, entendiendo la jerarquía que eso conlleva; en segundo, que reduzcan la cantidad de pasos manuales (como se vio en la figura 2.1, al menos siete pasos partiendo de un documento o imagen). Para entender cómo ambas propiedades pueden cumplirse utilizando redes, abordaremos brevemente el concepto de red neuronal convolucional (CNN) y, a partir de ellas, las arquitecturas que se pueden desarrollar utilizándolas.

2.2.1. Redes neuronales convolucionales

Una de las principales diferencias entre las redes densamente conectadas y las redes convolucionales es que, mientras las primeras reconocen patrones globales, las segundas tienen la capacidad de reconocer patrones locales. Las redes convolucionales logran analizando la imagen mediante un filtro, denominado "ventana", que identifica patrones en distintas posiciones de la imagen de manera secuencial. Esto les da a este tipo de redes dos características sumamente relevantes para los problemas que buscamos resolver (Chollet, 2017):

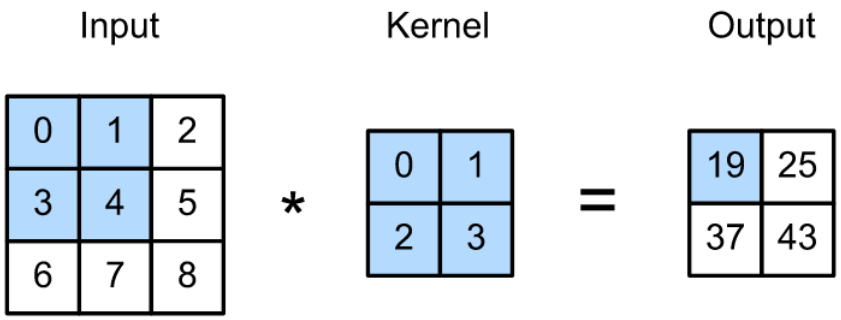
- Son invariantes a la traslación, es decir, una vez que reconoce un patrón en un lugar de la imagen pueden reconocerlo en otro, independiente-

mente de su ubicación en la misma. Esto las convierte en herramientas eficientes para el procesamiento de imágenes.

- Pueden aprender jerarquías espaciales de patrones. Al igual que en el mundo visual, pequeños patrones locales aprendidos en las capas más "bajas" de una red -aquellas de mayor concreción- pueden acoplarse para construir estructuras de mayor abstracción en las capas más altas (al igual que, por ejemplo, un rostro humano está compuesto por un conjunto de rectas, curvas y elipses).

Las convoluciones, en tanto operación matemática, transforma dos funciones en una tercera que refleja, de cierto modo, la respuesta de una a la otra. En el caso de las redes neuronales, las convoluciones se construyen sobre mapas de atributos (en inglés, *feature maps*), arreglos tridimensionales que constan en dos ejes espaciales -ancho y alto- y otro de profundidad -que, en un sentido visual, se podría identificar como los canales y se corresponde al espacio de los colores- (Goodfellow y col., 2016). Veamos, con un ejemplo de Zhang y col., 2020, una convolución -o, como se la define en el mundo de procesamiento de señales digitales, operación de correlación cruzada- con una ventana de 2x2:

Figura 2.2: Convolución o correlación cruzada con ventana de 2x2



La respuesta de la convolución, esto es, la magnitud en la que se superponen

ambas funciones luego de determinadas operaciones como la traslación, da como resultado lo que se conoce como "mapa de salida". A diferencia de lo mencionado anteriormente, donde la profundidad representa el espacio de los colores, aquí va a indicar la cantidad de filtros.

El concepto de filtro aquí es similar al de la literatura tradicional (Saba y col., 2011), donde se señala la presencia o ausencia de un determinado atributo o patrón. Por ese motivo, y a diferencia del espacio de entrada -donde para las imágenes en escala de grises corresponde un canal y aquellas en escala RGB tres-, la profundidad de los mapas de atributos de salida es arbitraria. Así, siguiendo a Chollet, 2017, podemos indicar dos parámetros clave de las convoluciones:

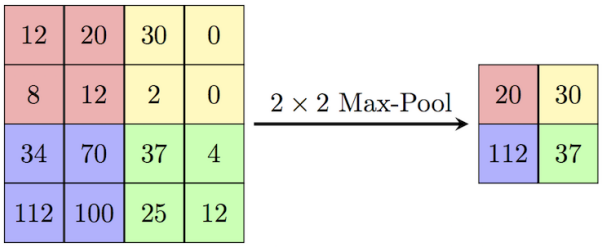
- El tamaño de las "ventanas" mencionados previamente, donde las opciones más habituales son 3x3 o 5x5.
- La profundidad del mapa de atributos de salida. En estos casos es habitual que a medida que se reduce el tamaño de la imagen de entrada se incremente la cantidad de filtros.

Como último punto, es importante destacar otras herramientas con las que cuentan las redes convolucionales para forzar, en cierta medida, el aprendizaje de atributos jerárquicos a medida que se va "subiendo" en las capas de la red. En primer lugar, los "saltos" o *strides* son una manera de recorrer el espacio de las imágenes de una manera no secuencial, sino deslizándose en unidades mayores que la original. Una segunda manera -más eficiente en términos de desempeño- es utilizando operaciones de *pooling*, donde se destacan dos: *average* y *max*.

Esta segunda, la más popular (siendo uno de los motivos de su popularidad que el valor máximo pareciera reflejar mejor que el promedio la presencia

de patrones), consiste en agrupar "ventanas" del mapa de atributos de entrada y dar como respuesta el valor máximo de cada canal. Como indica Chollet, 2017, es similar a la convolución con la diferencia de que, en lugar de utilizar transformaciones lineales aprendidas en el proceso se utiliza un operador hiperparametrizado, el *max tensor*. Se puede observar el concepto la siguiente figura:

Figura 2.3: *Max-pooling* con ventana de 2x2



Con esto, no sólo se reduce la complejidad computacional del entrenamiento de la red sino que se la hace más robusta y resistente al sobreajuste, reduciendo el número de parámetros y forzando la necesidad de pasar la mayor cantidad de información por un canal lo más acotado posible (MacKay, 2003).

2.2.2. Redes convolucionales basadas en regiones

La detección de objetos en imágenes es uno de los campos donde más avances se han registrado en los últimos años. Las primeras arquitecturas fueron las redes convolucionales basadas en regiones (RCNN), definiendo polígonos regulares (*bounding boxes*) que luego son analizadas en las regiones de interés (ROI) con el objetivo de clasificar cada una de las regiones propuestas según las clases previamente definidas (Ren y col., 2015).

Posteriormente se evolucionó a Fast-RCNN, una arquitectura de dos etapas que redujo la necesidad de pasar todas las regiones propuestas cada vez, sino que se realiza una convolución una única vez (mediante el *pooling* de ROI) y a partir de eso se genera un mapa de atributos. Así, primero se proponen regiones de interés -mediante una *Region Proposal Network* o RPN- y luego clasifica los objetos y propone los valores de los polígonos. La siguiente optimización vino de la mano de la eficiencia computacional, creando la Faster-RCNN (Ren y col., 2015).

Finalmente, la arquitectura de Mask-RCNN (He y col., 2017) tomó lo desarrollado por Faster-RCNN y le incorporó la capacidad de segmentar -además de las imágenes- las instancias definiendo máscaras para cada ROI, en paralelo con las ramas existentes para clasificación y regresión de polígonos regulares (*bounding boxes*).

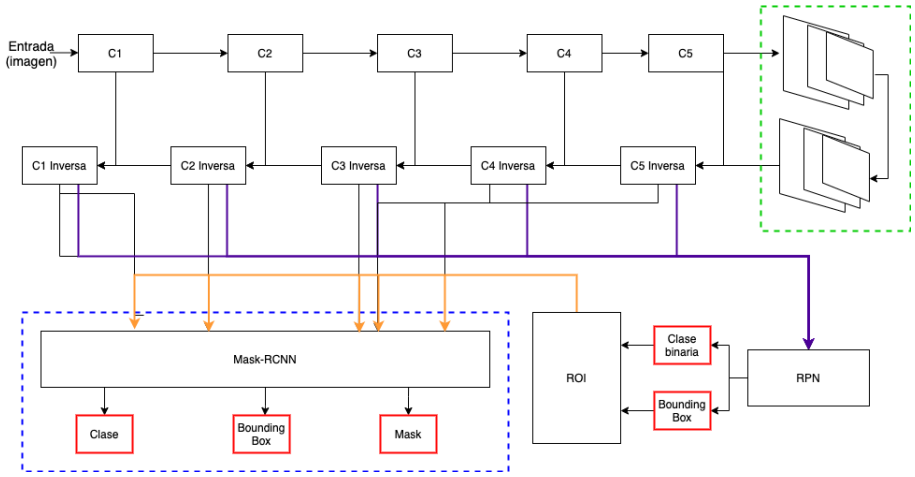
La detección y clasificación de objetos de Mask-RCNN consta dos etapas: en primer lugar, se realiza una segmentación semántica que clasifica distintas regiones que pueden ser similares entre sí de distintos objetos, es decir, definiendo los objetos y sus delimitaciones; en segundo, se reconocen las instancias distinguiendo entre cada uno de los objetos -esto es, se segmenta cada instancia mediante una máscara dentro de cada *bounding box* y dentro de cada una se busca identificar la clase correspondiente.

Para mejorar la predicción y construir de manera adecuada las máscaras, Mask-RCNN cuenta con un factor distintivo: el alineamiento de píxeles entre las regiones de interés, resaltada en naranja en la figura 2.4, reemplazando el mencionado *pooling* de las regiones de interés con el que cuenta Faster-RCNN.

Esta arquitectura mejora, según He y col., 2017 en desempeño, eficiencia computacional y flexibilidad a las versiones anteriores de redes basadas en

regiones, dado que no le agrega una carga computacional significativa y permite generalizar fácilmente. En términos de salidas, como se puede apreciar en la figura 2.4, a las de la red Faster-RCNN se le incorpora la de una máscara, identificada dentro la línea punteada azul. Fuera de la región de Mask-RCNN se puede observar, en dicha imagen, las métricas asociadas a la proposición de regiones.

Figura 2.4: Arquitectura de segmentación - FPN + Mask RCNN

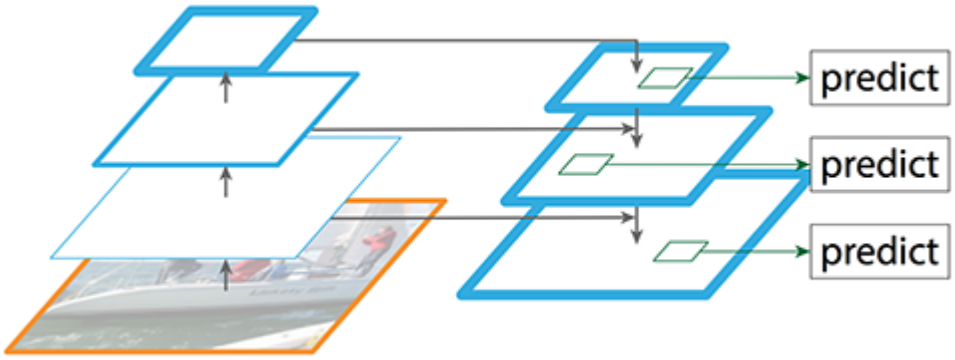


Además de sofisticar las técnicas de detección y clasificación de objetos mediante sus salidas se puede mejorar la capacidad de representación de los atributos. Para ello, puede incorporarse antes de la proposición de regiones -esto es, de la RPN- una *Feature Pyramid Network* (FPN), resaltada en verde en la figura 2.4. La gran ventaja de la FPN respecto de otras arquitecturas de detección de objetos como las *Single Shot Multibox Detector* (SSD, Liu y col., 2015) es que combina, en un mismo lugar, atributos de baja resolución y alta significatividad semántica con otros de alta resolución y poca significatividad semántica.

Para combinar ambos tipos de atributos, la FPN conecta las capas superiores de la red -o pirámide-, aquellas con mayor valor semántico, con las de

la base, que contiene la mayor resolución, mediante conexiones de laterales y verticales. La arquitectura que surge se puede observar en la figura 2.5, obtenida de Lin y col., 2017, y detalla lo que se aprecia en el recuadro verde de la figura 2.5.

Figura 2.5: Feature Pyramid Network



La representación de la información de las FPN como parte de una arquitectura de segmentación permite contar a la vez con resolución e información semántica útil mediante las conexiones verticales y mejorar la ubicación espacial de los objetos con las conexiones laterales, siendo a la vez eficiente a nivel computacional y de representación informacional. Así, podemos contar con un modelo de detección que detecta los polígonos (*bounding boxes*), propone regiones significativas y -en caso de que corresponda- define las máscaras para esas regiones (He y col., 2017).

2.2.3. Transformación espacial

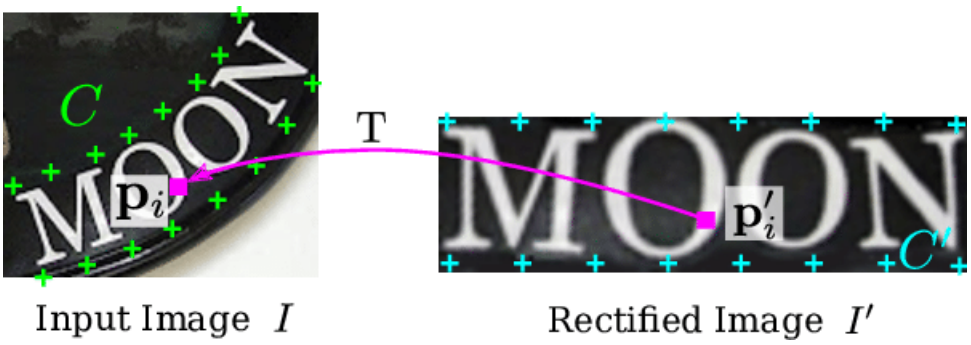
Un desafío recurrente para la clasificación de caracteres en imágenes es, además de la detección y segmentación, la normalización. En muchas oca-

siones, si bien el modelo implementado logra segmentar correctamente el espacio de acuerdo a la ausencia o presencia de entidades (lo que se denomina *segmentación semántica*), la imagen tiene una composición tal que los caracteres tienen una forma que dificulta su comprensión. Esto se debe a que las redes convolucionales, como mencionamos en 2.2.1, son invariantes a la traslación pero no a la rotación y a otro tipo de transformaciones afines.

Para resolver este problema, en los últimos años se desarrollaron redes neuronales que transforman espacialmente las imágenes, permitiendo robustecer la clasificación independientemente de la imagen original, generando un salto cualitativo y cuantitativo en el desempeño respecto de los métodos tradicionales de transformación espacial (Yang y col., 2019).

Las redes de transformación espacial (Baek y col., 2019), desarrolladas originalmente por Google DeepMind, facilitan las tareas de segmentación de los caracteres dentro de los polígonos detectados mediante la utilización de puntos de referencia (denominados *fiducial points* en la literatura). Estos anclajes -resaltados en verde y celeste en la figura 2.6- permiten generar una versión rectificada de la imagen de entrada mediante distintas transformaciones. Si bien las transformaciones más habituales han sido las afines, una de las que mejor resultado ha dado es la de tipo Thin-Plate Spline, basada en el método originalmente publicado en Bookstein, 1989 y la que se utiliza en Shi y col., 2016. Esta transformación permite obtener más grados de libertad en la modificación -también llamada, según la literatura, *deformación*- de la imagen de entrada (Dai y col., 2017).

Figura 2.6: Transformación espacial (Shi y col., 2016)



Una de las ventajas que presentan las redes de transformación espacial es su alto grado de modularidad, esto es, la capacidad de ser integradas con arquitecturas existentes de redes convolucionales sin mayores complejidades. En este sentido, son un componente ideal para la sofisticación de procesos de localización y reconocimiento de caracteres entrenables punta a punta mejorando la invariancia a la traslación ya existente en las redes convolucionales.

2.2.4. Redes recurrentes y bidireccionales

Habiendo identificado los tipos de modelos y arquitecturas que pueden ayudarnos a solucionar la segmentación de la página y de líneas de texto escrito a mano, el paso siguiente es reconocer el texto escrito a mano. Para ello, nos apoyaremos en lo detallado en 2.2.1.

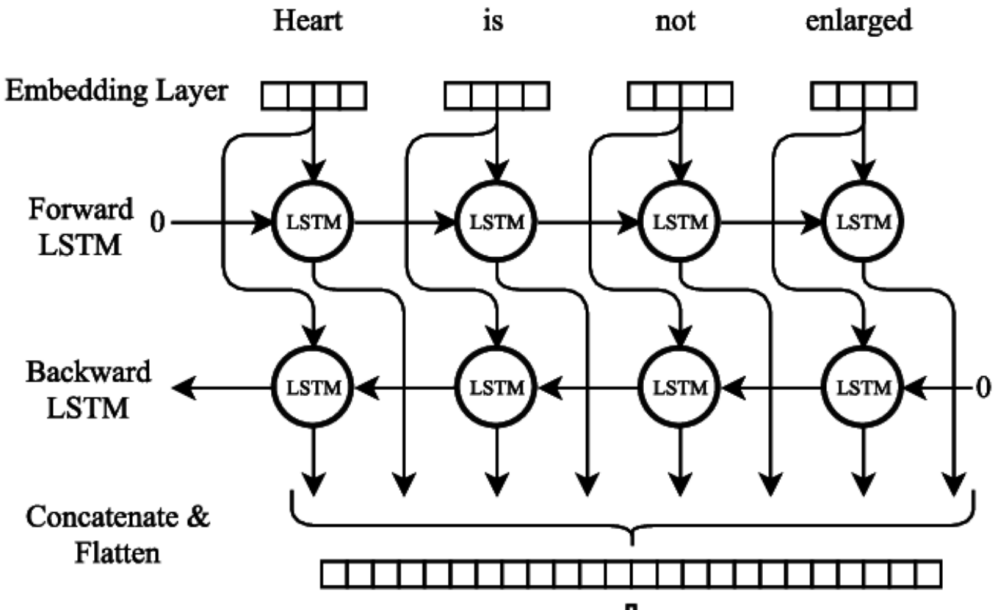
Una primera aproximación al reconocimiento de caracteres escritos a mano fue mediante las redes recurrentes de tipo LSTM (Long-Short Term Memory) multidimensionales (Graves & Schmidhuber, 2009). Sin embargo, co-

mo señalan Chung y Delteil, 2019, se pueden obtener desempeños similares con una complejidad computacional significativamente reducida con redes LSTM unidimensionales apoyadas en arquitecturas bidireccionales o apoyadas en redes convolucionales (como hizo Puigcerver, 2017).

Las redes convolucionales permiten extraer atributos -en este caso, con arquitecturas de tipo ResNet, VGG y RCNN (Chollet, 2017)- que luego serán procesados secuencialmente por una red neuronal recurrente en una arquitectura conocida como BiLSTM, que se puede observar en la figura 2.7 obtenida de Cornegruta y col., 2016. El resultado de esta red es luego clasificado con un vector de probabilidades en una forma similar en la que se detalló para la arquitectura del apartado anterior, en la figura 2.4.

Figura 2.7: Arquitectura BiLSTM

2.7



Además de la complejidad computacional mencionada, otra limitación de

las redes recurrentes es la necesidad de que los datos de entrada sean fijos y no se pueden obtener de cada estado actual de la red. Las arquitecturas bidireccionales, como permite apreciar la figura 2.7, no tienen ningunas de esas restricciones. Así, la información contextual -que para series temporales serían el futuro y el pasado y en este caso los caracteres ubicados a cada lado del caracter a analizar- aporta una ventaja significativa a la hora del reconocimiento de cada caracter, lo que las hace una solución superior a las redes recurrentes unidireccionales.

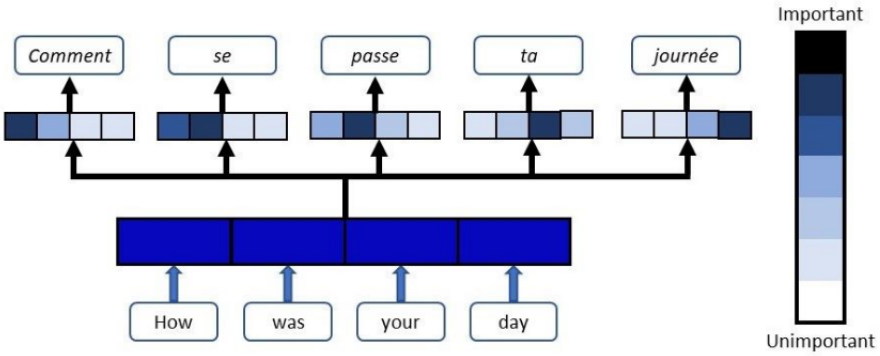
Al ser arquitecturas independientes desde el punto de vista de las conexiones entre sí (Cornegruta y col., 2016), las redes bidireccionales pueden ser entrenadas con los algoritmos similares a las unidireccionales, con la salvedad de la técnica de retropropagación (Chollet, 2017) a ser aplicada. En este caso, lo que se suele utilizar es la retropropagación a través del tiempo (mejor conocida por su denominación en inglés, *backpropagation through time* o BPTT) que requieren actualizaciones secuenciales de las capas de entrada y salida de datos (Schuster & Paliwal, 1997).

2.2.5. Mecanismos de atención

En términos generales, se denomina atención (más conocido por su término en inglés, *Attention*) a los componentes de las redes neuronales que le otorgan mayor relevancia -esto es, redirigen la atención- a determinados atributos a la hora de realizar el procesamiento de la información. Los tipos de atención que existen y que se optimizan habitualmente en las redes neuronales profundas son dos: los que gestionan la interdependencia entre los elementos de entrada y de salida (atención general) y los que lo hacen entre los elementos de entrada (auto-atención o *self-attention*) (Vaswani y col., 2017).

Los mecanismos de atención, originalmente diseñados para resolver problemas en modelos secuenciales como el *seq2seq*, son particularmente relevantes en tareas secuenciales dado que permiten otorgarle mayor relevancia a algunos aspectos del objeto de análisis. Esto resuelve -o, al menos, mitiga el impacto- uno de los principales desafíos con los que cuentan las redes recurrentes (cuya arquitectura y lógica fue introducida en el apartado 2.2.4), el del procesamiento de secuencias extensas de información. Para ello, retiene los estados de las capas ocultas mediante un mapeo de cada estado en cada etapa de la recurrencia, lo que permite comprender cuáles son aquellos que más información aportan respecto de la salida (Vaswani y col., 2017).

Figura 2.8: Mecanismo de Atención
(Vaswani y col., 2017)



Al ubicar un módulo de atención luego de un modelo secuencial, como una red recurrente bidireccional, se enriquece la información contextual y se mejora el desempeño en la salida.

Capítulo 3

Herramientas y conjuntos de datos

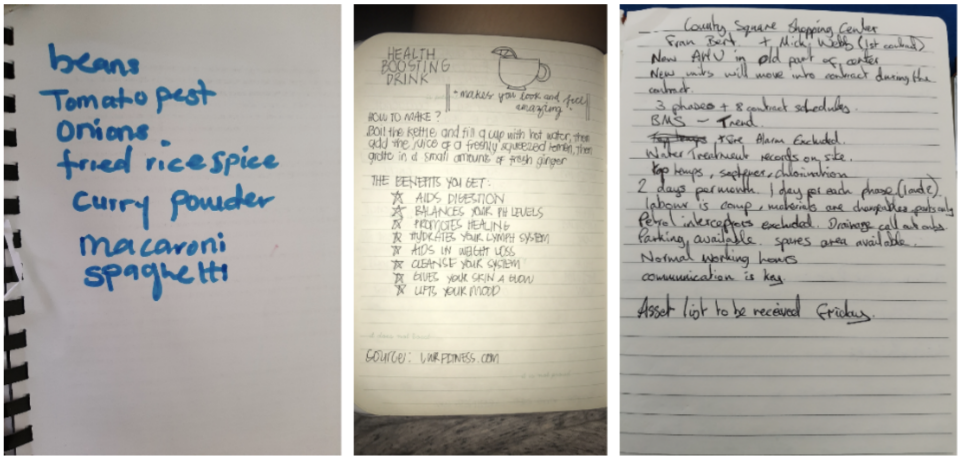
3.1. El conjunto de datos GNHK

El conjunto de datos a analizar, llamado GoodNotes Handwriting Kollection - GNHK (Lee y col., 2021), forma parte del acervo público. Fue presentado en la *16th International Conference on Document Analysis and Recognition (ICDAR 2021)* en septiembre de 2021 por un equipo de investigación en visión computacional e inteligencia artificial compuesto por miembros de las compañías GoodNotes y Amazon Web Services (AWS).

Este conjunto consiste en 687 páginas de texto escaneado correspondiente a distintas fuentes -listas ordenadas, libretas personales, notas de recordatorio- escritas en idioma inglés por personas de cuatro continentes. Son imágenes no recortadas, es decir, que registran contornos, contexto y diferencias de rotación, brillo y color, con el objetivo de profundizar la investigación en

técnicas de localización y reconocimiento de caracteres. Como destacan en Lee y col., 2021, es comparable en términos de volumen con los conjuntos de datos presentados en Marti y Bunke, 1999 y Grosicki y col., 2006.

Figura 3.1: Conjunto de datos GNHK (Lee y col., 2021) - Ejemplos

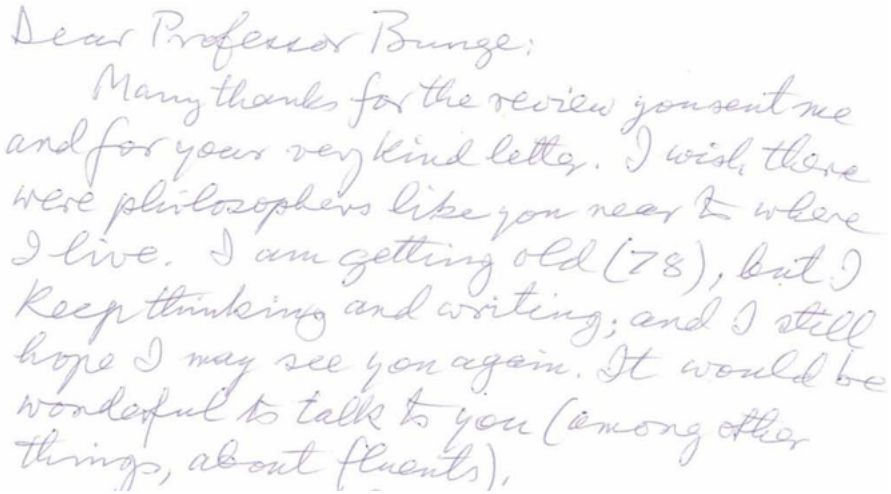


Además de caracteres alfanuméricos, el conjunto de datos cuenta con otros caracteres no imprimibles en ASCII, tanto en términos matemáticos (denominados %math %) como de texto no interpretable - identificados como garabatos o *scribble*. El cuarto tipo de caracter se compone por caracteres que no son texto a pesar de haber sido anotado como tal (ver 1.2).

Para la evaluación de los modelos se dividió el conjunto de datos de la siguiente manera: el 75 % para entrenamiento (516 imágenes) y el 25 % para *test* (173 imágenes), tomando como referencia a Lee y col., 2021, de donde tomaremos las métricas cuantitativas a analizar en la segmentación. Los resultados obtenidos y graficados en las secciones de *Análisis cuantitativo* del capítulo siguiente son aquellos correspondientes al conjunto de datos de *test*.

Para la evaluación cualitativa, tal como se mencionó en el aporte esperado, se utiliza la correspondencia de Mario Bunge disponible en la Biblioteca Digital de la Facultad de Ciencias Exactas y Naturales de la Universidad de Buenos Aires. A continuación se presentan dos documentos de ejemplo de la correspondencia, en este caso respuestas al profesor Bunge.

Figura 3.2: Documento de ejemplo 1- Correspondencia Mario Bunge

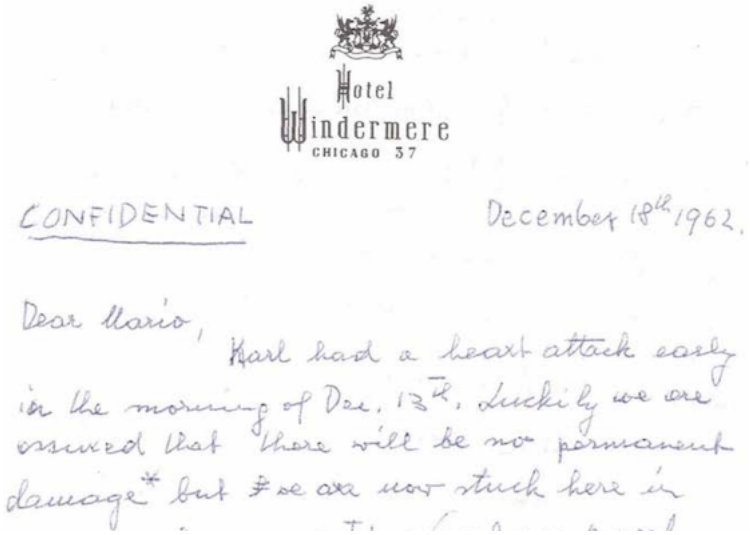


Dear Professor Bunge:

Many thanks for the review you sent me and for your very kind letter. I wish there were philosophers like you near to where I live. I am getting old (78), but I keep thinking and writing; and I still hope I may see you again. It would be wonderful to talk to you (among other things, about fluents).

Si bien ambas son en inglés existen también muestras en español, lo que permite evaluar, si bien de modo cualitativo, la capacidad de abstracción del modelo respecto del idioma del texto.

Figura 3.3: Documento de ejemplo 2 - Correspondencia Mario Bunge



Hotel
Windermere
CHICAGO 37

CONFIDENTIAL December 18th 1962.

Dear Mario, Karl had a heart attack early in the morning of Dec. 13th. Luckily we are assured that there will be no permanent damage* but we are now stuck here in

Es importante destacar que si bien el análisis es cualitativo el resultado del

mismo -esto es, los polígonos y caracteres predichos dentro de los mismos- estarán disponibles para cada uno de los documentos disponibles en el mencionado acervo digital.

3.2. Herramientas y *frameworks* utilizados

El requerimiento de cómputo acelerado hizo que el equipamiento con el que se contaba originalmente (una computadora Macbook Pro de alto rendimiento con una unidad de procesamiento gráfico) no fuera suficiente ejecutar los modelos de aprendizaje profundo.

Para lograr entrenar redes neuronales con un consumo de memoria y CPU por encima de lo considerado en la evaluación de factibilidad se utilizaron servicios de cómputo en la nube de Amazon Web Services (AWS). Esto nos permitió contar con instancias con hasta 488 GiB y 64 vCPU y cómputo acelerado con 8 GPUs NVIDIA Tesla V100 de una manera escalable y modular. El entrenamiento de los modelos se hizo utilizando *containers* con imágenes (*kernels)* de PyTorch 1.0 (Paszke y col., 2019), lo que permitió paralelizar el entrenamiento (Services, s.f.) mediante los *Training Jobs* de Amazon SageMaker.

Para replicar los modelos desarrollados por autores como Chung y Delteil, 2019 y Scheidl, 2018 se hizo uso del código disponible para su ejecución en repositorios públicos como *GitHub*, de donde además se tomaron ejemplo de segmentación y detección de Facebook AI Research (Wu y col., 2019). Para tener una base de código homogéneo se migró parte del código existente en Apache MXNET a PyTorch -una librería de aprendizaje profundo en Python con un alto grado de desarrollo y una creciente comunidad, que además incorporó *frameworks* otrora populares como *Caffe 2*. Del mismo

modo, se hizo uso de múltiples ejemplos y soluciones desarrollados en repositorios de AWS y disponibles también en línea para construir trabajos de entrenamiento (*Training Jobs*) de Amazon SageMaker.

Capítulo 4

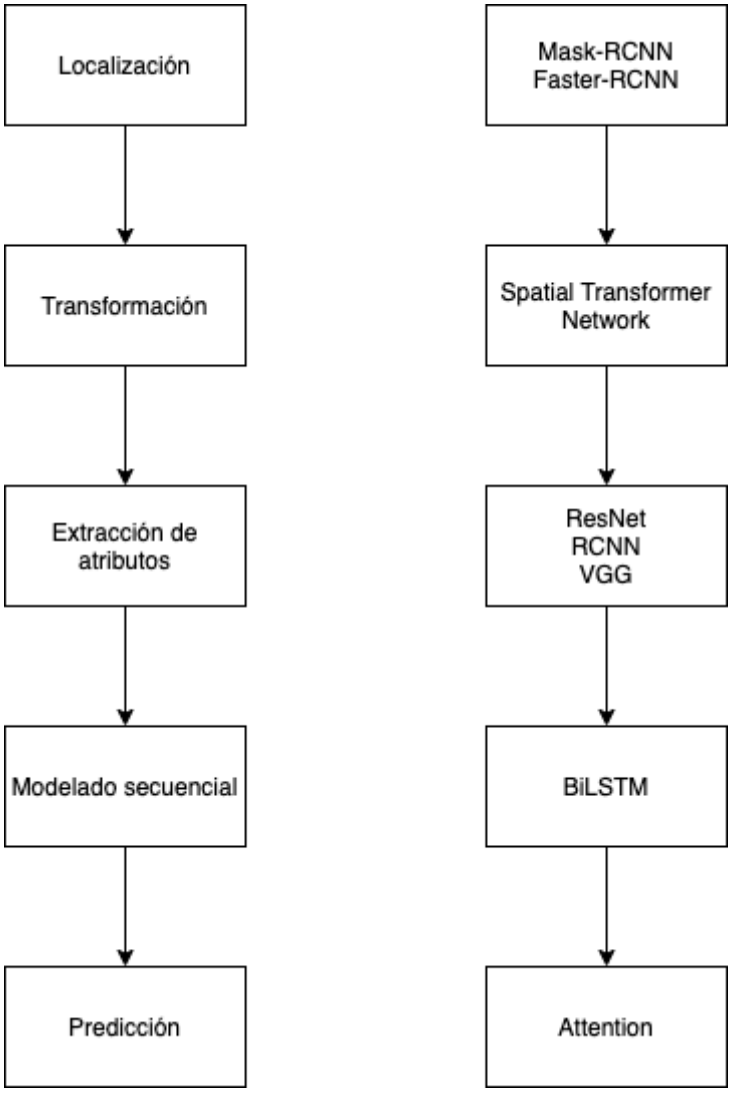
Resultados

4.1. Consideraciones generales

4.1.1. Revisión del proceso

El flujo de trabajo para segmentar objetos -palabras- y reconocer caracteres constó de cinco etapas, siendo la primera dedicada a localización y las cuatro siguientes a reconocimiento. La lógica se puede observar en la figura 4.1 y se basa en lo propuesto en Lee y col., 2021 tanto a nivel de pasos como de arquitecturas. En el lado izquierdo del diagrama 4.1 se observan los pasos desde el punto de vista conceptual y del derecho sus arquitecturas equivalentes. A continuación se describen los cinco pasos y cómo se vinculan los conceptos y sus respectivas arquitecturas.

Figura 4.1: Diagrama del proceso - Pasos y arquitecturas



1. **Localización:** la segmentación de la imagen para detectar palabras se hizo con dos arquitecturas para comparar resultados: Mask-RCNN y Faster-RCNN.

2. **Transformación:** una vez segmentada la imagen se buscó transformar cada polígono regular donde se detectó una palabra para facilitar la

extracción de atributos. Para ello se utilizó una Spatial Transformer Network.

3. **Extracción de atributos:** para la extracción de atributos -esto es, características destacadas de las formas de las palabras- se utilizaron tres arquitecturas para comparar entre sí: ResNet, RCNN y VGG.

4. **Modelado secuencial:** dada la relevancia de la secuencialidad en el texto, posteriormente a la extracción de atributos se aplica una transformación mediante una red recurrente bidireccional de tipo LSTM (*Long-Short Term Memory*) como la utilizada en Chung y Delteil, 2019, que toma la información visual y la convierte en información contextual.

5. **Predicción:** si bien existen varias alternativas para construir la salida respecto de los datos originales, en este caso se utiliza la pérdida de entropía cruzada (*Cross Entropy Loss*) proveniente de una red con atención (Vaswani y col., 2017) y dejar, en caso necesario, la pérdida de Connectionist Temporal Classification usada en otros trabajos similares.

4.1.2. Métricas de evaluación

Para evaluar el desempeño de las técnicas de localización de texto en las imágenes se hizo uso de la pérdida total, esto es, la suma de los términos habituales de pérdida que contempla la implementación de detección y localización de Wu y col., 2019. Las cinco métricas a evaluar (resaltadas en rojo en la figura 2.4) como parte de la pérdida total se dividen en tres tipos y están vinculadas a dos momentos distintos de la arquitectura de segmentación. Las primeras dos pueden observarse en detalle en Girshick, 2015 y la cuarta y quinta en Ren y col., 2015. Para la tercera métrica, sólo aplicada

en las arquitecturas de segmentación *Mask-RCNN*, se puede encontrar más información en He y col., 2017.

- *loss_cls:* pérdida de clasificación para la clase u,

$$L(p, u, t^u, v) = L_{\text{cls}}(p, u) + \lambda[u \geq 1]L_{\text{loc}}(t^u, v),$$

  donde $L_{\text{cls}}(p, u) = -\log p_u$, p es la distribución de probabilidades sobre la cantidad total de clases, u la clase real, t las tuplas de valores para predichas para los cuatro puntos clave de un polígono (x e y de un centroide, así como el ancho y la altura) y v las tuplas reales

- *loss_box_reg:* pérdida de regresión de los puntos críticos del polígono (*bounding box*),

$$L_{\text{loc}}(t^u, v) = \sum_{i \in \{\text{x,y,w,h}\}} \text{smooth}_{L_1}(t_i^u - v_i),$$

- *loss_mask:* pérdida de segmentación de objetos - sólo para Mask-RCNN - esto es,el promedio de la pérdida de la entropía cruzada (que combina en una sola clase la función $LogSoftmax(x_i)$ -definida como $log(\Sigma_j exp(x_j)exp(x_i))$- con la pérdida de *Negative Log Likelihood*) para regiones de interés identificadas

- *loss_rpn_cls:* pérdida de clasificación con entropía cruzada que evalúa si existe o no un objeto en la *Region Proposal Network* (RPN)

- *loss_rpn_loc:* pérdida de regresión con regularización LASSO (L1) cuando existe un objeto en la *Region Proposal Network* (RPN) y se evalúa su localización, siendo la función que comprende ambas pérdidas de RPN

$$L(\{p_i\}, \{t_i\}) = \frac{1}{N_{\text{cls}}} \sum_i L_{\text{cls}}(p_i, p_i^*) + \lambda \frac{1}{N_{\text{reg}}} \sum_i p_i^* L_{\text{reg}}(t_i, t_i^*)$$

Como señalamos anteriormente, se evaluaron dos opciones para la métrica de evaluación del modelo de reconocimiento. En primer lugar, la pérdida de la clasificación mediante *Connectionist Temporal Classification (CTC)* (Graves y col., 2006), que calcula la diferencia entre una serie continua -no segmentada- y una secuencia objetivo, integrando la probabilidad de posibles alineamientos entre el dato de entrada y la secuencia objetivo. Esto produce un valor derivable respecto de cada observación de entrada. En el presente trabajo, tal como se mencionó anteriormente, se decidió privilegiar la predicción con un módulo de atención y su evaluación mediante la entropía cruzada, que se desarrolla a continuación.

Como señala la documentación de implementación de PyTorch (Paszke y col., 2019), el criterio de evaluación por entropía cruzada combina en una sola clase la función $LogSoftmax(x_i)$ -definida como $log(\Sigma_j exp(x_j) exp(x_i))$- con la pérdida de *Negative Log Likelihood*. En este caso, la pérdida se calcula para cada clase y se promedia para cada conjunto de observaciones (o *minibatch* según la denominación habitual en la literatura de aprendizaje profundo), pudiendo asignar además ponderaciones a dicha operación.

Respecto de técnicas de evaluación del desempeño del modelo a lo largo de las iteraciones (*epochs*), si bien utilizar *Early Stopping* para identificar una meseta en la reducción del es una de las alternativas más populares, en este trabajo se decidió extender el entrenamiento de acuerdo a un valor fijo. En el caso de la segmentación de página, el valor se fijó en 10 mil iteraciones. Para el reconocimiento de caracteres, habida cuenta de la complejidad computacional del proceso, se definieron tres valores: 25, 100 y 250 *epochs*. Como se podrá observar más adelante, en los dos primeros se percibe la posibilidad de mejorar el desempeño, mientras que en el tercero los resultados parecen amesetarse alrededor de 50 iteraciones antes de llegar al final.

4.2. Segmentación de la página

En la detección de palabras en la página -esto es, la predicción de polígonos dentro de los que existe texto escrito a mano- se aplicaron técnicas de aprendizaje por transferencia. Para ello, se utilizaron los modelos pre-entrenados desarrollados por *Faceboook AI Research* y publicados como *Model Zoo* en la librería *Detectron 2* (Wu y col., 2019). Todos los modelos allí disponibles fueron pre-entrenados con el conjunto de datos COCO (Common Objects in Context, Veit y col., 2016) y permite reducir de manera considerable el esfuerzo de entrenamiento, ya que sólo requiere el ajuste de las capas superiores de la red neuronal convolucional.

Como se mencionó en 2.2.2, los dos modelos utilizados fueron Mask-RCNN y Faster-RCNN. En ambos casos se tomó como base la implementación de PyTorch, que presenta un *backbone* de ResNet 50 (He y col., 2015) con convoluciones estándar para el enmascaramiento y capas superiores densamente conectadas para predicción y una lógica de *Feature Pyramid Network* (FPN) para extracción de atributos. El desarrollo teórico y las referencias a la literatura de cada componente del modelo puede hallarse en el apartado correspondiente (2.2.2).

Para cada arquitectura se entrenaron -es decir, se realizó el *fine tuning*- dos versiones, una únicamente con caracteres alfanuméricos y otra, con la aclaración *Math* en la tabla 4.3, que también detecta y clasifica regiones con caracteres matemáticos, utilizando el conjunto de datos GNHK. Para ello, se tomaron como base los dos modelos pre-entrenados de *COCO_R_50_FPN_1x* disponible en el *Model Zoo* mencionado y se modificaron los siguientes hiperparámetros para la superior, aquella correspondiente a la predicción de regiones de interés (ROI):

- Tasa de aprendizaje (*Learning Rate*): variando de 0,0001 a 0,0005, encontrando un valor óptimo en 0,00025
- Cantidad de imágenes por lote (*Batch Size*): variando de 8 a 32, encontrando un valor óptimo en 16

4.2.1. Análisis cuantitativo

En este apartado se presentan los resultados para las dos arquitecturas analizadas (Faster-RCNN y Mask-RCNN), detallando la métrica en (*test*) y el tiempo requerido para entrenar cada modelo con 100 iteraciones (*epochs*). Ambas arquitecturas se presentan en dos versiones: una que detecta segmentos con caracteres alfanuméricos y otra que, adicionalmente, identifica regiones con caracteres matemáticos -definidos en el conjunto de datos con un identificador especial.

Como se mencionó anteriormente, los modelos de detección y clasificación utilizados ya cuentan con una base convolucional pre-entrenada con el conjunto de datos COCO (Veit y col., 2016). Sobre eso, se entrenaron las capas superiores utilizando con instancias de 64 GiB de memoria, 16 vCPU y una unidad de procesamiento gráfico (GPU) NVIDIA T4 Tensor Core. En cada caso se definieron diez mil iteraciones, con una tasa de aprendizaje de 0,00025.

Como se puede observar en el cuadro 4.3 que se presenta a continuación, los tiempos de entrenamiento son similares en los cuatro modelos. En los casos de Faster-RCNN el error o pérdida total es más bajo que en el de Mask-RCNN, pero eso puede estar asociado al hecho de que la métrica de evaluación cuenta con un término más -loss_mask- para dicha arquitectura, como se detalló en 4.1.2. Lo mismo aplica para las arquitecturas que

incorporan los caracteres matemáticos para localizar.

Para todos los gráficos de localización de polígonos, el eje Y, definido como *Métrica*, representa la pérdida total definida previamente.

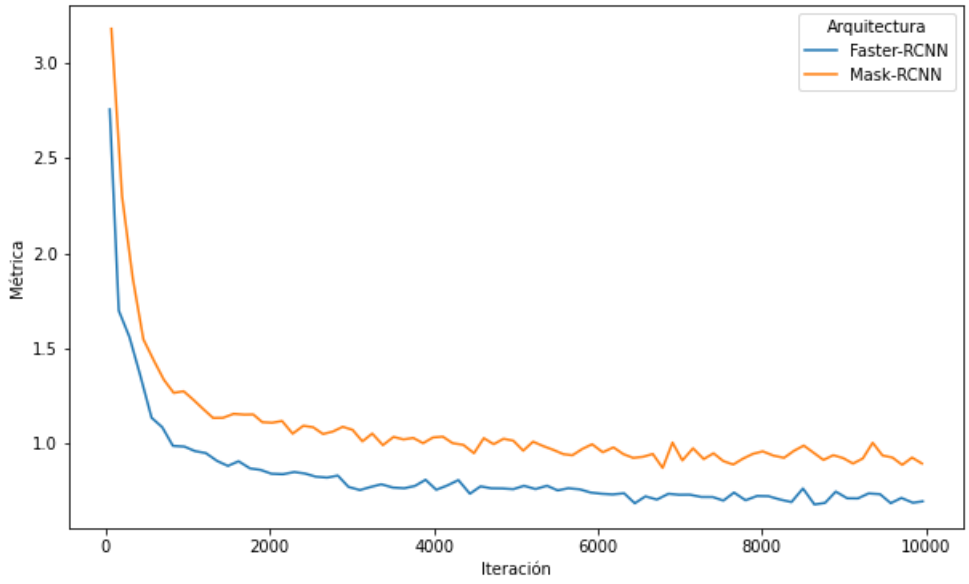
Cuadro 4.1: Desempeño según técnica de localización

Arquitectura	**Pérdida total**	**Pérdida de máscara**	**Duración**
Faster-RCNN	0,6908	-	1h:24m
Mask-RCNN	0,8970	0,2292	1h:32m

Si evaluamos los términos equivalentes de cada modelo -esto es, las pérdidas del regresión y clasificación para las dos salidas del modelo de localización- el resultado da un desempeño similar, con una leve ventaja para la Mask-RCNN. Esto puede corroborarse mediante los valores de la columna *Pérdida de máscara* del cuadro 4.3, si se tiene en cuenta que la pérdida total es la suma de todas las métricas detalladas y, en el caso de Mask-RCNN, al sustraer la pérdida de máscara el valor de la pérdida total quedaría en 0,6678.

Con relación a la evolución en el tiempo de las métricas, el descenso más significativo ocurre en las primeras dos mil iteraciones, donde se lleva a un valor cercano a 1,2 de pérdida total. El descenso del error continúa hasta la iteración 7 mil, donde parece amesetarse hasta el final del entrenamiento.

Figura 4.2: Desempeño según arquitectura de localización



Al incorporar la detección de polígonos donde pueden existir caracteres matemáticos, el desempeño es el siguiente:

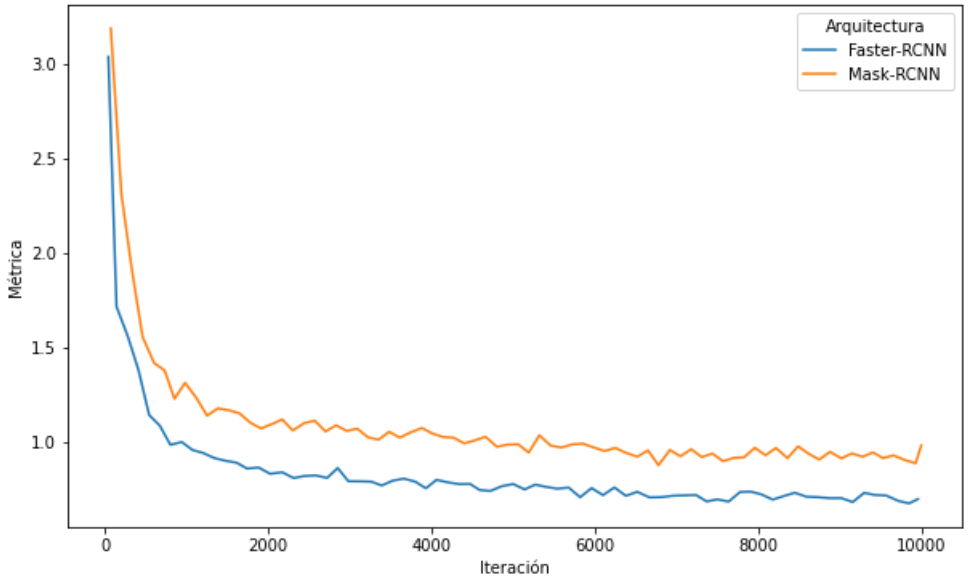
Cuadro 4.2: Desempeño según técnica de localización

Arquitectura	**Pérdida total**	**Pérdida de máscara**	**Duración**
Faster-RCNN *(Math)*	0,7019	-	1h:24m
Mask-RCNN *(Math)*	0,883	0,2137	1h:26m

Tal como se mencionó previamente, los tiempos de entrenamiento y el desempeño general es comparable con lo analizado para los modelos que no detectan regiones con caracteres matemáticos. Esto está fuertemente asociado con el hecho de que los modelos no cambian de arquitectura ni de base convolucional pre-entrenada, sino únicamente incrementan la cantidad de

clases -objetos- a detectar y clasificar.

Figura 4.3: Desempeño según arquitectura de localización- *(Math)*



Por último, evaluaremos las métricas disponibles en el trabajo de Lee y col., 2021 referidas al desempeño en la detección de polígonos. Allí se describen los valores de *Recall*, *Precision* y *F1-Score* -media armónica- para las arquitecturas bajo estudio. Para que el análisis sea comparable, el valor de *Intersection Over Union* (IOU), esto es, el valor relativo de superposición del área de dos polígonos a partir del que se considera que son en realidad el mismo, fue también de 0,5.

Como se puede apreciar, los resultados son similares, detectando una mejora en el recall a a costa de una menor precisión, es decir, una mayor cantidad de falsos positivos predichos respecto de lo que se observa en Lee y col., 2021.

Cuadro 4.3: Desempeño respecto de métricas disponibles en Lee y col., 2021

Arquitectura	**Recall**	**Precision**	**F1-Score**
Faster-RCNN	0,8522	8945	0,86494
Mask-RCNN	0,8507	0,8957	0,86499
Faster-RCNN (Lee y col., 2021)	0,8077	0,9215	0,860
Mask-RCNN (Lee y col., 2021)	0,8237	0,9079	0,864

4.2.2. Análisis cualitativo

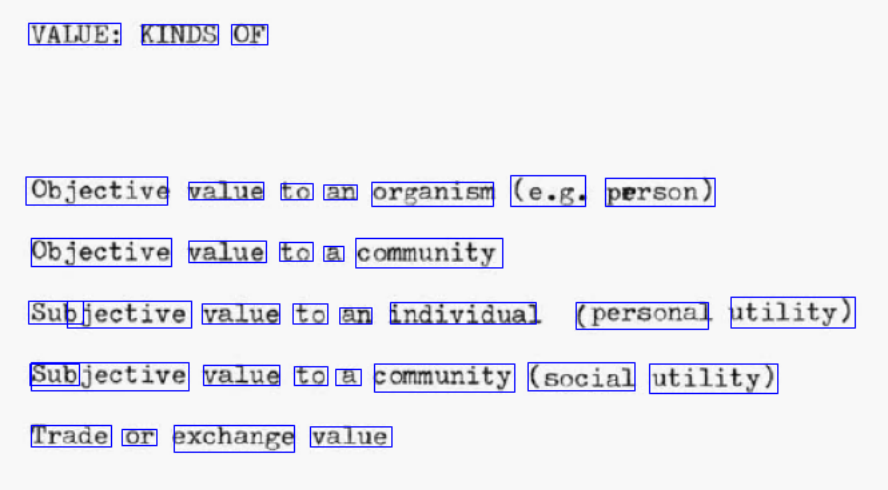
Para el análisis cualitativo de los modelos de segmentación se grafican los polígonos predichos en cinco textos seleccionados del acervo de Mario Bunge disponible en la Biblioteca Digital de la Facultad -una nota corta escrita a máquina, una nota larga manuscrita, una carta recibida manuscrita, una carta recibida manuscrita con membrete y una página completa correspondiente también a una carta de Guido Beck. Cada uno de estos textos tiene una característica distintiva que los hace interesantes para observar el desempeño cualitativo. Como modelo de base para obtener las predicciones se utilizó el Mask-RCNN analizado en el apartado anterior.

En este apartado buscaremos identificar algunos aspectos a nuestro juicio relevantes que pueden mostrar, sujeto a interpretación, algunos comportamientos particulares del modelo de segmentación bajo estudio. A diferencia de lo ocurrido con el análisis cuantitativo, en este caso analizaremos una muestra de imágenes de un conjunto de datos no etiquetado, por lo que no se calcularán las métricas del apartado previo (Recall, Precision y F1-Score) y, cuando se analicen valores, como en la figura 4.5.

El primer documento a analizar es la tarjeta de apuntes corta escrita a máquina. Cada término es identificado de manera correcta, logrando una

cobertura de cada una de las palabras. Cada término, además, es identificado con un polígono que se ajusta casi perfectamente al tamaño mínimo posible de la palabra, esto es, sin espacio en blanco. Sin embargo, en algunos casos las palabras se superponen y se identifican dos términos como si fueran palabras separadas (el *Sub* dentro de *Subjective*).En este sentido, pareciera que el modelo de localización es exhaustivo pero no excluyente.

Figura 4.4: Nota corta - Mario Bunge



Es importante recordar, respecto de los polígonos, que uno de los aportes al acervo público es la entrega sistematizada de los puntos críticos (vértices) para cada término identificado en cada documento de Mario Bunge recolectado del sitio web de la Bibloteca. En el cuadro siguiente se observa un extracto de lo que se disponibiliza, en formato csv, para los alrededor de 16 mil polígonos identificados.

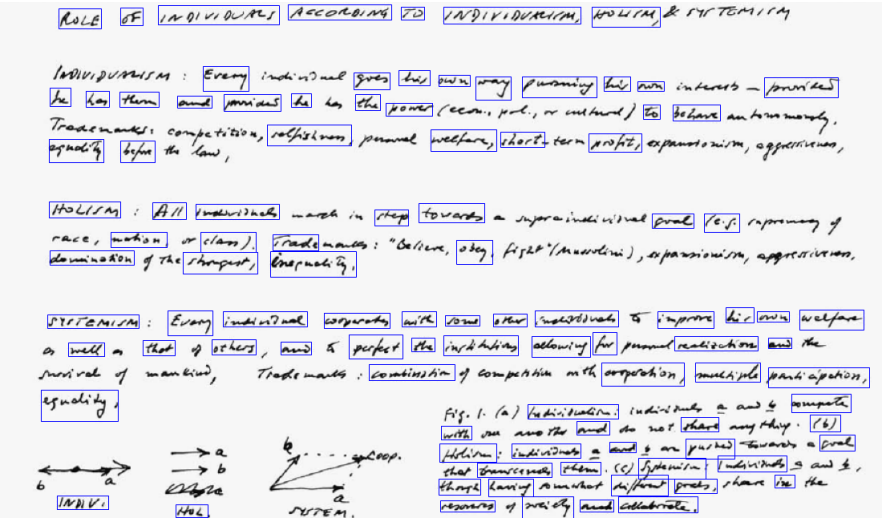
En la primera nota manuscrita (figura 4.5) que pareciera ser de apuntes sobre tres abordajes filosóficos (individualismo, holismo y sistemismo) se observa que el modelo capta la mayoría de los términos, aunque un porcentaje considerable -alrededor del 45 %- no es evaluado como términos.

Cuadro 4.4: Coordenadas de polígonos de los documentos de la Biblioteca Digital

Documento	x1	x2	y1	y2
335_2	642	661	68	91
335_2	248	269	32	55

Esto puede tener dos explicaciones posibles: en primer lugar, el tamaño de la imagen, que es inversamente proporcional a la cantidad de polígonos que se pueden detectar; mientras más extenso el texto más desafiante para el modelo detectar todos los términos, a mismo tamaño de entrada de imagen. En segundo, el umbral de detección de objetos (*objectness*) del modelo, que busca mantener la parsimonia entre la exhaustividad y la exclusividad de la detección de términos, para evitar en lo posible lo que se vio en la nota anterior con el *Sub* de la palabra *Subjective*. Dicho umbral está asociado al concepto previamente descripto de IOU.

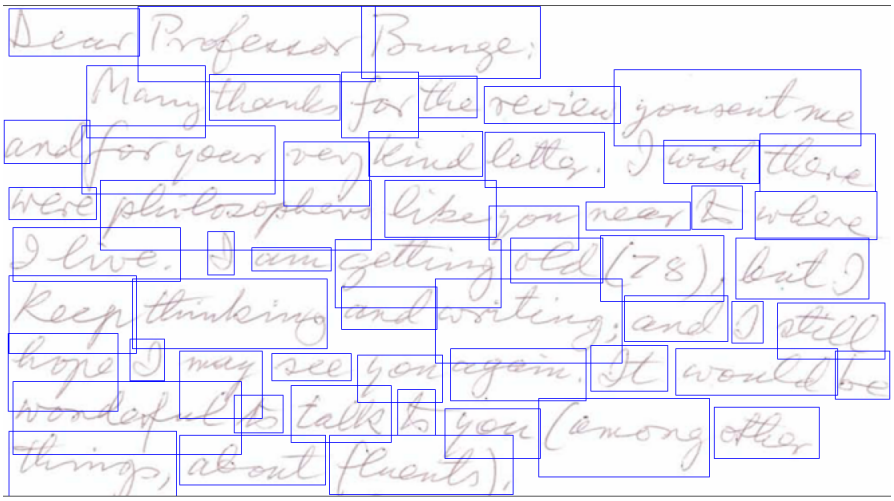
Figura 4.5: Nota larga - Mario Bunge



A diferencia de lo que ocurre en la figura 4.5, la figura 4.6 pareciera ser el

documento donde mejor se identifican los *bounding boxes* de cada término, al clasificar correctamente todos excepto uno (I), incluyendo aquellos con una superposición significativa. Un fenómeno que se aprecia, muy probablemente debido a la grafía, es la dificultad para separar palabras escritas con algún tipo de ligazón, como en el caso de *you sent me* en la segunda línea de texto. Otro aspecto interesante -y que forma parte de los objetivos buscados en un modelo de segmentación de este tipo- es la capacidad de adaptar la detección de palabras a la altura de las mismas, aún si el texto no está perfectamente alineado.

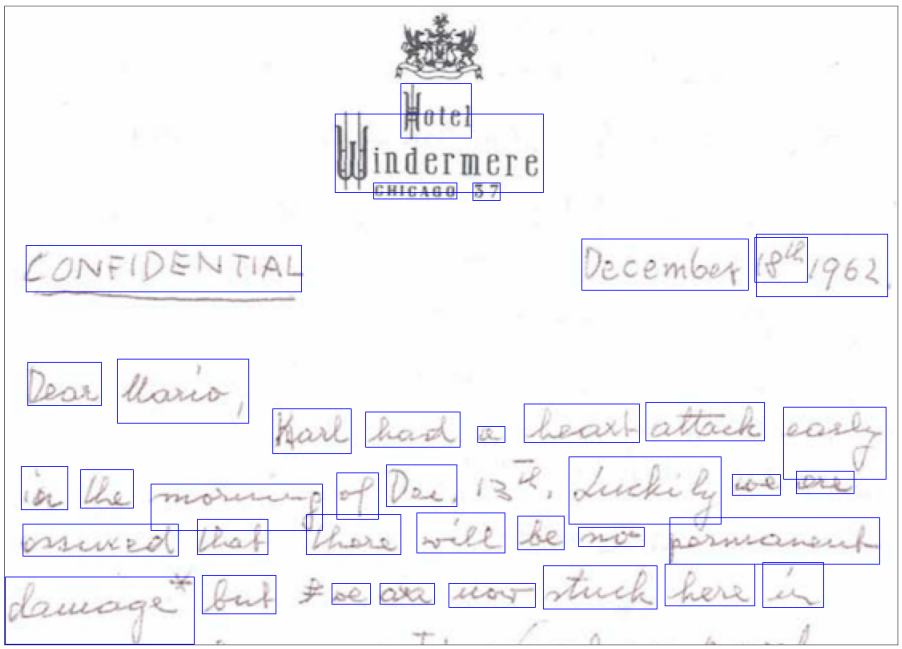
Figura 4.6: Correspondencia - Mario Bunge



La nota con membrete de la figura 4.7, que cuenta con una grafía similar a la anterior, es interesante para observar hasta dónde términos vagamente identificables son considerados caracteres pasibles de ser reconocidos, como el escudo del hotel y la primera letra de ambas palabras del nombre, *Hotel* y *Windermere*. Sobre este punto hay que destacar que para el *fine-tuning* de los modelos de Mask-RCNN y Faster-RCNN incluye la identificación de zonas que en el conjunto de entrenamiento están etiquetadas como garabatos o *scribble*. Tal como observamos en la nota de la figura 4.6, el modelo detecta

correctamente términos con polígonos que se solapan (tales como *Chicago*) aún cuando el *Intersection Over Union* (IoU) es alto.

Figura 4.7: Nota con membrete - Mario Bunge



Al igual que en los casos anteriores, cada término pareciera contar con un polígono ajustado correctamente a su tamaño y, así como en la figura 4.5, el modelo identifica correctamente aquello que es texto y aquello que no lo es (como las llamadas en los términos *damage* y *but*).

Figura 4.8: Correspondencia - Página completa

22290 - Rio de Janeiro, 22 de Julio de 1983
c/o CLAP - Av. Venveslau Braz 71 fundos)

Querido Pauchito,

Muchas gracias por sus noticias y felicitaciones por el Principe de Asturias.

Siempre me alegro oir de Ud. y de ver como Ud. funciona. Y me divierto enterarme que Ud. discute con el papa y con un señor Quintanilla, directamente de Salamanca. (Cuidado, como ya dice el Pibe: allá no "prestan"!).

No puedo conseguir aquí los números correspondientes de la INTERCIENCIA. Tiene Ud. una fotocopia del discurso del Papa y una de su carta a INTERCIENCIA?

Está bien como Ud. se mete. Pero, por favor, no se haga ilusiones. Lo que se puede lograr es mucho menos de lo que uno se imagina y querría lograr. Lo que yo podía hacer aquí era solamente formar alguna gente. Eso seguramente era necessario, pero de lejos no era suficiente. Mire lo que quedó en la Argentina. Casi nada. Aquí parece más, pero, seguramente será solamente una fracción que sobreviverá. En un caso assim hay mucha gente que se mete. Los del gobierno no entienden nada porque no hay tradición, los chicos buenos exageran porque no ven todas las dificultades y los chicos malos tratan de hacer negocios. Además: en el mundo grande las cosas tampoco andan bien y lo que hacen ahora en la física teórica (no en la experimental) está lejos de la calidad que el trabajo tuvo en mi tiempo. Un ejemplo: Hace 20 años que se tiene el laser y ya tiene muchísimas aplicaciones. Hicieron grandes teorias pero nunca entendieron siquiera el origen y el caracter del fenomeno. Ahora vi que sale directamente de "first principles" con todas las propriedades caracteristicas.

Bueno: que tenga mucho exito y muchas felicidades

Guido Beck

GBB - Gran Feria Blanco

La correspondencia completa (figura 4.8) muestra el impacto negativo de dar como insumo al modelo una página de texto completa (sin separarla, por ejemplo, en subimágenes de 5 a 10 renglones), lo que reduce la capacidad de predecir correctamente los polígonos a igual tamaño de imagen de entrada. A pesar de esto -y de la baja eficacia del modelo- se conserva la capacidad de reconocer términos con polígonos superpuestos, como se evidencia en aquellos debajo de la firma de Guido Beck, aunque uno de ellos de manera incompleta. Habida cuenta de las claras limitaciones de este tipo de modelos de localización para reconocer los polígonos en texto sin segmentar (esto es, páginas de tamaño A4, por ejemplo, completas) es razonable pensar

que ese tipo de procesamientos funcionarían mejor con abordajes de OCR tradicionales, pensados para resolver esos problemas.

Es importante volver destacar, a este respecto, que ambos modelos -el de segmentación y el de reconocimiento de caracteres, fueron entrenados como parte de un mismo proceso y con el mismo conjunto de datos, que consiste en notas en inglés. Si bien esto afecta la capacidad de reconocer de manera adecuada los caracteres (el tópico que se aborda en el apartado subsiguiente), no debiera tener impacto en la segmentación de la imagen, por lo que para la figura 4.8 los argumentos detallados anteriormente son los más plausibles.

4.3. Reconocimiento de caracteres

Posteriormente a la detección y clasificación de los objetos en nuestras imágenes es necesario reconocer los caracteres. Esto implica no sólo identificar cada una de las letras de las palabras sino cómo se vinculan entre sí, es decir, cómo se convierten caracteres en términos con un sentido determinado. Con el objetivo de realizar eso y basado en estudios recientes que han abordado la temática (Chung & Delteil, 2019; Lee y col., 2021), para este paso se hacen uso de un conjunto de técnicas, introducidas en el apartado 4.1.1 y, de forma visual, en la figura 4.1.

El primer paso, como mencionamos, se realiza mediante la transformación espacial, que toma la imagen del polígono identificado en la segmentación y modifica su tamaño. Para el proceso de rectificación y regularización detallado en el apartado 2.2.3 se definieron 20 puntos de referencia o anclajes, siguiendo la literatura (Lee y col., 2021).

La tarea de reconocimiento de caracteres se realizará comparando el desem-

peño de 3 arquitecturas para detección de atributos: las redes de tipo VGG, uno de los primeros desarrollos de redes profundas para reconocimiento a escala en imágenes (Shi y col., 2015), las redes basadas en regiones (Wang & Hu, 2021) -ya utilizadas para la detección y clasificación de palabras- y una red de conexiones residuales denominada ResNet (He y col., 2015) que se ha posicionado como una de las mejores alternativas para estas tareas. Para los tres casos se definieron como valores máximos a analizar 512 capas ocultas, considerando la capacidad de cómputo existente. Para cada una se analizará su desempeño primero con 25 iteraciones y luego con 100, con una tasa de aprendizaje de 0,001 y 64 observaciones por cada lote (*batch*) a analizar. Como se verá más adelante, cada uno de los hiperparámetros mencionados fue definido de acuerdo a criterios de desempeño de la métrica de evaluación (entropía cruzada).

La salida del modelo de extracción de atributos actúa como insumo para un modelado secuencial con una red recurrente bidireccional (detallada en 2.2.4). En este caso se utilizaron 256 capas ocultas y se definió, de acuerdo a la literatura (Lee y col., 2021) un *gradient clipping* -esto es, el valor que limita la variación del gradiente y resuelve problemas de *exploding gradients* detallados en Goodfellow y col., 2016- de 5.

Como paso final, las predicciones se realizan tomando los atributos contextuales del modelado secuencial como entrada en un módulo de atención que tiene, al igual que la red bidireccional, 256 capas ocultas. La salida del módulo de atención (Cheng y col., 2017) es una probabilidad para cada caracter con una matriz de tamaño equivalente al del alfabeto definido de 72 elementos, con una longitud máxima para la atención de 51 observaciones. Los valores del optimizador utilizados, un Stochastic Gradient Descent (Goodfellow y col., 2016) con 0,9 de *momentum* y 0,0005 de *weight decay* se definieron de acuerdo a la literatura (Lee y col., 2021).

A continuación se detallan las modificaciones realizadas a lo largo del entrenamiento para cada uno de los hiperparámetros mencionados en el presente apartado. A diferencia de lo ocurrido con el modelo de segmentación, no se realizó *fine tuning* sobre un modelo pre-entrenado sino que se entrenó desde cero cada modelo y se modificaron los siguientes hiperparámetros hasta encontrar los valores óptimos:

- Tasa de aprendizaje (*Learning Rate*): variando de 0,0005 a 0,005, encontrando un valor óptimo en 0,001
- Cantidad de imágenes por lote (*Batch Size*): variando de 16 a 128, encontrando un valor óptimo en 64
- *Gradient clipping*: variando de 3 a 5, sin mejorar significativas y tomando el valor de 5 de acuerdo a la literatura citada previamente.
- Capas ocultas para la red bidireccional: variando de 128 a 512, encontrando el valor óptimo en 512 pero, por limitaciones en el cómputo en 250 iteraciones, definiendo el valor final en 256
- Capas ocultas para el mecanismo de atención: variando de 128 a 512, encontrando el valor óptimo en 256

La cantidad de caracteres a ser predichos se obtiene de la suma de los caracteres alfabéticos, numéricos y caracteres especiales, tanto del texto como los requeridos para la clasificación el *padding* -entre ellos, el correspondiente a caracteres matemáticos. Finalmente, se evalúa el desempeño de la salida mediante el cálculo de la entropía cruzada.

4.3.1. Análisis cuantitativo

En este apartado se presentan los resultados para cada una de las arquitecturas analizadas de detección de atributos, detallando la métrica en *test* y el tiempo requerido para entrenar cada modelo, según la cantidad de iteraciones (*epochs*). Es importante resaltar que si bien se considera únicamente el tiempo dedicado al entrenamiento del modelo de reconocimiento de caracteres -esto es, no al *fine-tuning* del modelo de segmentación, ya analizado- este contempla tanto la extracción de atributos con las arquitecturas detalladas como la transformación espacial, el modelado secuencial y la predicción.

En cada caso, los modelos fueron entrenados con instancias con unidades de procesamiento gráfico (GPU) NVIDIA Tesla V100. Para las arquitecturas VGG y RCNN se utilizaron instancias de 244 GiB de memoria, 32 vCPU y 4 GPUs. En el caso de la arquitectura ResNet, debido a su mayor necesidad de cómputo, fue necesario contar con una instancia de 488 GiB de memoria, 64 vCPU y 8 GPUs. Relacionado a esto, dentro del período de entrenamiento no fue posible alcanzar las 100 iteraciones en una primera instancia -debido a limitaciones de infraestructura que colocaban el tiempo máximo en 24 horas, pero luego dicha limitación fue subsanada.

Se presentan a continuación los valores finales y los tiempos de entrenamiento para las tres arquitecturas con 25 iteraciones.

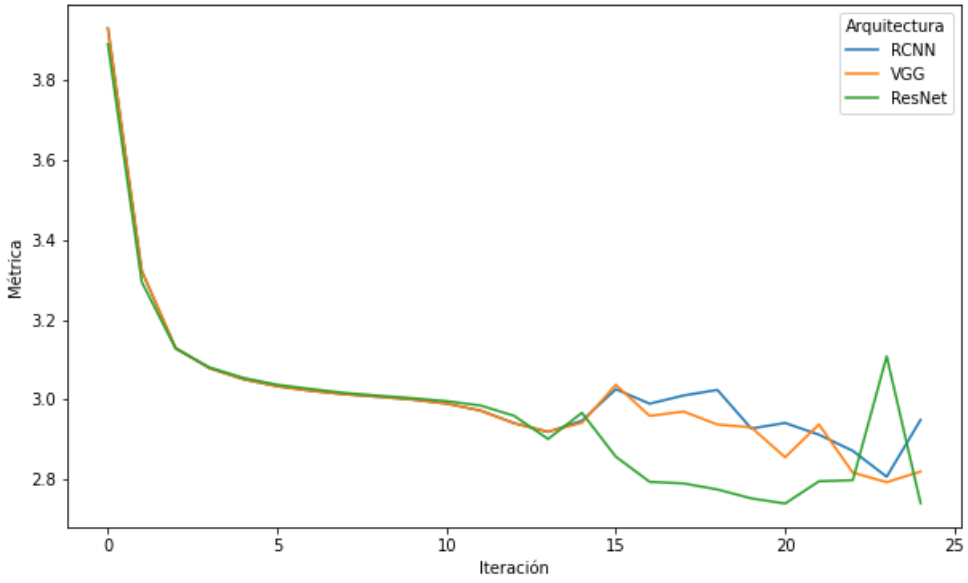
Cuadro 4.5: Desempeño por arquitectura extracción de atributos - 25 *epochs*

Arquitectura	**Entropía cruzada**	**Duración**
ResNet	2,7620	7h:28m
VGG	2,9577	3h:29m
RCNN	2,9536	3h:30m

La evolución del desempeño, al entrenar los modelos con 25 iteraciones, presenta un declive significativo de la pérdida de entropía cruzada en las primeras 5 *epochs* de cerca de un punto, alrededor del 30 %. En las siguientes cinco iteraciones la disminución del error se ameseta y, a partir de la *epoch* 14 el comportamiento diverge: mientras que el modelo ResNet alcanza valores de entropía cruzada más bajos, las otras dos arquitecturas (VGG y RCNN) incrementan levemente el error para luego bajarlo paulatinamente.

En el caso de ResNet, el error presenta una suba excepcional apenas antes del cierre, con una métrica levemente inferior -como se observó en el cuadro 4.5. Se identifican, entonces, dos patrones a priori: un descenso más pronunciado del error de la arquitectura ResNet a partir de la iteración 15 conjuntamente con una menor estabilidad de dicho valor cerca del final del entrenamiento.

Figura 4.9: Desempeño según arquitectura - *(25 epochs)*



A pesar de las oscilaciones -esto es, que no es una curva monotónica-, hubo

un decremento del error que podría haber continuado. Para evaluar si eso es efectivamente así, lo que se hizo fue incrementar las iteraciones, utilizando la misma infraestructura.

Cuadro 4.6: Desempeño por arquitectura de extracción de atributos - 100 *epochs*

Arquitectura	**Entropía cruzada**	**Duración**
ResNet	1,7293	37hs
VGG	2,3881	11h:19m
RCNN	2,2772	19h:31m

Como se puede apreciar, la arquitectura ResNet tuvo un desempeño considerablemente mejor. Por otro lado, no pareciera haber diferencias significativas en los resultados finales entre VGG y RCNN. Sin embargo, el modelo de menor tiempo de entrenamiento requirió para realizar 100 *epochs* -poco más de la mitad que RCNN y alrededor de un tercio de lo que necesitaríamos para ResNet- también fue el que menos desempeño mostró entre ellos doos, con 0,1 puntos por encima de RCNN en la perdida de entropía cruzada.

En el caso de ResNet, que requirió un tiempo de entrenamiento y una capacidad de cómputo considerablemente mayor que las otras arquitecturas, la diferencia en el desempeño es significativa: una reducción de casi 25 % del error en *test*.

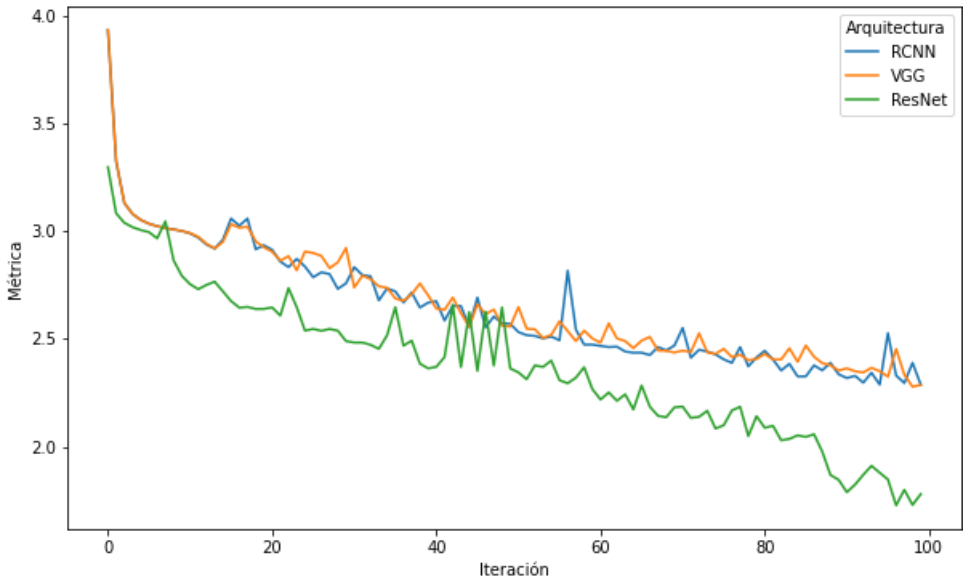
Respecto de la evolución de la métrica de evaluación en el tiempo, se pueden observar dos patrones distintivos. Al igual que cuando se analizaron las 25 iteraciones, las arquitecturas RCNN y VGG presentan una evolución similar, tanto desde el punto de partida como del de llegada. La principal diferencia entre ambas se ve, tal como se mencionó previamente, en la duración del tiempo de entrenamiento, con 11:19 horas en el caso de VGG y 19:31 horas

para la red basada en regiones (RCNN).

La principal similitud entre VGG, RCNN y ResNet se relaciona al descenso casi monotónico de ambas, lo que hace suponer -aunque escapa al alcance de este trabajo- que de continuar el tiempo de entrenamiento con los mismos hiperparámetros -esto es, los valores no directamente entrenables de las redes utilizadas sino a través de un proceso externo como la optimización bayesiana o aleatoria- la métrica de error podría haber continuado su descenso.

La arquitectura ResNet presenta un comportamiento distintivo tanto en el tiempo -y la capacidad de cómputo requerida- como en la evolución de la pérdida de entropía cruzada. Desde el comienzo del proceso, como se observó al analizar el desempeño con 25 *epochs*, muestra una pendiente mayor que RCNN y VGG, esto es, reduce a mayor velocidad el error.

Figura 4.10: Desempeño según arquitectura - *(100 epochs)*



Un fenómeno que se observa a partir de la iteración 40 el valor empieza a oscilar -con picos de error pronunciados- y es sólo a partir de la iteración 55 que se estabiliza y comienza una paulatina mejora del desempeño hasta cerca del final del entrenamiento. Sobre este aspecto no contamos con una comprensión en profundidad de lo que ocurrió, pero algunas hipótesis posibles están vinculadas a fenómenos como el de los *exploding gradients* de las redes recurrentes utilizadas. Una manera de evaluar si ese puede haber sido efectivamente el problema es reducir el valor de *gradient clipping* (definido en la sección 4.3), pero escapa a los objetivos del presente trabajo. A partir de la iteración 85 el entrenamiento incrementa nuevamente la tasa de descenso del error, divergiendo de las otras dos arquitecturas.

Luego de evaluar las tres arquitecturas con 100 epochs se realizó una última prueba para entender, a mismos parámetros, el margen de mejora que aún tenía cada una. Así, se definieron 250 iteraciones y se obtuvieron los siguientes resultados:

Cuadro 4.7: Desempeño por arquitectura de extracción de atributos - 250 *epochs*

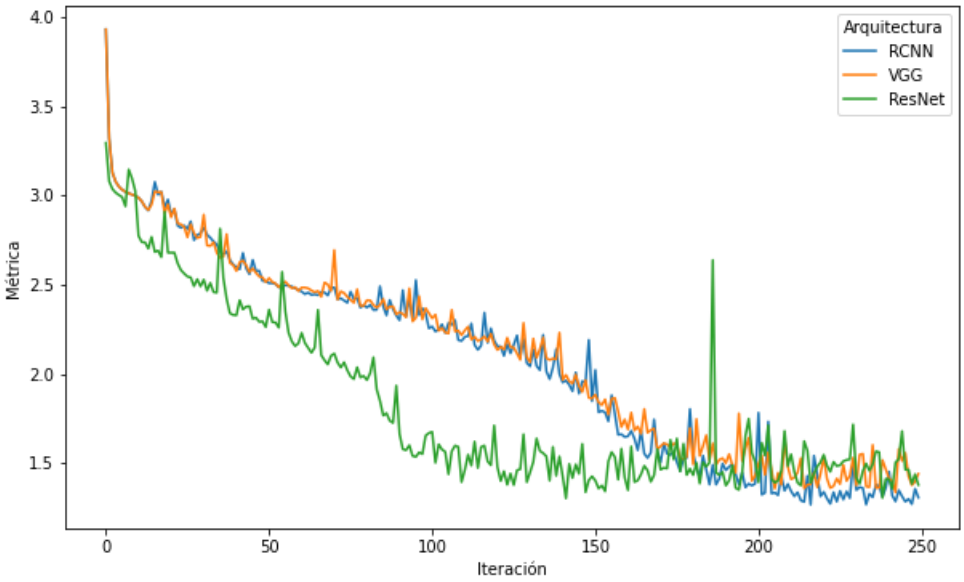
Arquitectura	**Entropía cruzada**	**Duración**
ResNet	1,2968	3d:4h:27m
VGG	1,4517	2d:7h:3m
RCNN	1,3354	3d:4h:50m

Como se puede apreciar en la imagen 4.11, el modelo ResNet converge más rápidamente hacia su cota inferior de métrica de pérdida en *test*, pero luego de hacerlo no presenta ninguna mejora signficativa (aunque sí marginal) por las siguientes 100 iteraciones. En caso de que el tipo de entrenamiento no fuera para evaluar el desempeño en el tiempo sino para encontrar el mejor

modelo, sería esperable que este comportamiento limite el impacto en tiempo y costos aplicando *Early Stopping*.

En el caso de las arquitecturas RCNN y VGG vemos que su convergencia es muy similar en las primeras 100 *epochs* pero que diverge al continuar y vuelve a acercarse alrededor iteración 200, donde la primera encuentra un punto de equilibrio más bajo para su métrica de pérdida. Es interesante señalar que las tres arquitecturas encontraron su métrica de pérdida más baja alrededor de la iteración 200 (en ese valor para RCNN, 206 para ResNet y 213 para VGG).

Figura 4.11: Desempeño según arquitectura - *(250 epochs)*



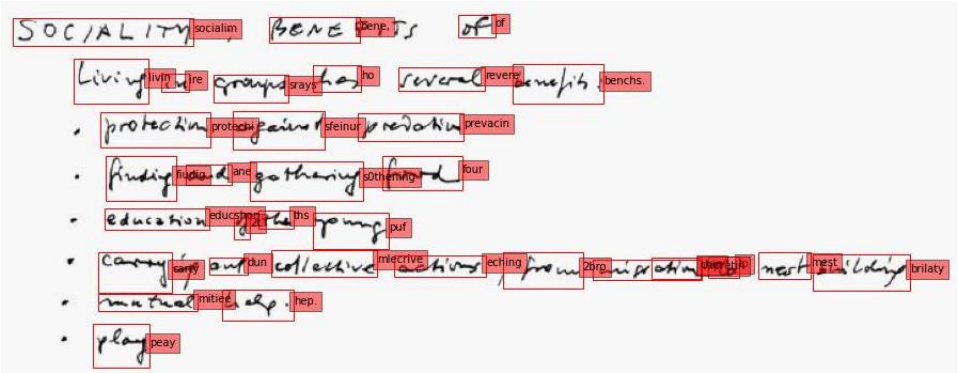
4.3.2. Análisis cualitativo

Además de lo evaluado previamente en términos cuantitativos, y replicando lo realizado para los modelos de segmentación de polígonos de texto en las

imágenes, en el presente apartado analizaremos cualitativamente el reconocimiento de texto. Para ello usaremos la arquitectura Mask-RCNN para la segmentación y la ResNet para extracción de atributos (previo al modelado secuencial y con atención y posterior a la transformación espacial).

En el caso de la nota corta manuscrita de la figura 4.12 se destacan dos aspectos. En primer lugar, las letras mayúsculas del título se reconocen mayormente de manera correcta, identificando el término *Socialim* y *Of*. En el caso de *Benefits*, donde identifica correctamente las primeras cuatro letras, el error surge de una incorrecta segmentación del término y no de una falla en la predicción. En segundo lugar, el texto en cursiva presenta una correcta identificación en muchos casos de la letra inicial y de conjuntos de vocales y consonantes (como *eve* en *several*) pero halla limitaciones considerables en múltiples palabras para identificar la totalidad. El caso de la palabra *protection* es tal vez el que mejor refleja este comportamiento.

Figura 4.12: Nota corta - Mario Bunge



Como se mencionó en el apartado anterior, los datos relevados para cada página de cada documento se entregan de manera sistematizada a la Biblioteca digital de la Facultad de Ciencias Exactas y Naturales de la Universidad de Buenos Aires. En este caso, un desafío a futuro puede ser la aplicación

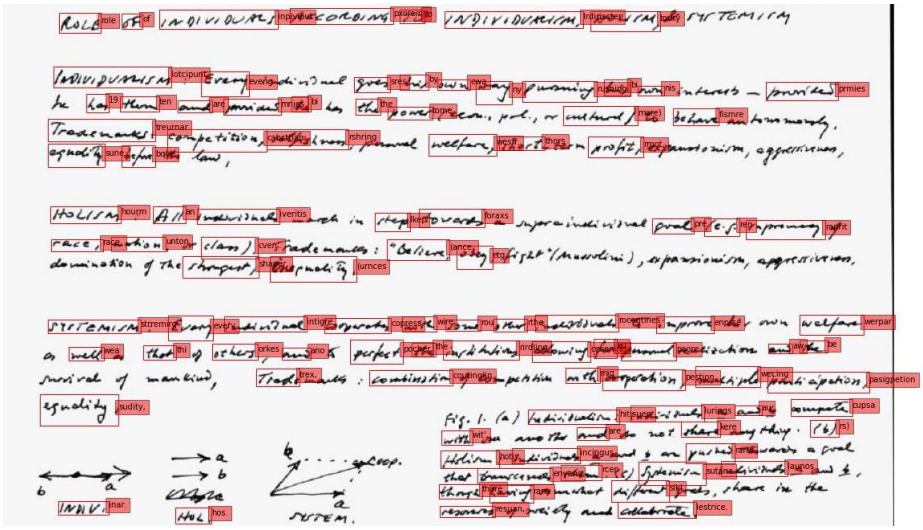
de un *lexicon* o de una técnica de *human-in-the-loop* para refinar la predicción de cada palabra, habida cuenta de que se dispone de los polígonos y el término inferido. Colocar una interfaz de usuario que permita ir corrigiendo aquellas no acertadas puede ser un trabajo interesante que enriquezca aún más la información disponible.

Cuadro 4.8: Polígonos de los documentos de la Biblioteca Digital

Documento	**Término**	**x1**	**x2**	**y1**	**y2**
335_2	socialim	642	661	68	91
335_2	livin	248	269	32	55

Respecto de la nota larga, se aprecia en la figura 4.13 que sobre los polígonos identificados -tal como señalamos anteriormente, el hecho de ingresar una imagen de gran tamaño sin segmentar impacta negativamente la capacidad de predecir todos los polígonos- se clasificó de manera incorrecta la mayoría de las letras. En algunos casos (*Role*, *Of*, *Holism*) se predijo correctamente el término completo, aunque en el último de ellos la grafía generó que la palabra predicha fuera *Housm*. En este sentido cobra importancia el conocimiento de dominio a la hora de refinar las inferencias.

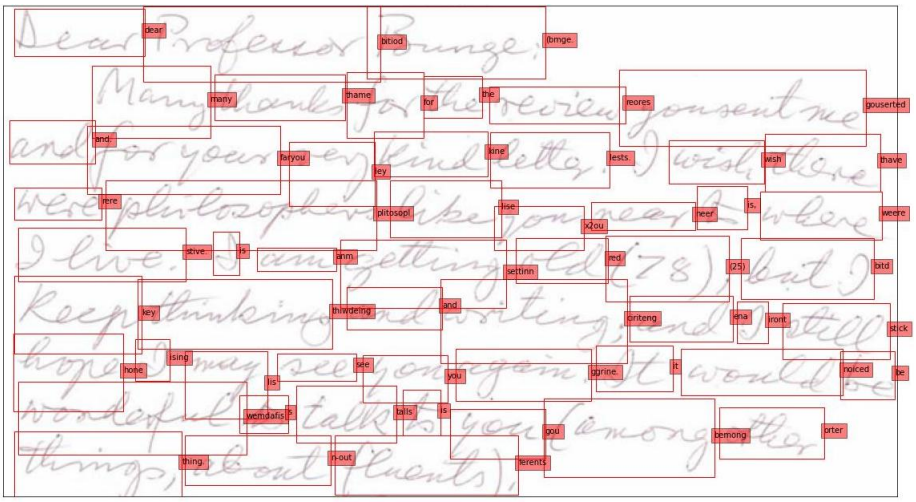
Figura 4.13: Nota larga - Mario Bunge



Esta nota extensa y que combina mayúsculas, minúsculas y un tipo de escritura muy particular es tal vez el caso que mayores desafíos presenta a la hora del reconocimiento.

La correspondencia manuscrita dirigida a Mario Bunge (figura 3.2) tiene características que las destacan de las dos anteriores. En primer lugar,la identificación de los caracteres en este documento representa un avance respecto de la lista anterior y podría ser evidencia de la importancia contextual en caracteres de palabras "habituales"del idioma del conjunto de datos de entrenamiento. Del mismo modo, lo que puede estar siendo identificado correctamente tal vez no sea el idioma sino el tipo de grafía, más similar en este caso al utilizado en el conjunto de datos *GNHK*. Sin embargo, sería necesario un análisis en profundidad del funcionamiento de ambos módulos contextuales -la red recurrente bidireccional y el de atención- para basar fácticamente dicha aseveración en el presente trabajo.

Figura 4.14: Nota - Mario Bunge

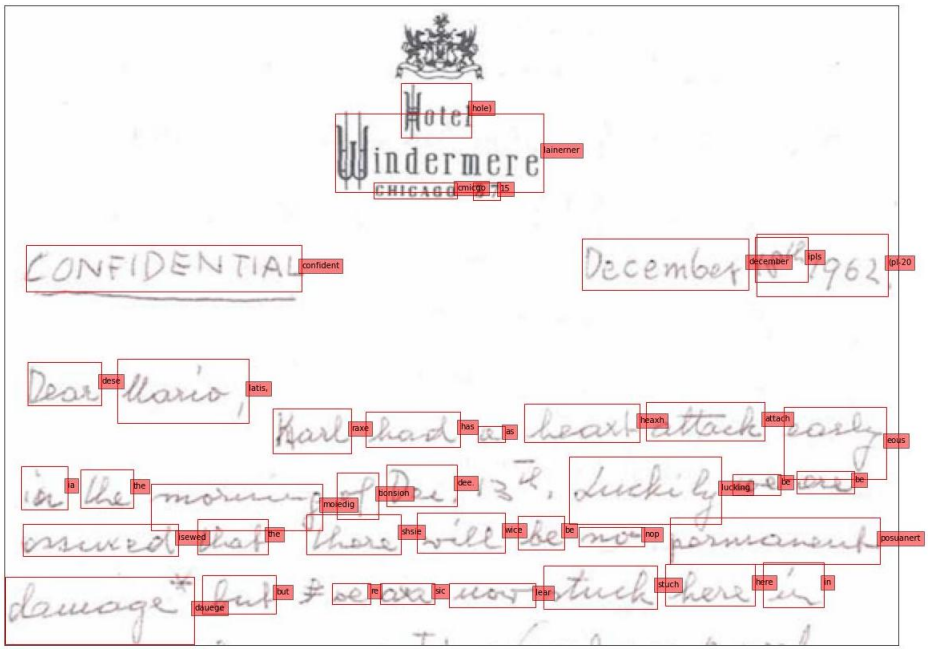


Como última figura a analizar cualitativamente tenemos la nota manuscrita con membrete. Como se mencionó en el apartado anterior, están contemplados como texto dado que se utilizaron los objetos identificados caracteres de garabatos o *scribbles* del conjunto de datos de entrenamiento de la localización. En este caso, el modelo identifica parcialmente las letras del nombre del hotel y la ciudad, principalmente las vocales, y lo mismo hace con la identificación del número posterior a la ciudad del hotel y al año en el que está fechada la carta. Así como en las notas revisadas al comienzo del apartado, las letras en mayúscula se identifican mayormente de forma correcta (en el caso de *Confidential*) y lo mismo ocurre con las vocales. Para algunas consonantes la grafía parece afectar la capacidad de reconocerlas, como la confusión entre *h* y *k* en *attack* y *stuck*.

Si bien en la literatura existen ejemplos de modelos de reconocimientos de caracteres agnósticos al lenguaje -y, en este caso, nuestro modelo cumpliría con lo señalado por Graves y Schmidhuber, 2009- los nombres propios no logran ser clasificados correctamente. Es el caso del nombre del destinatario,

la ciudad y el hotel.

Figura 4.15: Nota 2 - Mario Bunge



Capítulo 5

Conclusiones

5.1. Observaciones generales

La utilización de técnicas de aprendizaje profundo permite reconocer texto escrito a mano sin la necesidad de aplicar filtros manuales o modelos de lenguaje, aportando flexibilidad respecto de los métodos tradicionales de OCR (incluyendo métodos de aprendizaje automático como SVM). En este sentido, tiene el potencial de generalizar de manera eficiente a través de distintos idiomas y ámbitos, como lo menciona Graves y Schmidhuber, 2009, hecho que no se verifica necesariamente en el análisis cualitativo -tanto a nivel de segmentación como de reconocimiento- presente trabajo.

Con respecto a los desafíos que se presentaron en el presente trabajo, uno de los más relevantes estuvo vinculado al conjunto de datos. En primer lugar, su tamaño reducido respecto de otros *corpus* es, de por sí, un desafío para el entrenamiento de numerosas arquitecturas de aprendizaje profundo con millones de parámetros a ajustar. La forma en que fueron obtenidas las imágenes -fotografías, versus páginas escaneadas en el conjunto de datos

del análisis cualitativo- y el modelo de lenguaje que se puede inferir en el reconocimiento de caracteres al sólo contar con muestras en idioma inglés, también impactaron en la capacidad de generalización del modelo sobre otra colección.

En términos de procedimientos, el primer paso del reconocimiento de texto escrito a mano, consistente en la detección de las palabras, hizo un uso provechoso de las técnicas de aprendizaje por transferencia. La utilización de modelos pre-entrenados del *Model Zoo* de PyTorch permitió no sólo reducir los tiempos dedicados a esa tarea sino obtener resultados consistentes con otros conjuntos de datos (Lee y col., 2021). Se observó también una alta capacidad de invariancia a la traslación, registrando texto a lo largo y ancho de todas las imágenes.

Además de verse beneficiado el desempeño del modelo en términos de *Total Loss* se puede pensar que al modularizar la detección de las palabras logramos también hacer más eficiente el flujo de trabajo, sin necesidad de generar inferencias en una infraestructura que tenga la memoria suficiente para ejecutar una red neuronal que realice el proceso de punta a punta. En este aspecto, más allá de las dos alternativas de detección, no se evidenciaron diferencias considerables en el desempeño (si eliminamos de la pérdida total aquella correspondiente a la máscara).

La última apreciación acerca de la segmentación proviene del tipo de caracteres a detectar. Como hemos visto, la posibilidad de definir si se busca localizar únicamente conjuntos de caracteres alfanuméricos o también de otro dominio (en particular, caracteres matemáticos) es un atributo interesante para los distintos usos que pueden tener estas técnicas.

Respecto de la automatización del proceso de reconocimiento, el proceso de regularización (con la transformación previa mediante redes de transforma-

ción espacial) reduce la necesidad de hacer ajustes manuales en las imágenes para identificar de manera correcta -rectificando, por ejemplo- los bloques de texto. Para la extracción de atributos hemos visto que se evidencia una diferencia significativa de tiempos de ejecución y costo computacional. El tercer paso, el modelado secuencial, enriquece el insumo de las predicciones convirtiendo información visual en contextual al utilizar redes recurrentes bidireccionales. El último paso, la predicción, incorporó una técnica -la atención- que reduce la atenuación de la señal a medida que se incrementa la extensión de la palabra. Finalmente, esas predicciones fueron evaluadas con la métrica de entropía cruzada.

Como mencionamos, en términos cuantitativos se evidenciaron diferencias considerables entre las arquitecturas de extracción de atributos en los resultados con 100 epochs, con un destacado desempeño de la arquitectura ResNet, que -por otro lado- requirió una infraestructura y un tiempo de entrenamiento significativamente mayor. Al incrementar a 250 la cantidad de iteraciones el desempeño de las arquitecturas presenta diferencias a favor de ResNet, tal como cuando se evaluaron 100 iteraciones, pero las brechas son significativas. Desde el punto de vista cualitativo, observamos que aquellos documentos con menor cantidad de texto parecieran ser el mejor tipo de imagen para detectar y reconocer, mientras que la correspondencia de cursiva pronunciada y múltiples renglones se superpone en sus polígonos (*bounding boxes*) y presenta un desafío aún no resuelto.

El análisis cualitativo del reconocimiento de caracteres presentó desafíos para reconocer los textos escritos con una grafía altamente compleja, siendo capaz de detectar correctamente en ocasiones palabras completas o casi completas pero en otras no lográndolo. Más allá de eso, se puede definir como aporte la capacidad de haber segmentado correctamente una cantidad significativa de palabras en esos textos, que se disponibilizan de forma

sistematizada para la Biblioteca de la Facultad, y generado una base de conocimiento a partir de los términos detectados en más de 15 mil polígonos identificados para más de 400 documentos manuscritos.

En relación al idioma a reconocer, y con las limitaciones evidentes respecto del modelo del lenguaje subyacente, dado que el conjunto, se apreció una invariancia -en términos cualitativos- para la detección de algunos caracteres dentro de las palabras, resaltando las vocales como aquellos mejor predichos. Algo similar puede resaltarse de los caracteres en mayúsculas. Por último, observamos que el modelo tiene la capacidad -aunque con limitaciones- de reconocer texto que incluye tanto números como caracteres matemáticos, lo que presenta una ventaja en este tipo de material. A este respecto, es evidente que el modelo

Como última observación - y en términos generales- entendemos que el desarrollo teórico y práctico con un alto nivel de detalle de las técnicas, arquitecturas y tecnologías realizado a lo largo del presente trabajo representa un aporte al acervo bibliográfico escrito en castellano referido al aprendizaje profundo, escaso en este tipo de áreas.

5.2. Desafíos a futuro

En los últimos años han surgido métodos, entre ellos *Segmentation-Free Writer Identification* de Kumar y Sharma, 2020, que permiten detectar y reconocer caracteres sin la necesidad de aplicar una segmentación previa. Esa línea de investigación es una que escapa a nuestros objetivos actuales pero que merece ser estudiada en futuros trabajos. El otro desafío es incorporar el uso de *lexicons* para mejorar, aunque tal vez impactando la automatización, la correcta detección de palabras y reducir palabras con un único caracter

errado (tal como vimos en el apartado correspondiente).

En términos de la comparación con los métodos tradicionales de OCR, uno de los aspectos que sigue teniendo un potencial de mejora significativa en el desempeño, aún a costa de quitarle flexibilidad a los métodos, es el de los modelos de lenguaje, particularmente en dominios específicos como la matemática o la medicina.

De acuerdo a lo mencionado por Chung y Delteil, 2019 las métricas (en su caso, el CER) evidencian una mejora al utilizar un modelo de lenguaje (*lexicon searching*) pero esto pareciera estar más influenciado por el tipo de búsqueda -*beam* versus *greedy*- que por el modelo de lenguaje en sí. Para próximos desarrollos, la combinación de las técnicas actuales con modelos de lenguaje como el CROHME para matemática o la utilización de vademecums médicos podrá cuantificar el margen de mejora en este sentido. Del mismo modo, evaluar distintas predicciones con métricas alternativas (como la mencionada de CTC) puede aportar en el sentido de la comparabilidad con otros estudios o conjuntos de datos.

Otro aspecto a desarrollar es el del aprendizaje por transferencia. Se demostró como una herramienta útil principalmente en la detección de objetos pasando de modelos pre-entrenados con el conjunto de datos COCO a palabras en una imagen, reduciendo los tiempos de entrenamiento y dando resultados similares a los de la literatura. Respecto del aprendizaje por transferencia en los modelos de reconocimiento de caracteres, incorporarlo y utilizar *fine-tuning* en algunas de las arquitecturas entrenadas (las redes convolucionales, las redes recurrentes bidireccionales y el módulo de atención) podría mejorar el desempeño.

De la mano de la mejora del desempeño, ampliar el conjunto de datos tanto a nivel de idioma como cantidad de imágenes podría tener un impacto con-

siderable en la capacidad de segmentación y reconocimiento de caracteres. Respecto de este punto, si bien se hizo un análisis cualitativo con datos del conjunto *CSafe* (Crawford y col., 2019) y distintos documentos en diversos idiomas, creemos que existe un camino interesante para el *fine-tuning* con colecciones de documentos como *RIMES (Reconnaissance et Indexation de données Manuscrites et de fac similÉS)*, *Ratsprotokolle* y los correspondientes al proyecto *tranScriptorium*.

A nivel de infraestructura, uno de los principales desafíos es el de contar con una capacidad de cómputo suficiente para obtener los mejores resultados posibles. Si bien en el presente trabajo se utilizaron instancias con una capacidad significativa -con hasta 488 GiB y 64 vCPU y cómputo acelerado con 8 GPUs NVIDIA Tesla V100, como se mencionó anteriormente- utilizar entrenamiento distribuido puede mejorar aún más los resultados obtenidos. Del mismo modo, para mejorar el análisis de caracteres dentro de los polígonos -y, en términos generales, el reconocimiento de todos los polígonos existentes en un documento- las alternativas son el incremento de capacidad de cómputo para poder procesar documentos completos o, caso contrario, la segmentación y posterior reconstrucción de cada uno de modo de optimizar los recursos disponibles.

Finalmente, y en términos de aporte a las instituciones públicas, la digitalización a escala y puesta a disposición de este tipo de materiales presenta un área de gran oportunidad en dos aspectos. En primer lugar, por la capacidad de generar nuevas investigaciones utilizando material hasta ahora no analizado cuantitativamente: el modelado de tópicos, por ejemplo, es una posibilidad a desarrollar en la correspondencia bajo estudio cualitativo en este trabajo. En segundo, el desarrollo de aplicaciones que faciliten al usuario final el uso de los datos analizados y permitan acceder mediante palabras clave a documentos específicos puede ser otra línea a profundizar en futuros

trabajos.

Bibliografía

Awal, A. M., Mouchère, H. & Viard-Gaudin, C. (2010). The Problem of Handwritten Mathematical Expression Recognition Evaluation. https://doi.org/10.1109/ICFHR.2010.106

Baek, J., Kim, G., Lee, J., Park, S., Han, D., Yun, S., Oh, S. J. & Lee, H. (2019). What Is Wrong With Scene Text Recognition Model Comparisons? Dataset and Model Analysis, En *International Conference on Computer Vision (ICCV)*.

Bluche, T., Louradour, J. & Messina, R. (2017). Scan, Attend and Read: End-to-End Handwritten Paragraph Recognition with MDLSTM Attention. https://doi.org/10.1109/ICDAR.2017.174

Bookstein, F. (1989). Principal warps: thin-plate splines and the decomposition of deformations. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *11*(6), 567-585. https://doi.org/10.1109/34.24792

Cheng, Z., Bai, F., Xu, Y., Zheng, G., Pu, S. & Zhou, S. (2017). Focusing Attention: Towards Accurate Text Recognition in Natural Images, En *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*.

Chollet, F. (2017). *Deep Learning with Python*. Manning.

Chung, J. & Delteil, T. (2019). A Computationally Efficient Pipeline Approach to Full Page Offline Handwritten Text Recognition. *In preprint*.

Cornegruta, S., Bakewell, R., Withey, S. & Montana, G. (2016). Modelling Radiological Language with Bidirectional Long Short-Term Memory Networks. https://doi.org/10.18653/v1/W16-6103

Crawford, A., Ray, A., Carriquiry, A., Kruse, J. & Peterson, M. (2019). CSAFE Handwriting Database. https://doi.org/10.25380/iastate.10062203.v1

Dai, J., Qi, H., Xiong, Y., Li, Y., Zhang, G., Hu, H. & Wei, Y. (2017). Deformable Convolutional Networks.

Géron, A. (2017). *Hands-on machine learning with Scikit-Learn and TensorFlow : concepts, tools, and techniques to build intelligent systems*. Sebastopol, CA, O'Reilly Media.

Girshick, R. (2015). Fast R-CNN. arXiv.

Goodfellow, I. J., Bengio, Y. & Courville, A. (2016). *Deep Learning*. Cambridge, MA, USA, MIT Press.

Graves, A. & Schmidhuber, J. (2009). Offline Handwriting Recognition with Multidimensional Recurrent Neural Networks, En *Advances in Neural Information Processing Systems (NIPS) 21*. Cambridge, MA, MIT Press.

Graves, A., Fernández, S., Gomez, F. & Schmidhuber, J. (2006). Connectionist Temporal Classification: Labelling Unsegmented Sequence Data with Recurrent Neural Networks, En *Proceedings of the 23rd International Conference on Machine Learning*, Pittsburgh, Pennsylvania, USA, Association for Computing Machinery. https://doi.org/10.1145/1143844.1143891

Grosicki, E., Carré, M., Augustin, E. & Prêteux, F. (2006). La campagne d'évaluation RIMES pour la reconnaissance de courriers manuscrits, En *Colloque International Francophone sur l'Ecrit et le Document*.

He, K., Gkioxari, G., Dollár, P. & Girshick, R. B. (2017). Mask R-CNN. *CoRR*, *abs/1703.06870*arXiv 1703.06870. http://arxiv.org/abs/1703.06870

He, K., Zhang, X., Ren, S. & Sun, J. (2015). Deep Residual Learning for Image Recognition. *CoRR*, *abs/1512.03385*arXiv 1512.03385. http://arxiv.org/abs/1512.03385

Kamalanaban, E., Gopinath, M. & Premkumar, S. (2018). Medicine Box: Doctor's Prescription Recognition Using Deep Machine Learning. *International Journal of Engineering and Technology(UAE)*, *7*, 114-117. https://doi.org/10.14419/ijet.v7i3.34.18785

Kowsar, K., Heidarysafa, M., Brown, D. E., Meimandi, K. J. & Barnes, L. E. (2018). RMDL: Random Multimodel Deep Learning for Classification. *Proceedings of the 2018 International Conference on Information System and Data Mining*.

Kumar, P. & Sharma, A. (2020). Segmentation-free writer identification based on convolutional neural network. *Computers & Electrical Engineering*, *85*, 106707. https://doi.org/https://doi.org/10.1016/j.compeleceng.2020.106707

LeCun, Y. & Cortes, C. (2010). MNIST handwritten digit database. http://yann.lecun.com/exdb/mnist/

Lee, A. W. C., Chung, J. & Lee, M. (2021). GNHK: A Dataset for English Handwriting in the Wild, En *International Conference of Document Analysis and Recognition (ICDAR)*.

Lin, T.-Y., Dollár, P., Girshick, R., He, K., Hariharan, B. & Belongie, S. (2017). Feature Pyramid Networks for Object Detection, En *2017*

IEEE Conference on Computer Vision and Pattern Recognition (CVPR). https://doi.org/10.1109/CVPR.2017.106

Liu, W., Anguelov, D., Erhan, D., Szegedy, C. & Reed, S. (2015). SSD: Single Shot MultiBox Detector.

MacKay, D. J. C. (2003). *Information Theory, Inference, and Learning Algorithms*. Copyright Cambridge University Press.

Marti, U.-V. & Bunke, H. (1999). A full English sentence database for off-line handwriting recognition, En *In Proc. Int. Conf. on Document Analysis and Recognition*.

Mitchell, T. M. (1997). *Machine Learning*. New York, McGraw-Hill.

Nasien, D., Haron, H. & Yuhaniz, S. S. (2010). Support Vector Machine (SVM) for English Handwritten Character Recognition, En *2010 Second International Conference on Computer Engineering and Applications*.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Kopf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., ... Chintala, S. (2019). PyTorch: An Imperative Style, High-Performance Deep Learning Library (H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox & R. Garnett, Eds.). En H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox & R. Garnett (Eds.), *Advances in Neural Information Processing Systems 32*. Curran Associates, Inc. http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf

Puigcerver, J. (2017). Are Multidimensional Recurrent Layers Really Necessary for Handwritten Text Recognition?, En *2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR)*. https://doi.org/10.1109/ICDAR.2017.20

Ren, S., He, K., Girshick, R. B. & Sun, J. (2015). Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks., En *NIPS*.

Saba, T., Rehman, A. & Elarbi-Boudihir, M. (2011). Methods and strategies on off-line cursive touched characters segmentation: a directional review. *Artificial Intelligence Review*, *42*, 1047-1066.

Schantz, H. F. (1982). *The history of OCR: optical character recognition*. Manchester, VT, Recognition Technologies Users Association.

Scheidl, H. (2018). *Handwritten Text Recognition in Historical Documents*. Technische Universität Wien.

Schuster, M. & Paliwal, K. (1997). Bidirectional recurrent neural networks. *Signal Processing, IEEE Transactions on*, *45*, 2673-2681. https://doi.org/10.1109/78.650093

Services, A. W. (s.f.). Overview of Amazon Web Services [Accedido el 28/08/2021].

Shi, B., Bai, X. & Yao, C. (2015). An End-to-End Trainable Neural Network for Image-based Sequence Recognition and Its Application to Scene Text Recognition.

Shi, B., Wang, X., Lyu, P., Yao, C. & Bai, X. (2016). Robust Scene Text Recognition with Automatic Rectification.

Singh, A. & Desai, S. (2016). Optical character recognition using template matching and back propagation algorithm, En *2016 International Conference on Inventive Computation Technologies (ICICT)*. https://doi.org/10.1109/INVENTIVE.2016.7830161

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł. & Polosukhin, I. (2017). Attention is All you Need, En *Advances in Neural Information Processing Systems 30*. Curran Associates, Inc.

Veit, A., Matera, T., Neumann, L., Matas, J. & Belongie, S. (2016, 26 de enero). COCO-Text: Dataset and Benchmark for Text Detection and

Recognition in Natural Images, En *arXiv preprint arXiv:1601.07140*. http://vision.cornell.edu/se3/wp-content/uploads/2016/01/1601.07140v1.pdf

Wang, J. & Hu, X. (2021). Convolutional Neural Networks with Gated Recurrent Connections. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1-1. https://doi.org/10.1109/tpami.2021.3054614

Wu, Y., Kirillov, A., Massa, F., Lo, W.-Y. & Girshick, R. (2019). Detectron2.

Yang, M., Guan, Y., Liao, M., He, X., Bian, K., Bai, S., Yao, C. & Bai, X. (2019). Symmetry-constrained Rectification Network for Scene Text Recognition.

Zhang, A., Lipton, Z. C., Li, M. & Smola, A. J. (2020). *Dive into Deep Learning* [https://d2l.ai]. Preview version.