Sistema de Reconocimiento de Dígitos Impresos para Agilizar el Registro de Facturas Emitidas por SUNAT.

Printed digit Recognition System to Speed Up the Registration of Invoices Issued by SUNAT.

Andres Benjamin Rojas Alza[1], Jazmin Aracelly Reyes Guzman[1], Victor Felipe Alza Tesén[2] y Dayana Selene Reyes Guzman[3]

[1]Universidad Nacional de Trujillo, La Libertad – Perú
[2]Universidad Nacional Pedro Ruiz Gallo, Lambayeque – Perú
[3]Universidad Cesar Vallejo, Trujillo – Perú

RESUMEN

Actualmente, la creciente fichas de facturas en los centros, es uno de los factores muy comunes en la demora al asignar a los sistemas de registro los valores numéricos en la base de datos. Debido a esto, el objetivo principal del proyecto se basa en diseñar e implementar un sistema de reconocimiento de dígitos impresos para identificar y clasificar con precisión los dígitos del 0 al 9. Enfocándonos por el aprendizaje automático, siendo un método eficaz de clasificación y obtener con precisión y rapidez, mediante el sistema de visión por computadora, la comprobación, funcionamiento y comparación de los resultados de manera automática, para esto se realiza la extracción de detalles en base de variedad de dígitos impresos, utilizando red neuronal convolucional con en el procesamiento de imágenes del conjunto de datos de TMNIST basado en el muy conocido dataset MNIST enfocado en dígitos impresos.

Palabras clave: Dígitos impresos, Deep Learning, Procesamiento de Imágenes, TMNIST.

ABSTRACT

Currently, the increasing number of invoice tokens in the centers is one of the very common factors in the delay in assigning numerical values in the database to the registration systems. Due to this, the main objective of the project is based on designing and implementing a printed digit recognition system to accurately identify and classify digits from 0 to 9. Focusing on automatic learning being an effective method of classification and obtaining accurately and quickly, through the computer vision system, the verification, operation and comparison of the results automatically, for this the extraction of details based on variety of printed digits, using convolutional neural network with image processing of the TMNIST dataset based on the well-known MNIST dataset focused in printed digits.

KEY WORDS: Printed digits, CNN, Deep Learning, Image Processing, TMNIST.

INTRODUCCION

En la visión por computadora respecto al sistema de reconocimiento de caracteres al extraer de una imagen ciertos caracteres que contiene dígitos impresos para almacenarlos a través de grandes formatos de texto. Entonces se debería convertir el formato de texto descrito por un valor en cada uno de los píxeles de la imagen.

Actualmente, en el país se obtienen cada año problemas respecto a la demora en diversidad de tiendas, negocios, bancos, centros, locales, entre otras, en cuanto al poder transcribir datos de formularios, facturas, cheques bancarios y números de identificación hacia una computadora. La cual en procesos de transcribir llega ser muy lento y demasiado tiempo para en los locales de mayor aglomeración; algunos problemas de transcripción mayores en alguna equivocación en el proceso, sobre todo en los números. Según (Espejo at el.,2017) menciona que, en el Perú, los trabajadores nombrados digitadores pueden ganar entre 850 y 2000 soles mensuales en un trabajo de tiempo completo con respecto a sus tareas asignadas, mientras los trabajadores en tiempo parcial aproximadamente una ganancia de 400 soles. Por ello, el proyecto tiene como finalidad agilizar el llenado de la base de datos y evitar errores por las personas que pueden suceder al transcribir los datos a la computadora. Para así obtener una optimización de datos, resultados apropiados y más accesible en el momento que el trabajador coloque en el formulario los datos con el simplemente de tomar foto u obtener las imágenes y procesarlos mediante algoritmos hacia la computadora y se complete el llenado de la base de datos.

REALIDAD PROBLEMÁTICA

La documentación y registro de información contenida en facturas emitidas por la SUNAT o cualquier otra entidad puede ser un proceso manual, tedioso y propenso a errores. La tarea de ingresar manualmente los datos relevantes impresos en las facturas, como el número de RUC, la fecha, el número de serie, el número de la factura, el importe, la base y el IGV, consume tiempo y puede resultar en errores de transcripción. Esto puede afectar la eficiencia, la precisión y la automatización de los procesos relacionados con el registro de facturas y el manejo de información financiera.

Según Yusit (2019), si bien se destaca que la implementación de la facturación electrónica ha demostrado ser una solución eficiente para reducir errores y minimizar los riesgos y daños significativos que pueden surgir en una empresa al registro de la documentación, todavía existe la posibilidad de cometer errores humanos. Siempre se busca la disminución de estos errores, pero es crucial tener en cuenta que la intervención humana en el proceso puede generar equivocaciones.

Además, si hay una gran cantidad de facturas que deben ser procesadas, el tiempo y los recursos necesarios para ingresar manualmente los números impresos se vuelven significativos, lo que puede retrasar la documentación y la toma de decisiones basadas en esa información, en relación según el informe estadístico por la SUNAT (SUNAT, 2023). Mostrados en el Anexo B y C.

A partir de esto, surge la necesidad de automatizar y agilizar el proceso de identificación y registro de los números impresos en las facturas. Un sistema de reconocimiento de números mediante visión por computadora puede proporcionar una solución eficiente, precisa y

automatizada para extraer y registrar la información contenida en las facturas, mejorando la eficiencia de los procesos y reduciendo la posibilidad de errores humanos.

IMPORTANCIA DEL PROYECTO

La realización del proyecto del Sistema de reconocimiento de dígitos impresos con el dataset TMNIST tiene como finalidad la contribución de reconocer patrones en imágenes para identificar y digitalizar los números en estas imágenes sin la necesidad de estar tipeando manualmente, sino simplemente detectando los números a través de la cámara. Con la identificación instantánea de los números, los cuales representan información relevante en las facturas emitidas por la SUNAT, se permitirá que se realice una buena documentación y un registro más eficiente y automatizado.

OBJETIVOS

1. **General:** Diseñar e implementar un sistema de reconocimiento de números impresos en facturas emitidas por la SUNAT, utilizando técnicas de visión por computadora y aprendizaje automático.
2. **Específicos:**
   - Manejar un conjunto de datos de facturas de compras para su posterior procesamiento y entrenamiento del modelo.
   - Desarrollar el preprocesamiento de las imágenes de las facturas para mejorar su calidad y facilitar la segmentación de los números.
   - Diseñar y emitir el modelo utilizando el conjunto de datos recopilado y optimizar sus parámetros para mejorar la precisión del reconocimiento de números.
   - Evaluar la efectividad del sistema de reconocimiento de números en facturas mediante la aplicación de pruebas utilizando un conjunto de datos separado.

ANTECEDENTES

Así mismo tenemos antecedentes algunos trabajos enfocados en la utilización de CNN (red neuronal convolucional) y el reconocimiento de dígitos y caracteres.

Como primer antecedente tenemos la "Una metodología para el reconocimiento de caracteres y revisión del procedimiento de resolución de ecuaciones lineales" (Guevara Neri et al., 2023). Este artículo habla sobre como desarrollaron un sistema para el reconocimiento automático de caracteres y revisión de procedimientos de resolución de ecuaciones lineales escritas a mano. El proceso incluía la adquisición de imágenes mediante una cámara, preprocesamiento, segmentación de caracteres y líneas de ecuación, y el uso de una red neuronal convolucional (CNN) para el reconocimiento de caracteres. El sistema se evaluó con un conjunto de datos de 2800 imágenes (2100 para entrenamiento y 700 para pruebas), que incluían diez dígitos y cuatro símbolos matemáticos. Los resultados mostraron una precisión del 99% en el reconocimiento de caracteres escritos a mano.

Como segundo antecedente tenemos la "OCR-Redes: variantes de CNN pre entrenadas para el reconocimiento de caracteres escritos a mano en urdu a través del aprendizaje por transferencia" (K.o & Poruran et. al, 2020). Este artículo tuvo como objetivo desarrollar en el OCR-Nets con variantes de (AlexNet y GoogleNet) para el reconocimiento de caracteres urdu escritos a mano a través del aprendizaje por transferencia. La cual se comparó la tasa

de reconocimiento con los métodos tradicionales de reconocimiento de caracteres y garantizar la imparcialidad del experimento, generando manualmente un conjunto adicional de datos de caracteres urdu con diferentes fuentes y tamaño. Teniendo como resultados experimentales muestran el OCR-AlexNet y OCRGoogleNet un alto rendimiento, con tasas de éxito promedio del 96,3% y 94,7%, respectivamente para confirmar la efectividad de las redes propuestas en el reconocimiento de caracteres urdu escritos a mano.

Como tercer antecedente tenemos el "Método mejorado de reconocimiento de dígitos escritos a mano probado en la base de datos MNIST" (Kussul & Baidyk et al, 2004). Este artículo tuvo como objetivo desarrollar un método mejorado de reconocimiento de dígitos escritos a mano utilizando un clasificador neuronal de Área Receptiva Limitada (LIRA). El método se probó en la base de datos MNIST y en una base de datos de microdispositivos de ensamblaje. El clasificador LIRA constaba de capas de neuronas: sensora, asociativa y de salida, con la capa del sensor conectada a la capa asociativa sin cambios aleatorios modificables. Los resultados mostraron una tasa de reconocimiento del 99,41%, superando a otros clasificadores probados en la base de datos MNIST. Además, en la aplicación de micro ensamblaje, se logró una posición aproximada del agujero de alfiler con tolerancias de 1 a 2 píxeles, con un tiempo de procesamiento de 0,5 segundos en un Pentium III de 500 MHz.

MATERIALES Y MÉTODOS

1. **Data set con TMNIST**

   Este dataset está inspirado en la base de datos MNIST (Base de datos de la Organización Nacional de Puntos de Referencia e Innovación Modificada) para dígitos escritos a mano. Consiste en imágenes que representan dígitos del 0 al 9, generados utilizando 2.990 archivos de fuentes de Google. Este conjunto de datos se tomó de Kaggle y está disponible para la investigación a través del enlace: https://www.kaggle.com/datasets/nimishmagre/tmnist-typefacemnist.

   El dataset consta de un único archivo en formato csv, el cual contiene 29.900 ejemplos con etiquetas y nombres de fuentes. Cada fila contiene 786 elementos: el primer elemento representa el nombre de la fuente (por ejemplo, Chivo-Italic, Sen-Bold), el segundo elemento representa la etiqueta (un número del 0 al 9) y los 784 elementos restantes representan los valores de píxeles en escala de grises (de 0 a 255) para la imagen de 28x28 píxeles. (Bojer et al.,2021). Visualizados en el **Anexo A.**

2. **Deep Learning**

   El deep learning es parte del campo de machine learning para procesar grandes cantidades de datos de manera no lineal utilizando redes neuronales con múltiples capas, se enfoca en redes neuronales para obtener y extraer características desde los datos y poder realizar predicciones muy detalladas y precisas desde los grandes volúmenes de datos la cual se logra por las capas de la arquitectura, donde cada capa recibe la información y pasa a la siguiente capa, así poder tener la red con más características complejas.(Wang et al., 2021).

   El deep learning siendo una subcategoría del aprendizaje automático en la cual las redes neuronales se tienen que organizar en niveles secuenciales, aquí a partir de cada nivel tiende a tener las salidas del nivel anterior como entrada durante el aprendizaje para poder identificar y clasificar objetos con un mínimo preprocesamiento como se demuestra en tareas de reconocimiento visual que se obtiene las características importantes de las imágenes de entrada. (Lee et al., 2022).

3. **CNN(Convolutional Neural Network)**

   Las redes neuronales convolucionales son una arquitectura de redes neuronales profundas especialmente diseñada para el procesamiento de imágenes y datos estructurados en forma de cuadrícula, sus datos de entrada se filtran de detalles innecesarios, la cual nos permitirá un procesamiento posterior solo de información útil, ya que resulta efectivamente en el reconocimiento de objetos. Entonces, en la arquitectura de las redes neuronales convolucionales fue introducida por primera vez en 1998 por Jan Lekun (arquitectura LeNet). Se prestó especial atención a las redes convolucionales sólo después de 14 años, cuando en 2012 la arquitectura de AlexNet (LeNet modificada y mejorada) ganó el concurso ImageNet por un amplio margen. Por ello, actualmente las redes neuronales convolucionales se han utilizado activamente con datos visuales (e Silva et al., 2023).

   Las CNNs siendo una herramienta muy importante en el campo de la visión por computadora y el procesamiento de imágenes, ya que permiten obtener características y patrones relevantes a partir de los datos para la clasificación, estando agrupadas en las capas de procesamiento para reducir la dimensionalidad en diversos niveles de abstracción y la detección de objetos de manera precisa y eficiente (Gan et al., 2023).

4. **Jupyter**

   Es muy útil la herramienta porque desarrolla el software mediante código abierto, brindando servicios para la computación interactiva. Así mismo está conformado por varios lenguajes de programación y se basa en iPython. El proyecto utilizó la herramienta por el lenguaje de programación Python. (Jupyter, 2022).

5. **Numpy**

   NumPy es una librería fundamental para la computación científica en Python. Proporciona una poderosa estructura de datos llamada arreglo (array) multidimensional, junto con funciones para operar en estos arreglos. NumPy es eficiente para el almacenamiento y manipulación de grandes conjuntos de datos numéricos. Se utiliza para realizar operaciones matemáticas y numéricas, como cálculos estadísticos, álgebra lineal y manipulación de arreglos. Puedes utilizar NumPy para procesar y manipular las imágenes de tus facturas en formato de arreglos numéricos. (NumPy, 2023.)

6. **Tensor Flow**

   TensorFlow es una plataforma de código abierto para el aprendizaje automático (machine learning) y el aprendizaje profundo (deep learning). Proporciona un conjunto completo de herramientas y bibliotecas para construir y entrenar modelos de aprendizaje automático, incluyendo redes neuronales profundas. TensorFlow es especialmente útil cuando se trabaja con grandes conjuntos de datos y se necesita un alto rendimiento en el entrenamiento y la inferencia de modelos. (Tensorflow, 2022)

7. **Pandas**

   Pandas es una librería de Python que proporciona estructuras de datos y herramientas para el análisis de datos. Se utiliza para manipular y analizar datos tabulares de manera eficiente. Pandas proporciona el objeto DataFrame, que es una estructura de datos bidimensional que puede contener datos etiquetados y heterogéneos. Esta librería es útil para la carga, limpieza, transformación y

exploración de datos. Puedes utilizarla para manejar y procesar los datos de tus facturas de manera conveniente. (Pandas, 2023)

8. **Matplotlib**

Matplotlib es una librería de visualización de datos en 2D en Python. Proporciona una amplia variedad de gráficos y visualizaciones para representar datos de manera efectiva. Matplotlib se utiliza para crear gráficos, diagramas y figuras personalizadas. Puedes utilizar esta librería para visualizar tus datos, generar gráficos estadísticos o mostrar imágenes procesadas en tu proyecto. (Matplotlib, 2023)

9. **Métricas de evaluación:**

9.1. Matriz de Confusión: En el área de inteligencia artificial y aprendizaje automático, esta es una herramienta que se utiliza para visualizar el rendimiento de los algoritmos de aprendizaje supervisado.

En esta matriz cada columna representa el número de predicciones para cada clase y cada fila representa las instancias en la clase real. Para una matriz de confusión de 2x2 se evalúa de la siguiente forma:

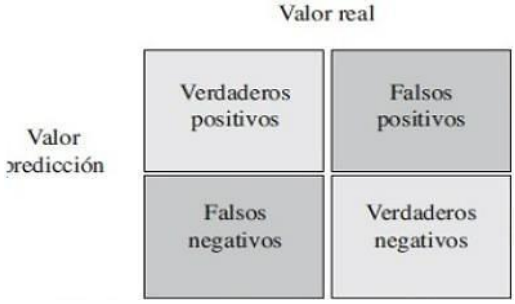


Figura 1: Matriz de confusión 2x2

En la **Fig. 1** se representa los 4 posibles valores, debido a que surgen un par de posibles valores reales y un par de posibles valores de predicción (Barrios Arce, 2022).

9.2. Accuracy: Se enfoca en lo que se estima el resultado de una medición del valor verdadero hacia lo relacionada con el sesgo de una estimación y es representada como la proporción de resultados verdaderos (verdaderos positivos y verdaderos negativos) dividido entre el número total de casos examinados (verdaderos positivos, falsos positivos, verdaderos negativos, falsos negativos). (Barrios Arce, 2022).

En la práctica, el accuracy o exactitud es la cantidad de predicciones positivas que son correctas, como se muestra en la **Fig. 2.**

$$Accuracy = \frac{TrueNegatives + TruePositive}{TruePositive + FalsePositive + TrueNegative + FalseNegative}$$

-

Figura 2:
Fórmula del Accuracy

CASO DE ESTUDIO

1. Análisis Inicial

Para empezar con la planificación de nuestro proyecto debemos definir correctamente el alcance que tendrá este:

- Identificación de dígitos impresos: El proyecto se enfocará en la identificación de dígitos impresos en imágenes de facturas utilizando técnicas de reconocimiento de patrones. El objetivo es extraer y reconocer los números relevantes, como el número de RUC, la fecha, el número de serie, el número de la factura, el importe, la base y el IGV.
- Ingreso de imágenes: El sistema permitirá analizar capturas de imágenes de facturas que se hayan tomado a través de la cámara de un dispositivo. Los números impresos en la factura serán detectados y reconocidos directamente desde la imagen capturada, sin la necesidad de ingresarlos manualmente.
- Automatización del registro de datos: La identificación instantánea de los números en las facturas permitirá una documentación más eficiente y automatizada. Los números reconocidos podrán ser entregados en un formato que facilitará y reducirá la carga de trabajo manual y minimizará los errores humanos.

Además, nuestro proyecto cuenta con las siguientes limitaciones:

- El rendimiento del sistema estará influenciado por la calidad de las imágenes de las facturas capturadas. Imágenes borrosas, mal iluminadas o con distorsiones pueden afectar la precisión del reconocimiento de números.
- La dificultad de realizar una correcta segmentación de los números impresos en las facturas, especialmente cuando los números están cerca de otros elementos gráficos o están superpuestos.
- La eficiencia y el rendimiento del sistema para garantizar un procesamiento rápido y preciso de los números en las imágenes capturadas.

El modelo general que seguirá nuestro proyecto para el desarrollo del modelo computacional será el siguiente:

Figura 3: Modelo General antes del escenario

Figura 4: Modelo General después del escenario

La cual se comprenderá el funcionamiento del software en unos simples pasos para el análisis de imágenes de dígitos impresos, la clasificación y segmentación de dígitos impresos en las facturas de la SUNAT es muy importante como se muestran en la **Fig. 3 y Fig. 4.**

2. Diseño de alto nivel

El modelo general que seguirá nuestro proyecto para el desarrollo del modelo computacional será el siguiente, el cual lo representamos como un diagrama de bloques **Fig.5.**

Figura 5: Diseño de alto nivel del proyecto.

En primera instancia se debe dar la adquisición de imágenes a través de dispositivos de captura de imágenes, como cámaras o escáneres. Luego se debe preparar adecuadamente las imágenes antes de la clasificación. Luego se busca la extracción de características de las imágenes. Esto podría incluir métodos como la detección de bordes, el cálculo de histogramas o la aplicación de filtros para resaltar características relevantes. Para la clasificación se implementará una red neuronal con la cual se procederá al etiquetado de cada clase que se están reconociendo en el proyecto. Finalmente, se mostrará el resultado donde la salida del sistema, será una etiqueta que indica el número reconocido junto a la imagen de entrada correspondiente.

3. Diseño detallado

3.1. Adquisición de imágenes: Utilizar dispositivos de captura de imágenes, como cámaras o escáneres, para obtener imágenes de las facturas impresas. Lo esencial de este proceso es obtener las imágenes con la mayor calidad posible.

3.2. Preprocesamiento de imágenes: Se aplican una serie de técnicas para mejorar la calidad y la legibilidad de las imágenes antes de utilizarlas para el procesamiento y la clasificación de los datos impresos.

Filtrado y morfología: Se pueden aplicar filtros para resaltar los detalles importantes de los números impresos, eliminar el ruido y mejorar el contraste. Por ejemplo, se pueden usar filtros de realce de bordes o filtros de mejora de contraste para mejorar la visibilidad de los números, además, así como técnicas dilatar o erosionar para obtener los números más detallados posibles.

Identificación de regiones de interés: En el caso de las facturas, que contienen información relevante en diferentes secciones, es necesario identificar las regiones de interés que contienen los datos impresos relevantes. Esto implica utilizar técnicas de segmentación para separar las diferentes secciones de la factura y detectar las regiones que contienen los datos importantes, como el número de RUC, la fecha, el número de serie, etc.

Recorte de las regiones de interés: Una vez identificadas las regiones de interés, se puede recortar esas secciones de la factura para aislar los datos impresos relevantes.

3.3. Convertir las imágenes preprocesadas a una representación adecuada:

Para el procesamiento y la clasificación de los números, las redes neuronales, como las CNN, requieren una entrada en un formato específico. Para preparar las imágenes para el procesamiento y la clasificación, se realizan una serie de pasos de preprocesamiento, que pueden incluir:

Normalización: Asegurarse de que los valores de los píxeles están en un rango específico, para facilitar el procesamiento y evitar problemas de escala.

Redimensionamiento: Ajustar el tamaño de las imágenes a un tamaño común, por ejemplo, redimensionar todas las imágenes a una resolución de 28x28 píxeles.

Conversión de imágenes a escala de grises: Si las imágenes originales están en color, se pueden convertir a escala de grises para simplificar el procesamiento y reducir la complejidad del modelo.

Aplanamiento de la imagen: Transformar la matriz de píxeles en un vector unidimensional. En el caso de imágenes de 28x28 píxeles, esto implicaría convertir la matriz de 28x28 en un vector de longitud 784.

3.4. Diseño y entrenamiento del modelo de reconocimiento:

Diseño adecuado de la red neuronal: Definir la arquitectura de la red neuronal, definiendo las capas y funciones de activación adecuadas como se muestra la **Fig. 6.** Visualizado en el **Anexo D**.

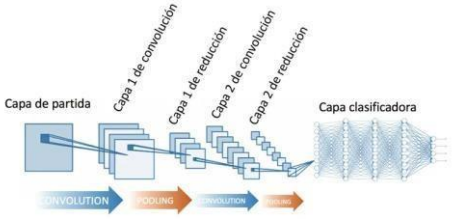


Figura 6: Esquema general de las capas de la CNN.

Capa de Convolution2D: Es la capa convolucional que realiza la operación de convolución en las entradas de la red (imágenes), emplea diversos filtros en paralelo para extraer diferentes características de la entrada, para cada uno de los filtros detecta patrones específicos, como bordes, texturas o formas en la imagen para obtener imágenes o mapas de características que resaltan las regiones importantes de la entrada.

Capa de MaxPooling2D: Es la capa de la operación de convolución agrupamiento (pooling) se utiliza para reducir la dimensionalidad espacial de la representación de una imagen, la cual toma una ventana

(kernel) para extraer el valor máximo en cada región seleccionada y obtener una nueva representación reducida. Así mismo, reduce la cantidad de parámetros y operaciones en la red, lo que puede ayudar a evitar el sobreajuste y acelerar el entrenamiento y mejorando el rendimiento general del modelo.

Capa de Flatten: Es la capa especial que se representa después de las capas convolucionales y de agrupamiento, la cual se utiliza para transformar el volumen 3D de datos (matriz de imágenes) en un vector 1D antes de conectarlo a las capas densamente conectadas (fully connected layers) de la red como una secuencia lineal de valores (unidimensional).

Capa de Dense, Fully Connected o capa completamente conectada: Es una capa que sigue a las capas convolucionales y de agrupación para extraer características y patrones de la entrada y poder realizar la clasificación y tomar decisiones basadas en esas características extraídas. Siendo compuesta por neuronas o nodos que están completamente conectados con los nodos de la capa anterior para aplicar una función de activación para producir una salida.

La fórmula general de la capa Dense es la siguiente:

$$\text{Salida} = \text{Función\_Activación}(W * \text{Entrada} + b)$$

Donde:

"W": Matriz de pesos

"b": vector de sesgos (biases)

"Función_Activación": Función no lineal que introduce no linealidad en la capa

- Capa de Dense: Es una capa que se encuentra entre capas totalmente conectadas, antes de la capa de salida. Su función es evitar el sobreajuste (overfitting) en la red neuronal, la cual reduce aleatoriamente un porcentaje de las neuronas (unidades) en la capa anterior durante el entrenamiento, consiste en desaparecer ciertas neuronas temporalmente durante el entrenamiento teniendo un rango de 0.2 a 0.5 (20% a 50%) lo que conduce a una mejor generalización y un mejor rendimiento en el conjunto de prueba.(Rafiee & Farhang, 2023)

Dividir el dataset: uno para entrenar el modelo y otro para evaluar su rendimiento. Esto es importante para evaluar la capacidad de generalización del modelo

Alimentar las imágenes de entrenamiento al modelo y ajustar los pesos utilizando algoritmos de optimización, como el descenso de gradiente.

Realizar iteraciones hasta alcanzar una precisión aceptable: La precisión se evalúa comparando las predicciones del modelo con las etiquetas reales de las imágenes.

3.5. Evaluación del modelo:

- Utilizar el conjunto de pruebas para evaluar el rendimiento del modelo: Una vez entrenado el modelo utilizando el conjunto de entrenamiento, se utilizan las imágenes del conjunto de prueba como entrada y obtener las predicciones del modelo para cada imagen.

Mostrados en el **Anexo E**. Teniendo como resultados:

- Training loss: 0.003150217467918992
- Training accuracy: 0.9991434812545776
- Validation loss: 0.05833325535058975
- Validation accuracy: 0.9914529919624329

- Calcular métricas de evaluación: Para medir el rendimiento del modelo, utilizar métricas como la precisión, el recall y la matriz de confusión, para medir la capacidad del modelo para clasificar correctamente los números impresos.

3.6. Visualización de resultados: Mostrar los resultados obtenidos por el modelo, como las imágenes de entrada, junto con las etiquetas predichas por el modelo. Visualizar métricas de evaluación para comprender el rendimiento del modelo.

RESULTADOS

1. Resultados Obtenidos

El modelo evaluado se utilizó una red neuronal convolucional (CNN), ya que esta es ideal para la clasificación de imágenes, en nuestro caso, detectar dígitos impresos del 0 al 9. Además, CNN tiene asociado el uso compartido de parámetros, esto hace que el número de parámetros se reduzca teniendo como consecuencia directa la disminución de los cálculos.

2. Discusión

Luego de entrenar la red neuronal CNN, pre entrenada con el conjunto de datos de TMNIST basado con el dataset MNIS se obtuvieron las gráficas de pérdida y precisión del modelo mostradas en el **Anexo F**

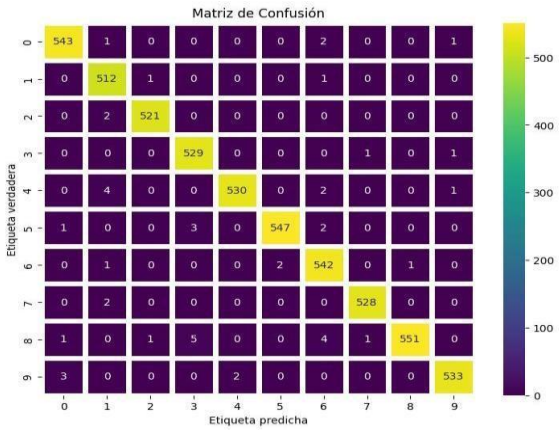
El modelo entrenado presenta los siguientes resultados para precisión y pérdida:

Modelo	Accuracy	Loss
TMNIST	0.991	0.058

Tabla 1. Resultados del modelo entrenado con las imágenes de dígitos impresos

Como se visualiza en la **Tabla 1,** el modelo perteneciente a las imágenes de las facturas de la SUNAT, tiene una precisión y una pérdida se puede afirmar que al utilizar la normalización en las imágenes de las facturas dígitos impresos se logra un mejor resultado con respecto a la red neuronal que se procedió al testeo, para este caso se utilizó un 10% del total de imágenes del dataset, y evaluación de la precisión de cada uno de los modelos previamente mencionados.

También, se consideró el uso de la matriz de confusión para la evaluación gráfica con respecto a los resultados del rendimiento del modelo.



- **Figura 7.** Matriz de confusión del modelo

En la **Fig. 7** se muestra la matriz de confusión del modelo y se puede resaltar el número de aciertos y errores al momento de la predicción. En síntesis, se tiene un total de 5336 aciertos y 46 predicciones erróneas, Con respecto a estos resultados, se puede evidenciar una relación directa con los resultados obtenidos en la **Tabla 1**, los dígitos "8" son los que ha tenido más acierto en clasificarlos y los que menos son los dígitos "1".

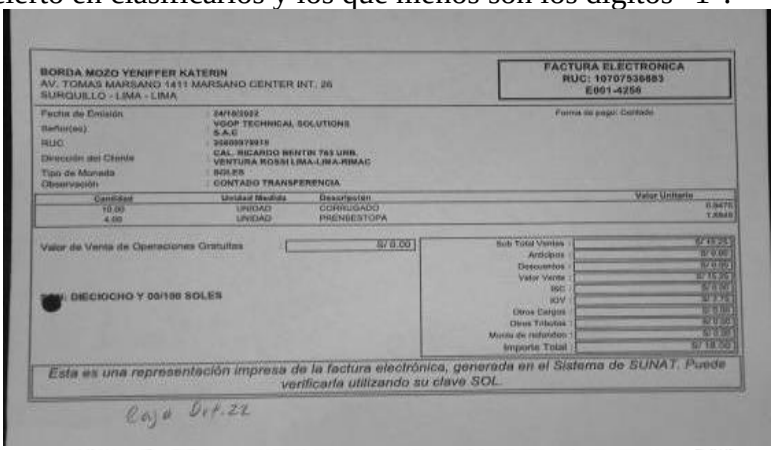


Figura 8. Imagen Original de la factura de la SUNNAT

- **Figura 9.** Imagen Procesada de la factura de la SUNNAT

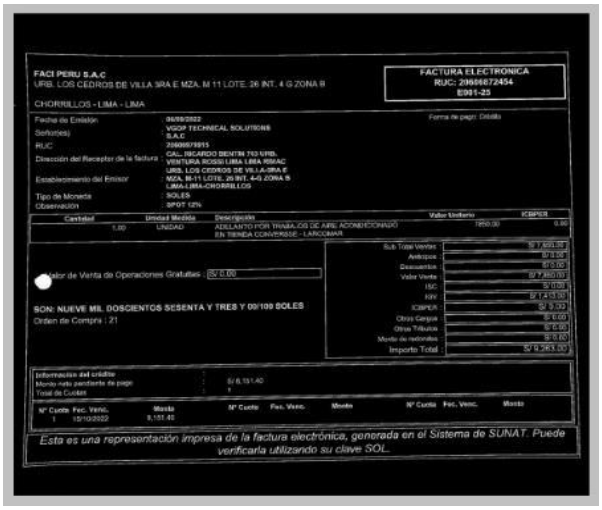


Figura 10. Complemento de la imagen de una factura de la SUNNAT

Figura 11. Segmentación de regiones de una factura de la SUNNAT

Al obtener los resultados del preprocesamiento y la clasificación de las imágenes se llegó tener en cuenta que el modelo es el mejor al realizar la predicción ya que se agregó un apartado para la evaluación de imágenes utilizando este modelo, en donde la salida nos muestra la imagen original, la imagen procesada, la imagen complementada, la imagen con segmentación de regiones y seguimientos de bordes con la red neuronal convolucional, como se puede visualizar en la **Fig.8 , Fig.9, Fig.10 y Fig.11.**

CONCLUSIONES

En el presente proyecto se lograron los objetivos al realizar el modelo de la red neuronal convolucional con el fin de realizar la clasificación de dígitos impresos del 0 al 9 se puede concluir que el modelo implementado es capaz de clasificar imágenes de dígitos impresos en cada una de sus clases de manera correcta.

El modelo de clasificación de dígitos impresos puede ser desarrollados para realizar un sistema automatizado, en este caso emitido para las facturas de la SUNAT teniendo como base a la propuesta del modelo se obtuvo una precisión de 76. %, el cual fue entrenado con imágenes de facturas de la SUNAT previamente procesadas, este preprocesamiento se realizó con una serie de técnicas de procesamiento de imágenes lo cual ayudó a aumentar la precisión del modelo propuesto.

Para futuras investigaciones, se recomienda explorar la aplicación del modelo de clasificación de dígitos impresos en otros tipos de documentos más allá de las facturas de la SUNAT. Esto podría incluir recibos, cheques, formularios gubernamentales u otros documentos financieros, lo que ampliaría la utilidad y aplicabilidad del sistema.
Sería valioso investigar la incorporación de técnicas de aprendizaje profundo más avanzadas para mejorar aún más la precisión del modelo. Esto podría incluir el uso de arquitecturas de redes neuronales más complejas o la implementación de técnicas de aumento de datos para enriquecer el conjunto de entrenamiento.

Se sugiere también explorar la posibilidad de desarrollar un sistema end-to-end que no solo clasifique los dígitos, sino que también extraiga y procese automáticamente toda la

información relevante de las facturas. Esto podría incluir la detección y extracción de campos como fechas, nombres de empresas y montos totales, lo que aumentaría significativamente la automatización del proceso de manejo de facturas.

REFERENCIAS BIBLIOGRÁFICAS

Bojer, C. S., & Meldgaard, J. P. (2021). *Kaggle forecasting competitions: An overlooked learning opportunity. International Journal of Forecasting*, 37(2), 587-603. https://doi.org/10.1016/j.ijforecast.2020.07.007

Neri, M. J., Villegas, O. O. V., Sánchez, V. G. C., De Jesús Ochoa Domínguez, H., Nandayapa, M., & Azuela, J. H. S. (2023). *A methodology for character recognition and revision of the linear equations solving procedure. Information Processing and Management*, *60*(1), 103088. https://doi.org/10.1016/j.ipm.2022.103088

O, M. A. K., & Poruran, S. (2020). *OCR-Nets: Variants of Pre-trained CNN for Urdu Handwritten Character Recognition via Transfer Learning. Procedia Computer Science*, *171*, 2294-2301. https://doi.org/10.1016/j.procs.2020.04.248

e Silva, L. C., de Carvalho César Sobrinho, Á. A., Cordeiro, T. D., Melo, R. F., Bittencourt, I. I., Marques, L. B., da Cunha Matos, D. D. M., da Silva, A. P., & Isotani, S. (2023). Applications of convolutional neural networks in education: A systematic literature review. *Expert Systems with Applications*, 120621. https://doi.org/10.1016/j.eswa.2023.120621

Kussul, E., & Baidyk, T. (2004). Improved method of handwritten digit recognition tested on MNIST database. *Image and Vision Computing*, *22*(12), 971-981. https://doi.org/10.1016/j.imavis.2004.03.008

Gan, J., Chen, Y., Hu, B., Leng, J., Wang, W., & Gao, X. (2023). *Characters as graphs: Interpretable handwritten Chinese character recognition via Pyramid Graph Transformer*. *Pattern Recognition*, *137*, 109317. https://doi.org/10.1016/j.patcog.2023.109317

Wang, P., Fan, E., & Wang, P. (2021). *Comparative analysis of image classification algorithms based on traditional machine learning and deep learning. Pattern Recognition Letters*, 141, 61-67. https://doi.org/10.1016/j.patrec.2020.07.042

Lee, H., Eun, Y., Hwang, J. Y., & Eun, L. Y. (2022). *Explainable deep learning algorithm for distinguishing incomplete Kawasaki disease by coronary artery lesions on echocardiographic imaging*. Computer Methods and Programs in Biomedicine. Obtenido de https://doi.org/10.1016/j.cmpb.2022.106970

Yusit, G. M. H. (2019). *La Facturación Electrónica y los factores que influyen en su aplicación*. http://hdl.handle.net/20.500.12590/16747

NumPy. (2023). https://numpy.org/

pandas - Python Data Analysis Library. (2023). https://pandas.pydata.org/

Espejo Pérez, C.(2017) S*istema de reconocimiento de caracteres numéricos para actualización de base de datos* https://repositorioacademico.upc.edu.pe/bitstream/handle/10757/622402/Espejo_PC.pdf?sequence=5. Jupyter, P. (2022). Obtenido de https://jupyter.org/

Tensorflow. (2022). tensorflow. Obtenido de tensorflow.org: https://www.tensorflow.org/overview/

Matplotlib — Visualization with Python. (2022). https://matplotlib.org/

Ahlawat, S., & Choudhary, A. (2020). *Hybrid CNN-SVM Classifier for Handwritten Digit Recognition. Procedia Computer Science*, *167*, 2554-2560. https://doi.org/10.1016/j.procs.2020.03.309

Duan, L., Ren, Y., & Duan, F. (2022*). Adaptive stochastic resonance based convolutional neural network for image classification. Chaos*, Solitons & Fractals, 162, 112429. https://doi.org/10.1016/j.chaos.2022.112429

SUNAT. (2023). *Información estadística | Comprobantes de Pago Electrónicos*. https://cpe.sunat.gob.pe/informacion-estadistica

Barrios Arce, J. I. (2022). *La matriz de confusión y sus métricas*. Obtenido de Health Big Data: https://www.juanbarrios.com/la-matriz-deconfusion-y-sus-metricas/

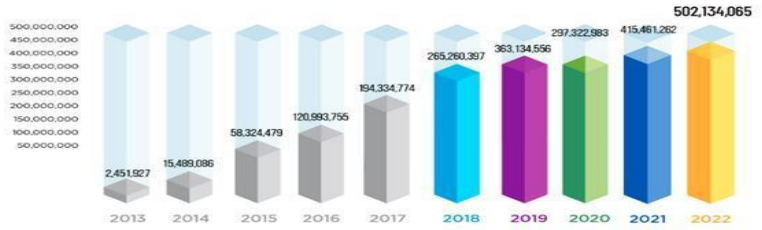
Rafiee, A. A., & Farhang, M. (2023). *A deep convolutional neural network for salt-and-pepper noise removal using selective convolutional blocks*. *Applied Soft Computing*, *145*, 110535. https://doi.org/10.1016/j.asoc.2023.110535

ANEXOS

Anexo A. Dígitos del TMNIST

Anexo B. Gráfica de la problemática en las facturas electrónicas emitidas acumulado del año 2014-2023

Anexo C. Gráfica de la problemática en las facturas electrónicas emitidas anualmente del año 2013-2022



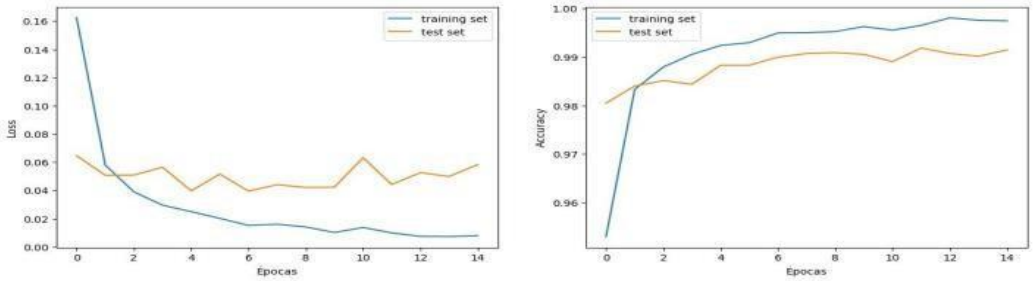
Fuente: SUNAT

Anexo D. Capas de la CNN

```
modelo = tf.keras.models.Sequential()
modelo.add(
    tf.keras.layers.Convolution2D(input_shape=(WIDTH, HEIGHT, CHANNELS),
                                  kernel_size=5,filters=8,strides=1,
                                  activation=tf.keras.activations.relu,
              kernel_initializer=tf.keras.initializers.VarianceScaling()))

modelo.add(
    tf.keras.layers.MaxPooling2D(pool_size=(2,2),strides=(2,2)))

modelo.add(
    tf.keras.layers.Convolution2D(kernel_size=5,filters=16,
                                  strides=1,
                                  activation=tf.keras.activations.relu,
              kernel_initializer=tf.keras.initializers.VarianceScaling()))

modelo.add(tf.keras.layers.Flatten())

modelo.add(tf.keras.layers.Dense(units=128,
                                 activation=tf.keras.activations.relu))

modelo.add(tf.keras.layers.Dropout(0.2))

modelo.add(tf.keras.layers.Dense(units=10,
                                 activation=tf.keras.activations.softmax,
                 kernel_initializer=tf.keras.initializers.VarianceScaling()))

#Visualizar el modelo
modelo.summary()
```

Anexo E. Evaluación del modelo con los datos del entrenamiento de la evolución de la presión y valor de pérdida

Anexo. F



Evaluación del modelo con los datos del entrenamiento.