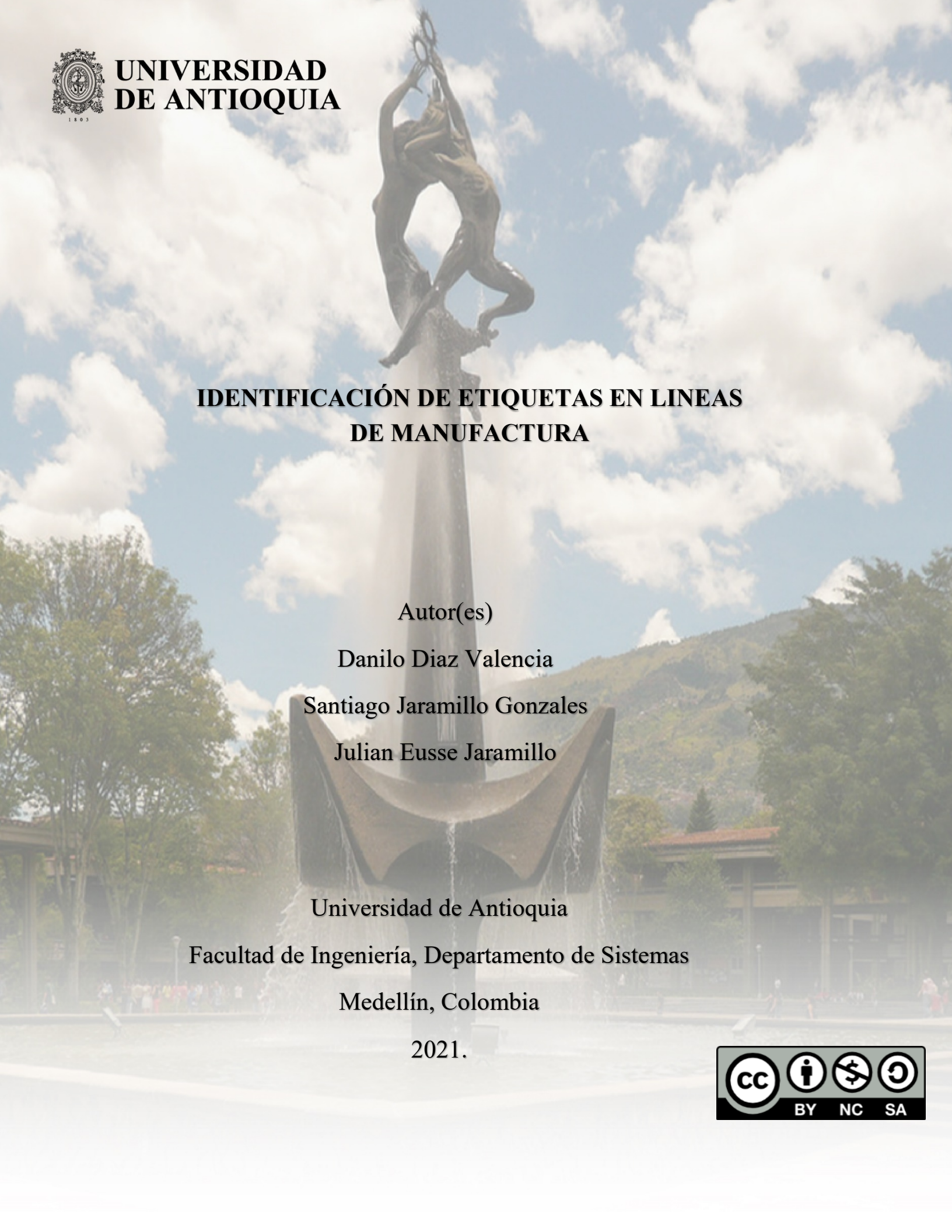


UNIVERSIDAD DE ANTIOQUIA

IDENTIFICACIÓN DE ETIQUETAS EN LINEAS DE MANUFACTURA

Autor(es)

Danilo Diaz Valencia

Santiago Jaramillo Gonzales

Julian Eusse Jaramillo

Universidad de Antioquia

Facultad de Ingeniería, Departamento de Sistemas

Medellín, Colombia

2021.

Identificación de Etiquetas en Líneas de Manufactura

Danilo Diaz Valencia

Santiago Jaramillo Gonzales

Julian Eusse Jaramillo

Tesis o trabajo de investigación presentada(o) como requisito parcial para optar al título de:

Especialista en Analítica y Ciencia de Datos

Asesores (a):

Sebastian Rodríguez Colina: Ingeniero Mecánico

Línea de Investigación:

Visión Artificial

Universidad de Antioquia

Facultad de Ingeniería, Ingeniero de Sistemas

Medellín, Colombia

2021.

1. Introducción

Dentro de la visión artificial, el reconocimiento de objetos en imágenes y su identificación es una del área más atractiva y que mayor desarrollo han tenido en los últimos años. Gracias a diferentes tecnologías desarrolladas en la última década, el reconocimiento de objectos en imágenes se ha convertido casi en un problema abordable y configurable mediante técnica de Machine Learning. El gran desafío actual es el mejorar estos sistemas de reconocimientos de imágenes, puliendo ciertos detalles como su requerimiento de grandes datos, capacidad de cómputo y precisión de reconocimiento.

Estos grandes avances que han sufrido el área de la visión artificial en los últimos años ya han permitido la creación e implementación de diversas aplicaciones basadas en el reconocimiento de imágenes parea mejoraras a ayudar a procesos actualmente realizados de forma manual en diferentes áreas de la sociedad como en la medicina, la seguridad de las ciudades, los automóviles, el comercio, la industria ... En el área de la industria es en el cual se centra este trabajo. La industria esta sufriendo un proceso evolutivo hacia lo que se denomina la 4ta revolución industrial, la cual tiene como principales objetivos lograr una digitalización total de sus procesos, con la finalidad de obtener la mayor cantidad de datos posibles para mejorar sus procesos productivos.

Este trabajo final tiene como finalidad desarrollar un sistema de cisión artificial, basado en el reconocimiento de imágenes, con el objetivo de ser implementado en procesos productivos de empresas que tienen líneas de producción y usan sistemas de etiquetas para procesos de calidad o trazabilidad.

La parte innovadora que pretende este proyecto es la de diseñar un demo de reconocimiento de imágenes implementado en un entorno industrial. Simularemos una información con imágenes de hojas de papel que tiene información de interés que debe ser extraída. Realizaremos varios experimentos centrándolos en los 3 ámbito fundamentales del proceso; detección del objeto, reubicación del objecto y ubicación de textos de interés

1.a. Proceso actual

En toda empresa, el diseño de líneas de manufactura para sistemas de producción es de suma importancia, ya que estas dan un rendimiento continuo de carácter económico que afecta directamente a la empresa. En las líneas de manufactura se realizan 2 procesos específicos; acción y verificación.

Para este proyecto nosotros nos centraremos netamente en el aspecto de verificación en líneas de manufactura con el uso de etiquetas. Estas son usadas para manejar algún tipo de trazabilidad que podría ser usada en la siguiente etapa de producción o para tomar algún tipo de decisión sobre esta.

En general los procesos de revisión de las líneas de producción se realizan de una forma manual, un operario es el encardo de revisar mediante operación visual si el componente manufacturado esta realizado de una manera correcta, esto se realiza comúnmente mediante aquietas o documentos que se van a anexando al producto, estos nos indican si el componente está realizando de una manera correcta Figura :1.

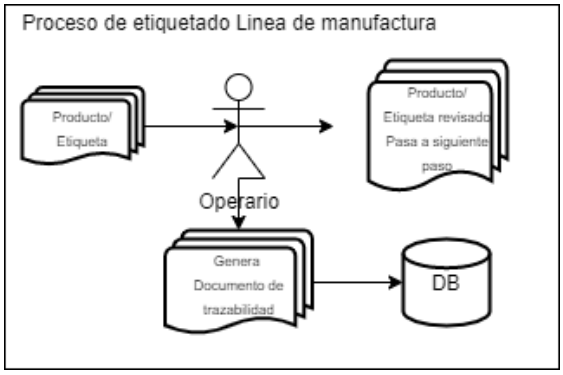


Figure 1: Diagrama proceso actual

1.b. Beneficios de una solución automatizada

La aplicación de los sistemas de visión artificial en las líneas de manufactura de las industrias y a las empresas trae consigo múltiples beneficios. La funcionalidad de estas herramientas no sólo permite solucionar los problemas de control de proceso de fabricación, sino también de ayudar al aumento de la producción. Del mismo modo, estos sistemas pueden brindar las siguientes ventajas.

- Permiten que aumenten la calidad de los productos al ser inspeccionados de forma detallada y precisa por las máquinas.
- Generan un aumento en la productividad puesto que las producciones defectuosas se van eliminando de forma continua
- El proceso de control de calidad es más preciso ya que no existe un margen de error elevado a causa de las limitaciones visuales de los operadores.

En toda empresa, el diseño de líneas de manufactura para sistemas de producción es de suma importancia, ya que estas dan un rendimiento continuo de carácter económico que afecta directamente a la empresa. En las líneas de manufactura se realizan 2 procesos específicos; acción y verificación.

Para este proyecto nosotros nos centraremos netamente en el aspecto de verificación en líneas de manufactura con el uso de etiquetas. Estas son usadas para manejar algún tipo de trazabilidad que podría ser usada en la siguiente etapa de producción o para tomar algún tipo de decisión sobre esta.

En general los procesos de revisión de las líneas de producción se realizan de una forma manual, un operario es el encardo de revisar mediante operación visual si el componente manufacturado esta realizado de una manera correcta, esto se realiza comúnmente mediante aquietas o documentos que se van a anexando al producto, estos nos indican si el componente está realizando de una manera correcta Figura

1.c. Requerimientos técnicos

Con el fin de desarrollar un sistema de visión artificial industrial económico, se van a necesitar una serie de herramientas fundamentales para su desarrollo.

En esta sección pretende hacer un listado de las herramientas o conceptos empleados al largo de la monografía, incluyendo lenguajes de programación empleados como "frameworks" o herramientas de "Machine Learning" usados. Todo el proyecto puede ser consultado en el repositorio en GitHub:

- **Python**
- **Jupyter Notebook**
- **OpenCv**
- **LabelImg**
- **TensorFlow**
- **Aumentación de datos**
- **Transfer Learning**

2. Diseño de la Solución

Teniendo en cuenta el problema descrito anteriormente, hemos decidido desarrollar una solución basada en deep learning que implementa técnicas de visión artificial.

Para alcanzar esta meta hemos decidido dividir el problema en 3 partes.

- Detección de hoja
- Reorientación
- Detección de palabras clave

Figure 2: Diagrama de flujo

2.a. Detección de Etiqueta en el espacio

Este paso consiste en encontrar de manera eficiente la ubicación de una hoja de papel en una fotografía. El objetivo de este proceso es determinar que parte de la imagen contiene información relevante acerca de un productro que está ubicado en una línea de producción. La salida de este modelo debe ser un recorte que delimite y muestre una hoja para luego enderezarla y posteriormente encontrar parabras clave Figura : 3, para extraer información relevante de dicho documento.

Para desarrollar esta tarea se propone implementar un modelo que delimite la localización de la hoja lo mas preciso posible, por medio de cajas que la encierren.

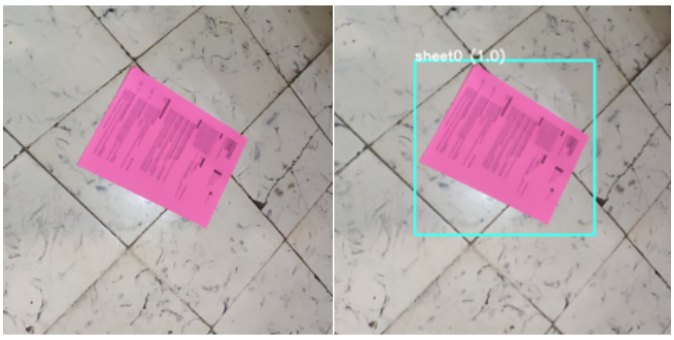


Figure 3: Corrección de posición de Etiqueta

2.b. Corrección de posición de Etiqueta

Una vez encontrada la hoja dentro de la imagen, el siguiente paso es corregir el ángulo en el que se encuentra. Esto se hace con el fin de obtener una imagen recta donde sea mucho mas sencillo encontrar las palabras clave que se encuentran en el documento.

Este proceso es necesario debido a que el modelo que detecta la posición de las palabras clave será entrenado con un dataset donde todas las hojas estan perfectamente alineadas y de no ser así, la efectividad del modelo puede verse comprometida.

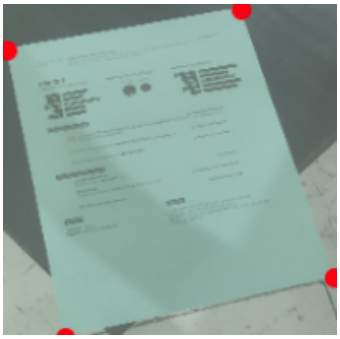


Figure 4: Deteccion de Esquinas

Para desarrollar esta tarea, se propone la generación de un modelo que determine los

keypoints referentes a las 4 esquinas de la hoja y a partir de ahí aplicar una transformación de cambie la orientación de la hoja.

2.c. Detección de Palabra

El último paso para obtener el resultado esperado es identificar una cantidad de parabras clave que se encuentran presentes en los documentos que se encuentran aderidos en los vehículos en producción Figura : 5. Esta tarea se hace para encontrar los lugares donde hay información crítica que se quiere almacenar, por lo cual es importante definir de antemano las palabras objetivo.

Figure 5: Detección de palabra

El alcance de este proyecto solo consive encontrar las siguientes palabras, pero la aproximación llevada a cabo debería ser fácil de extender a otras palabras clave de interés:

- **Total**
- **Logo**
- **Thanks**
- **Change**

3. Experimentación con Soluciones de Inteligencia Artificial

3.a. Ubicación de hojas

Una vez se divide el proyecto en las tres diferentes secciones, se procede a establecer las diferentes formas en las cuales se puede resolver el problema planteado. Para resolver este problema se determinó utilizar modelos de deep learning de detección. Particularmente se utilizó como base Mobilnet V2 224x224. Este modelo se seleccionó debido a su rápido entrenamiento originado por las depthwise convolutions, que permiten tener menos pesos que las capas convolucionales tradicionales

3.a.1. Recolección de datos

Para la recolección de los datos se imprimieron datos en hojas de diferentes colores y se recolectaron fotografías que contuvieran dichas hojas. Estas muestras debían contener la mayor bariabilidad posible para que el modera fuera significativamente robusto. En total se recolectaron 501 imágenes que contienen una hoja por cada imagen y se etiquetaron utilizando Labeling, herramienta que permite generar cuadros y etiquetas en cualquier parte de una imagen.

Para aumentar la variablilidad del dataset se tomó en cuenta las siguientes variables:

- Posición
- Color
- Luminocidad
- fondo de la fotografía
- Dispositivos para tomar los datos

subfig

(a) Ejemplo 1 (b) Ejemplo 2 (c) Ejemplo 3

Figure 6: ejemplos

3.a.2. Separación de datos

Una vez generado el dataset, se generó una funcion que aleatoriamente divide los datos en entrenamiento y prueba. Esta división se hizo tomando el 30 porciento de los datos para prueba y 70 porciento para entrenamiento debido a la corta extensión de datos.

3.a.3. Función de costo

Para el entrenamiento de modelos, es importante determinar una función que queremos optimizar. Para esta tarea, es importante tener en cuenta 2 variables.

- Localización
- Clasificación

Teniendo en cuenta lo anterior, la función de pérdida que se va a utilizar va a ser una mezcla entre weighted smooth l1 y weighted sigmoid focal.

3.a.4. Baseline

Para el desarrollo de este proyecto se utilizo unicamente SSD Mobilnet v2 224X224. Este modelo se escogió porque contiene convoluciones depthwise, lo que implica que las capas convolucionales tengan menos pesos para sintonizar, lo que desemboca en un entrenamiento mucho mas veloz.

En la siguiente figura se muestra los tipos de convoluciones que se utilizan para lograr este objetivo.

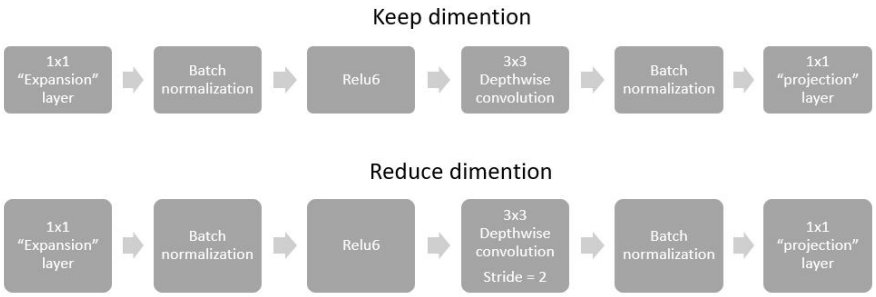


Figure 7: Estructuras base

Adicional a esto, es importante mostrar la evolución de las dimenciones de nuestra imagen de entrada hasta obtener el resultado final.

Los parámetros mas relevantes que se utilizaron para el entrenamiento de este modelo fueron los siguietes:

- input shape: 224x224
- num classes: 1
- batch size: 8
- optimezer: momentum

Conv / s2	$3 \times 3 \times 3 \times 32$	$224 \times 224 \times 3$
Conv dw / s1	$3 \times 3 \times 32$ dw	$112 \times 112 \times 32$
Conv / s1	$1 \times 1 \times 32 \times 64$	$112 \times 112 \times 32$
Conv dw / s2	$3 \times 3 \times 64$ dw	$112 \times 112 \times 64$
Conv / s1	$1 \times 1 \times 64 \times 128$	$56 \times 56 \times 64$
Conv dw / s1	$3 \times 3 \times 128$ dw	$56 \times 56 \times 128$
Conv / s1	$1 \times 1 \times 128 \times 128$	$56 \times 56 \times 128$
Conv dw / s2	$3 \times 3 \times 128$ dw	$56 \times 56 \times 128$
Conv / s1	$1 \times 1 \times 128 \times 256$	$28 \times 28 \times 128$
Conv dw / s1	$3 \times 3 \times 256$ dw	$28 \times 28 \times 256$
Conv / s1	$1 \times 1 \times 256 \times 256$	$28 \times 28 \times 256$
Conv dw / s2	$3 \times 3 \times 256$ dw	$28 \times 28 \times 256$
Conv / s1	$1 \times 1 \times 256 \times 512$	$14 \times 14 \times 256$
$5\times$ Conv dw / s1	$3 \times 3 \times 512$ dw	$14 \times 14 \times 512$
Conv / s1	$1 \times 1 \times 512 \times 512$	$14 \times 14 \times 512$
Conv dw / s2	$3 \times 3 \times 512$ dw	$14 \times 14 \times 512$
Conv / s1	$1 \times 1 \times 512 \times 1024$	$7 \times 7 \times 512$
Conv dw / s2	$3 \times 3 \times 1024$ dw	$7 \times 7 \times 1024$
Conv / s1	$1 \times 1 \times 1024 \times 1024$	$7 \times 7 \times 1024$
Avg Pool / s1	Pool 7×7	$7 \times 7 \times 1024$
FC / s1	1024×1000	$1 \times 1 \times 1024$

Figure 8: Estructuras base

3.a.5. Modelo usando data augmentation

Una vez generadoi el modelo y viendo que aun tenía algo por mejorar, se decidió utilizar aumentación de datos. Esta es una técnica que se basa en replicar las imágenes con ciertas variaciones, con el fin de tener ua base de datos mas rica en imágenes. Por otro lado, este método es muy eficaz cuando se quiere entrenar un modelo para realizar alguna tarea pero no se tienen suficientes datos para entrenarlo y teniendo en cuenta que tanto la obtención como el etiquetado se hicieron manualmente, era la aproximación mas razonable.

A continuación se muestran los tipos de aumentación de datos que se implementaron. subfig

3.a.6. Resultados

Una vez implementados todos los modelos, se utilizó intersection over union como métrica para evaluar el desempeo de cada una de las iteraciones con el fin de demostrar que tan eficaz es cada modelo para detectar las hojas de papel ypara observar que márgenes de

(a) Original (b) rgb to gray (c) Horizontal flip

Figure 9: ejemplos

	Baseline	Data augmentation
train	0.74	0.82
test	0.64	0.71

Table 1: IoU de los modelos

mejora tiene el modelo.

3.b. Ajuste de ubicación de las hojas

Para el ajuste de las hojas en la posición adecuada, se planteó el objetivo de detectatar las esquinas de las hojas para recolectar información de la posición actual de la hoja y realizar una transformación para dejarla en posición vertical . El planteamento de la solución fue basado en una la red MobileNet v2 con pesos prenetrenados de 'imagenet' solamente modificandole la última capa, siendo el output de la red cada una de las cordenadas X y Y de las esquinas de la hoja.

3.b.1. Recolección de datos

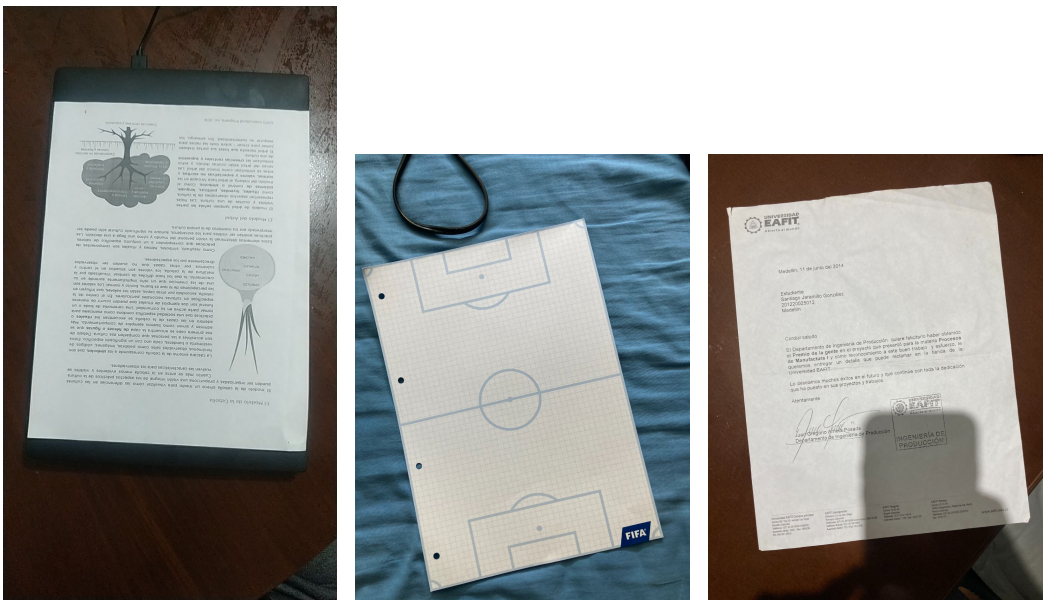
La recolección de los datos se hizo manualmente, tomando 500 imágenes con diferentes dispositivos mobiles para garantizar la heterogeneidad de las imágenes. Otro parámetro que se tomó en cuenta fue la inclinación de las hojas, que no podían tener un ángulo mayor a 30 grados, para facilitar el entrenamiento y replicar las circunstancias en las que el proyecto se desplegaría.

Adicionalmente se etiquetaron las imágenes utilizando 'VGG Image Annotator (VIA)' como herramienta predilecta, generando un archivo JSON con las coordenadas de los 4 puntos de interes más el nombre del archivo. En la siguiente imagen se pueden observar las etiquetas de los puntos.

TL_x_rel	TL_y_rel	TR_x_rel	TR_y_rel	BL_x_rel	BL_y_rel	BR_x_rel	BR_y_rel
0.074172	0.051567	0.887417	0.017189	0.030464	0.964611	0.969536	0.898888
0.000000	0.145478	0.643736	0.005898	0.210717	0.986239	0.985518	0.762779

Figure 10: Etiquetas

En la siguiente imagen se puede observar algunos ejemplos del dataset previamente descrito.



(a) Ejemplo 1 (b) Ejemplo 2 (c) Ejemplo 3

Figure 11: ejemplos

3.b.2. Separación de datos

La separación de los datos se realizó a partir de una función que genera aleatoriamente un numero entre 0 y 1 para cada imagen en el dataset. Si el número generado es mayor a 0.9, se genera una etiqueta de 'test', que se adicionaba en el metadata de los datos, si el número es menor, se genera una etiqueta de 'train'.

Split	Valor
Train	0.9
Test	0.1

Table 2: Separación de datos de entranamiento

3.b.3. Función de costo

Con el objetivo de tener una función que indique como deben variar los pesos del modelo que se está entrenando, se utilizó como función de costo el mean squared error (mse) que determina el error cuadrático medio en cada uno de los puntos que se estan intentando generar.

Esta métrica permite determinar la distancia a la que está el punto real y la predicción del modelo, lo que la hace una excelente opción para este tipo de problema que se planteo como un problema de regresión.

3.b.4. Selección de métrica

Con el fin de evaluar los modelos que se generan, se definió el mean squared error para evaluar el performance del modelo ya que este se plantea como un problema de regresión, donde el output del modelo son 8 valores de 0 a 1 indicando las coordenadas x,y de cada una de las esquinas de la hoja.

3.b.5. Baseline

El modelo que se utilizó para localizar los puntos clave fue:

```
Model: "sequential_4"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
mobilenetv2_1.00_224 (Functi (None, 7, 7, 1280)        2257984
_________________________________________________________________
global_average_pooling2d_4 ( (None, 1280)              0
_________________________________________________________________
dense_4 (Dense)              (None, 8)                 10248
=================================================================
Total params: 2,268,232
Trainable params: 2,234,120
Non-trainable params: 34,112
_________________________________________________________________
```

Figure 12: Arquitectura

Este modelo se le aplicó una técnica de transfer learning utilizando la red MobilNet v2 224x224 con unos pesos prentrenados en 'imagenet'.

Los parámetros mas relevantes que se utilizaron para el entrenamiento de este modelo fueron los siguietes:

Características	Valor
input shape	224x224
num salidas	8
rango salidas	(0,1)
batch size	10
optimezer	RMS prop
learning rate	1e-4
epochs	65

Table 3: Separación de datos de entranamiento

El proceso de entrenamiento se evidencia con la siguiente gráfica donde se aplica la métrica tanto a el set de entramiento como al set de validación y se utilizan callbacks y Tensorboard para visualizar estos valores a medida que sucede el entrenamiento:

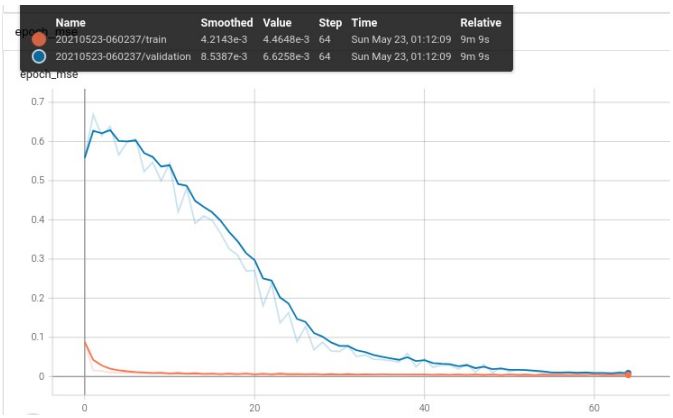


Figure 13: Entrenamiento

3.b.6. Modelo usando aumentación de datos

Para mejorar los resultados obtenidos por el modelo anterior, se decide aplicar aumentación de datos a las imágenes y se utiliza la misma configuración del modelo anterior, ver tabla 3. Las transformaciones realizadas son las siguientes:

Características	Valor
brightness	max_delta=0.2
contrast	lower=0.5, upper=1.5
hue	max_delta=0.2
saturation	lower=0.5, upper=1.5

Table 4: Procesos de aumentación de datos

El proceso de entrenamiento se evidencia con la siguiente gráfica donde se asegura que el proceso de entranamiento este reduciendo la perdida en cada paso y que la métrica en el set de entrenamiento baje y no se evidencie un crecimiento para asi evitar el sobre ajuste

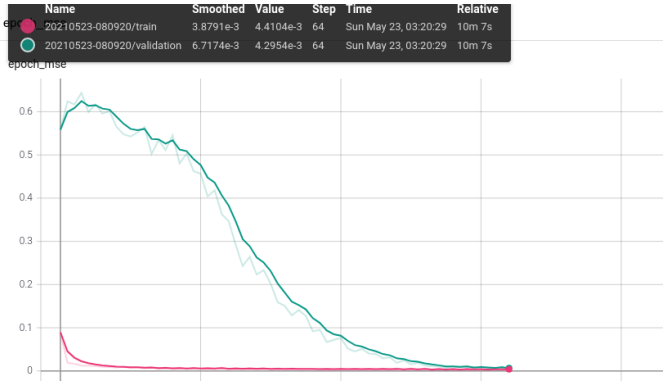


Figure 14: Entrenamiento Data Augmentation

La siguiente tabla resume de los resultados obtenidos con el modelo final en el cual se realizaron varios experimentos utilizando Transfer Learning y Aumentación de Datos, en la figura 12 muestra la arquitectura usada en el modelo.

Experimentos	MSE test
Transfer Learning	0.0066
Aumentación de datos + Transfer Learning	0.0043

Table 5: Resultados modelos detección de esquinas

La imagen mostrada a continuación corresponde a la predicción del modelo entrenado utilizando Transfer Learning y aumentación de datos

3.c. Identificación del elemento de interés en la hoja

Para la la detección de palabras dentro de un documento, se optó por utilizar trasnfer learning utilizando la api de tensorflow object detection y hacer transferlearning a partir de un modelo del tensorflow zoo Gho.

Una vez seleccionada la herramienta para hacer transfer learning, se seleccionó Mobilnet como modelo base y partir de ahí se generaron expérimentos con diferentes hiperparámetros para mejorar el modelo inicial.

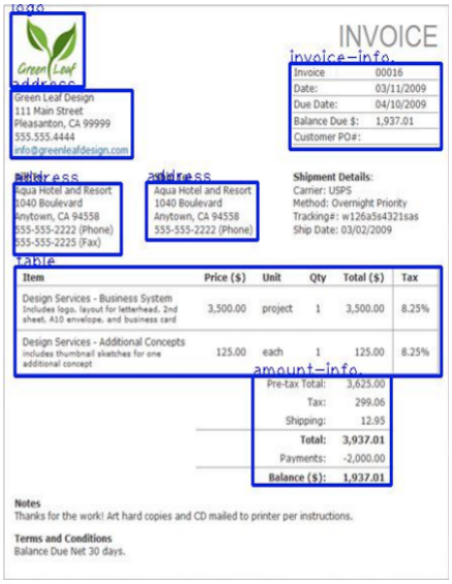
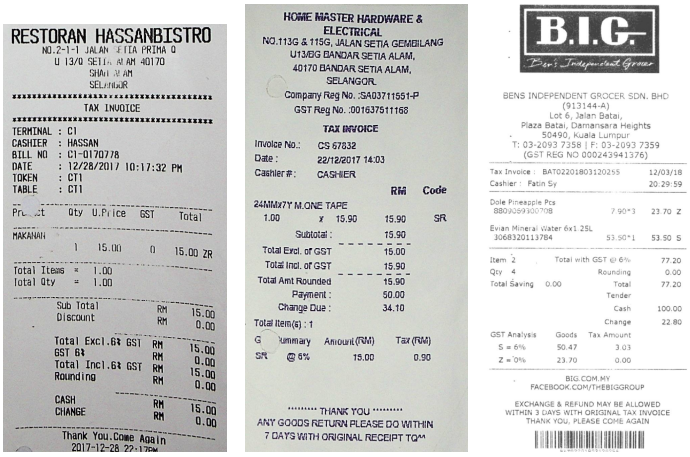


Figure 15: Invoice Example

3.c.1. Recolección de datos

La recolección de los datos se hizo descargando ejemplos de una página web que contiene imágenes de facturas y documentos Figura:16 muy parecidos a los que el modelo se puede encontrar en producción ver Dataset.



(a) Ejemplo 1 (b) Ejemplo 2 (c) Ejemplo 3

Figure 16: Ejemplo Dataset palabras

De esta fuente se recolectaron 500 imágenes y se etiquetaron las palabras usando la herramienta labelling Sax. Esta me permite generar un archivo XML donde se encuen-

tran las clases de las palabras a detectar, al realizar un procesamiento convertimos este un dataset Figura:17

filename	width	height	class	xmin	ymin	xmax	ymax	_xmin	_ymin	_xmax	_ymax
078.jpg	936	1841	Change	394	1579	551	1625	0.420940	0.857686	0.588675	0.882672
002.jpg	459	949	Change	14	728	109	755	0.030501	0.767123	0.237473	0.795574
049.jpg	1080	1527	Change	469	861	554	888	0.434259	0.563851	0.512963	0.581532
074.jpg	583	1303	Change	106	905	196	933	0.181818	0.694551	0.336192	0.716040
130.jpg	934	1927	Thanks	633	1827	868	1894	0.677730	0.948106	0.929336	0.982875

Figure 17: Dataset: Detección de palabra

3.c.2. Separación de datos

Una vez descargado y etiquetado el dataset, se hizo la separación a través de una funcion que aleatoriamente genera una etiqueta mencionando cada imagen a que set pertenece. La separación se hizo teniendo como objetivo tener el 30 porciento de las imágenes para pruiieba y 70 porciento para entrenamiento. Esta división se hizo en esa proporcion para mantener tantos datos como sea posible en entrenamiento, pero también para obtener un buen nivel de confianza a la hora de calcular las métricas.

3.c.3. Selección de métrica

Intersection Over Union (IOU) es una medida basada en el índice Jaccard(poner referencias) que evalúa la superposición entre dos cuadros delimitadores. Requiere un cuadro delimitador de entrenamiento y un cuadro delimitador predicho. Aplicando una división podemos saber si una detección es válida (verdadero positivo) o no (falso positivo).

El IOU [raf] viene dado por el área de superposición entre el cuadro delimitador predicho y el cuadro delimitador de verdad del terreno dividido por el área de unión entre ellos Figura:18:

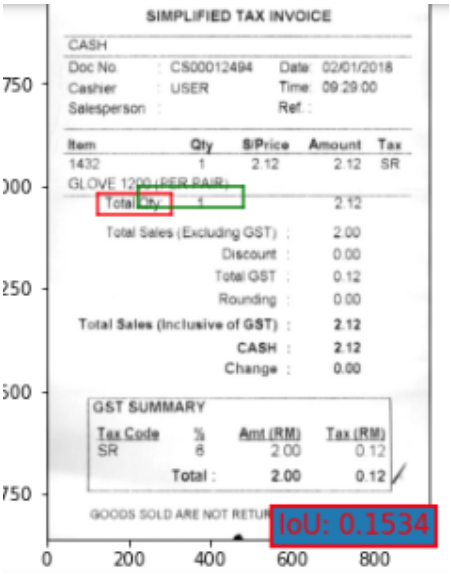


Figure 18: metrica IOU

Verdadero positivo, falso positivo, falso negativo y verdadero negativo

- Verdadero positivo (TP): una detección correcta. Detección con IOU >= threshold
- Falso positivo (FP): una detección incorrecta. Detección con IOU < threshold
- Falso negativo (FN): No detectada
- Verdadero negativo (TN): No la usaremos en los experimentos.
- threshold: Estableceremos 40%

3.c.4. Experimentos

Baseline

Como se menciiona anteriormente, el modelo base que se utilizó fue Mobilnet V2 224x224 que posee una estructura que incrementa la velocidad de entrenamiento. Las arquitectura que se utilizó para esta tarea es la siguiente, los experimentos realizados en este pueden encontrar en el repositorio del **Experimento Baseline** Notebooks .

Input	Operator	exp size	#out	SE	NL	s
$224^2 \times 3$	conv2d, 3x3	-	16	-	HS	2
$112^2 \times 24$	bneck, 3x3	16	16	✓	RE	2
$56^2 \times 24$	bneck, 3x3	72	24	-	RE	2
$28^2 \times 24$	bneck, 3x3	88	24	-	RE	1
$28^2 \times 40$	bneck, 5x5	96	40	✓	HS	1
$14^2 \times 40$	bneck, 5x5	240	40	✓	HS	1
$14^2 \times 40$	bneck, 5x5	240	40	✓	HS	1
$14^2 \times 40$	bneck, 5x5	120	48	✓	HS	1
$14^2 \times 48$	bneck, 5x5	144	48	✓	HS	1
$14^2 \times 96$	bneck, 5x5	288	96	✓	HS	2
$7^2 \times 96$	bneck, 5x5	576	96	✓	HS	1
$7^2 \times 96$	bneck, 5x5	576	96	✓	HS	1
$7^2 \times 96$	conv2d, 1x1	-	576	✓	HS	1
$7^2 \times 576$	Pool, 7x7	-	-	-	HS	7
$1^2 \times 576$	conv2d 1x1	-	1280	-	HS	1
$1^2 \times 1280$	conv2d 1x1	-	k	-	HS	1

Figure 19: Architecture

Los parámetros mas relevantes que se utilizaron para el entrenamiento de este modelo fueron los siguietes:

- input shape: 224x224
- num classes: 4
- batch size: 16
- optimezer: Adam

Mostramos algunas Figuras : 20 de detecciones encontradas con el Baseline:

(a) *DetecciónBaseline*1

(b) *DetecciónBaseline*2

(c) *DetecciónBaseline*3

Figure 20: Ejemplos Detección Baseline

Data Augmentation

El aumento de datos es una estrategia que permite aumentar significativamente la cantidad de datos disponibles para los modelos de entrenamiento, sin tener que recopilar datos nuevos.

Características	Valor
rotations	min =10,max=20
flip	lower=0.5, upper=1.5
Gamma	max_delta=0.2
Hue	lower=0.5, upper=1.5

Table 6: Transformaciones

Al realizar el aumento de datos convertimos nuestras 500 imágenes en 1500. utilizamos, rotaciones, flip , Gamma , Random gamma Tabla : 6 Bus+20 ver resultados **Experimento Data Augmentation** Notebook. Al realizar el aumento de datos obtenemos Figura : 21.

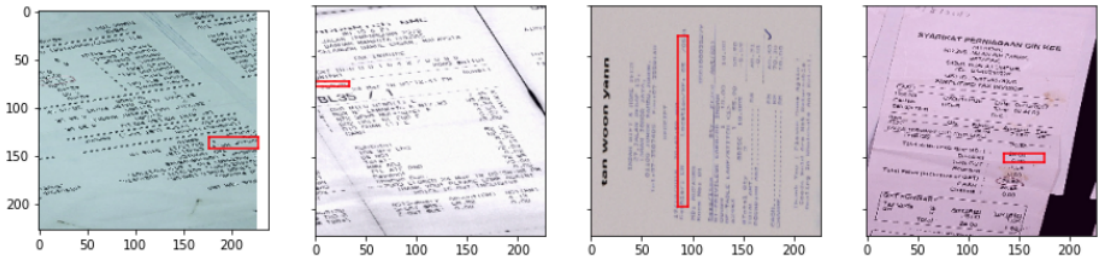
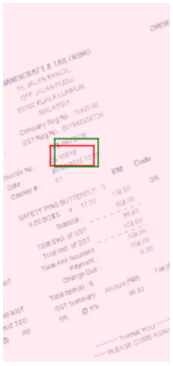
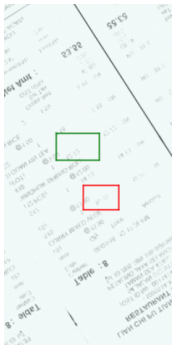


Figure 21: Resultados Data Augmentation

Mostramos algunas Figuras :22 de detecciones encontradas con la aumentación de datos:



(a) *AumentaciónndeDatos*1



(b) *AumentaciónndeDatos*2

(c) *AumentaciónndeDatos*3

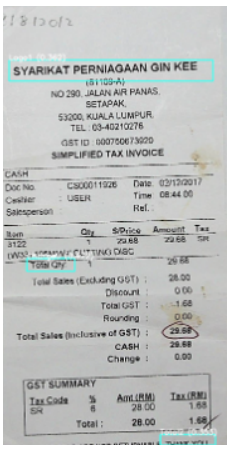
Figure 22: Ejemplos Detección Data Augmentation

Transfer Learning

TensorFlow 2 [Zan] proporciona una API de detección de objetos que facilita la construcción, el entrenamiento y la implementación de modelos de detección de objetos. En este proyecto, usaremos esta API y entrenaremos el modelo que se encuentra en el **repositorio de Git** en este encontraran un pipeline donde se encuentran todos lo parámetro de entrenamiento y los pasas para obtener le modelo.

Mostramos algunas Figuras :23 de detecciones encontradas con Transfer Learning.

(a) *TransferLearning*1



(b) *TransferLearning*2

(c) *TransferLearning*3

Figure 23: Ejemplos Transfer Learning

Resultados de Experimentos

La tabla : 5 muestra los resultados de entrenamiento de cada uno de los experimentos que se encuentra el repositorio. Tomamos como base las implementaciones realizadas. Podemos ver un resumen de los resultados de los experimentos en la tabla:7

Experimentos	mAP_{40}	IoU ≥ 0.4
Baseline	23%	No
Aumentacion de datos	35%	No
Transfer Learning	39%	Yes

Table 7: Resultados

4. Despliegue

Un servicio de machine learning o deep learning desplegado en un servidor de desarrollo [Mav], accesible a los desarrolladores para testear los modelos implementados, arquitectura del prototipo Figura: 24

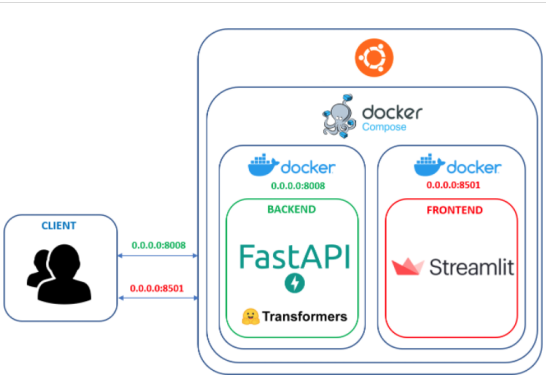


Figure 24: Arquitectura de Prototipo

El prototipo consta de:

- Una interfaz de usuario (UI) para que dichos usuarios puedan interactuar con el servicio usaremos streamlit Figura:27.
- El servicio backend con documentación API, de modo que pueda procesar múltiples solicitudes y pasarse a producción e integrarse con otras aplicaciones de manera robusta y escalable para esto utilizaremos FastApi.

Docker

Proporcionamos un prototipo de Docker en el cual pueden ver la aplicación prototipo **repositorio de Git**.Encontrará todos los pasos , para poner en marcha en contendor que contiene le prototipo.

Prototipo-Interfaz

- Al desplegar el contenedor encontrara una instancia local http://localhost:5000/ donde se encontrara la interfaz mostrada figura: 27
- El prototipo se le pueden incluir imágenes en formato jpg
- Seleccione el tipo de modelo WORD/POINTS/SHEET DETECTTION
- Ingrese la imagen
- El modelo empezar el proceso de carga y desplegara el resultado

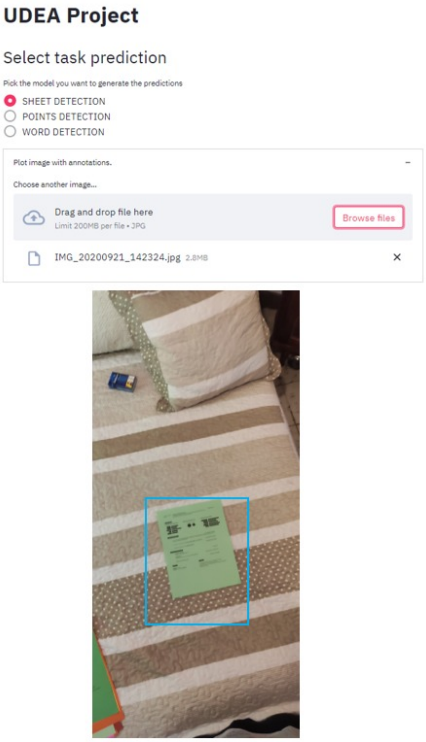


Figure 25: Prototipo localización hoja

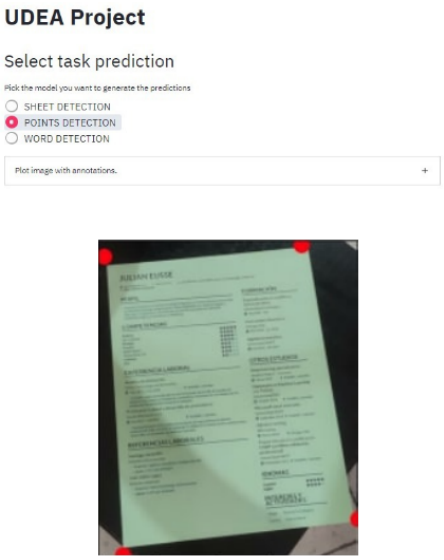


Figure 26: Prototipo detección de esquinas

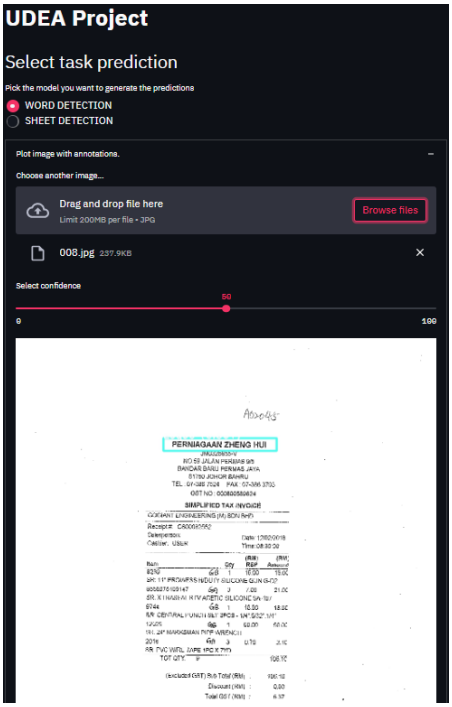


Figure 27: Prototipo detección de palabra

5. Conclusiones

- Se evidenció el impacto potitivo al usar 'DataAugmentation' en los resultados de clasificación y detección, ya que no se tuvo acceso a grandes volumenes de datos y se contaba con no mas de cientos de imagenes.
- Fue muy importante utilizar callbacks y herramientas de visualización como Tensorboard para registrar métricas a medida que se avanzaba en el entrenamiento y asi garantizar que el modelo estuviera aprendiendo y además que no estuviera sobre ajustandose
- Al no tener acceso a grandes volumenes de datos ni a gran capacidad de computo, realizar TransferLearning en el proceso de entrenamiento fue muy importante para ver buenos resultados en corto tiempo.
- En el problema de reconocimiento de esquinas de hojas, para llegar a resultados muy precisos se requieiren decenas de miles de imagenes más de las que utilizamos. Para estos experimentos se usaron imágenes con poca varición entre ellas en colores, inclinación de la hoja, fondos y luces para poder hacer un experimento de entrenamiento donde un modelo pequeño y sin acceso a muchas imagenes pueda aprender a reconocer esquinas en una distribución de imágenes no tan variada.
- En la tabla 5 de métricas de detección de la palabra se puede ver una mejora en la entapa de entrenamiento según los experimentos realizados. Al utilizar aumento de datos notamos que la métrica no mejoro. Los tipos de imágenes utilizados no dieron la suficiente variabilidad al modelo. Al utilizar la API de Tensorflow notamos una mejora considerable esto es debido a que utilizamos los pesos de modelo pre entrenados para estas tareas es especifico lo que nos da una mejora considerable.
- Utilizar imágenes y crear un Dataset fue una tarea exhaustiva debido a que las imágenes necesitan variabilidad por ejemplo ángulo de la foto, luz o segmentación. Al realizar la toma de fotos y realizar unas probas nos dimos cuenta de que estábamos tomando las imágenes desde el mismo punto o ángulo. Esto hizo que nuestros modelos aprendieran la misma posición del objeto a detectar (overfitting)

6. Referencias

References

[Bus+20] Alexander Buslaev et al. "Albumentations: Fast and Flexible Image Augmentations". In: *Information* 11.2 (2020). ISSN: 2078-2489. DOI: 10.3390/info11020125. URL: https://www.mdpi.com/2078-2489/11/2/125.

[Gho] Sourav Ghosh. *Invoice Information extraction using OCR and Deep Learning*. URL: https://medium.com/analytics-vidhya/invoice-information-extraction-using-ocr-and-deep-learning-b79464f54d69. (accessed: 01.05.2016).

[Mav] Amol Mavuduru. *How you can quickly deploy your ML models with FastAPI*. URL: https://towardsdatascience.com/how-you-can-quickly-deploy-your-ml-models-with-fastapi-9428085a87bf. (accessed: 18.04.2021).

[raf] rafaelpadilla. *Object-Detection-Metrics*. URL: https://github.com/rafaelpadilla/Object-Detection-Metrics. (accessed: 07.09.2018).

[Sax] Pranjal Saxena. *How to Label Images for Object Detection, Step by Step*. URL: https://towardsdatascience.com/how-to-label-images-for-object-detection-step-by-step-7ee317f98583. (accessed: 07.09.2020).

[Zan] Hugo Zanini. *Custom object detection in the browser using TensorFlow.js*. URL: https://blog.tensorflow.org/2021/01/custom-object-detection-in-browser.html. (accessed: 01.09.2021).