Learning Hiragana with Machine Learning

Eunice Mata, Ricardo Chuy, Andre Jo
Departamento de Computación, Deep Learning y Sistemas Inteligentes, Universida del Valle de Guatemala
mat21231@uvg.edu.gt
chu221007@uvg.edu.gt
jo22199@uvg.edu.gt
https://chuy-zip.github.io/PORTAFOLIO/#PAPER

Introducción

El sistema de escritura japonés se compone de tres conjuntos principales: kanji, hiragana y katakana. El hiragana, conformado por 46 caracteres básicos, constituye el primer nivel de aprendizaje para hablantes no nativos y es ampliamente utilizado en textos educativos y material introductorio. Sin embargo, la similitud visual entre varios de sus caracteres dificulta su correcta identificación, especialmente cuando son escritos a mano, lo que representa un desafío tanto para estudiantes como para sistemas automáticos de reconocimiento.

Este trabajo aborda el problema de la clasificación automática de caracteres de hiragana escritos a mano mediante técnicas de aprendizaje profundo. Se evalúan dos enfoques principales: un modelo convolucional entrenado desde cero y un modelo basado en Transfer Learning, utilizando MobileNetV2 preentrenado en ImageNet como extractor de características. El objetivo es comparar su desempeño, analizar sus capacidades de generalización y determinar la efectividad del uso de modelos preentrenados en tareas específicas de reconocimiento de escritura japonesa.

Adicionalmente, se incorpora el uso de técnicas de Explainable Artificial Intelligence (XAI), particularmente LIME, con el propósito de interpretar las decisiones del modelo y observar qué regiones de las imágenes influyen en la clasificación. Este análisis permite identificar patrones visuales relevantes y comprender los casos en los que los modelos tienden a confundirse entre caracteres similares.

Los modelos desarrollados se integran en una interfaz interactiva implementada en Streamlit, que permite al usuario dibujar un carácter y obtener una predicción en tiempo real. Este entorno facilita la evaluación práctica de los modelos y constituye un recurso potencial para aplicaciones educativas y sistemas de asistencia al aprendizaje del alfabeto japonés. A continuación se presenta el contexto general del problema, el cual describe la estructura del sistema de escritura japonés y la importancia del alfabeto hiragana dentro del proceso de aprendizaje del idioma.

Contexto general del problema

El idioma japonés cuenta con 3 alfabetos distintos: hiragana, katakana y kanji. Estos se complementan de distintas formas para formar el sistema de escritura de dicho país. En el caso del kanji, son cerca de unos 2136 símbolos que representan distintas cosas palabras, frases cortas, entre otros. Estos provienen del lenguaje chino y en total son aproximadamente 50k. Por otra parte está el hiragana y el katakana. Estos son los alfabetos base, cada uno con un total de 46 símbolos. Dichos símbolos no representan letras o sonidos individuales, como el español o inglés. Más bien, son vocales o sílabas. Lo interesante es que ambos alfabetos tienen la misma cantidad, porque ambos representan los mismos sonidos pero con distintos símbolos.

Por ejemplo el sonido “ka”, en Hiragana se escribe como “か” y en Katakana se escribe como “カ”. Distintos símbolos pero mismo sonido, en la actualidad el hiragana se usa generalmente para palabras nativas de japón y el katakana para palabras externas, generalmente del idioma inglés. Estos 3 sistemas se juntan para poder definir el sistema

de escritura japonés. Si una persona desea aprender el idioma como tal, usualmente el primer alfabeto que se aprende es el de Hiragana y justo por eso es el alfabeto que hemos elegido para este proyecto. A continuación se muestra dicho alfabeto.

En la actualidad, existen muchas herramientas para aprender el idioma y la comunidad es bastante activa. Existen aplicaciones, juegos y demás para practicar el alfabeto, palabras, gramática, entre otros. Entonces, con este proyecto buscamos realizar una herramienta para practicar los símbolos del alfabeto japonés en Hiragana, donde el mismo usuario en lugar de solo elegir y practicar de forma memorística dibuje manualmente un carácter del alfabeto y se valore con una puntuación dicho dibujo. Para realizar esto, proponemos el uso de un modelo entrenado con machine learning para reconocer imágenes.

Metodología

El problema es particularmente interesante debido a la naturaleza de los símbolos. Muchos de estos son realmente parecidos o bien algunos símbolos forman parte de otros aunque no tengan nada que ver fonéticamente. Por ejemplo el sonido “ko” (こ) está dentro del símbolo “ta” (た) y también en el símbolo “ni” (に), entonces nos pareció interesante saber si es posible realizar un modelo con alta precisión para predecir letras, basado en imágenes. Con esto podríamos analizar cómo es que un modelo de inteligencia artificial clasifica caracteres similares, incluso cuando hay porciones idénticas dentro de las imágenes. Para resolver este problema hay varios enfoques, pero nos enfocamos en el uso de Redes Neuronales Convolucionales (CNN) por sus siglas en inglés como metodología principal para entrenamiento de modelos. Para aumentar las posibilidades de éxito del entrenamiento del modelo y generar comparaciones, utilizamos 2 enfoques distintos para el entrenamiento. El primero fue realizar un modelo desde cero, configurando las capas de la red, filtros y técnicas de entrenamiento elegidas manualmente. Mientras que el segundo enfoque, lo que buscamos fue aplicar el concepto de Transfer Learning. Utilizando un modelo pre entrenado para una tarea de clasificación similar, lo que buscamos es reutilizar un modelo y adaptarlo con datos nuevos, imágenes del alfabeto japonés.

Luego de tener cada modelo entrenado, haremos un proceso de explainable AI utilizando LIME, Local interpretable model-agnostic explanations. Esta es una técnica que ayuda a comprender las decisiones de modelos de caja negra en machine learning, como lo es el caso de los modelos que se entrenaron. LIME se le dice un método local debido a que no toma todos los datos y predicciones para ayudar a explicar la toma de decisiones del modelo, sino que se intenta explicar una predicción en específico en un dato de ejemplo. En el contexto de redes convolucionales lo que hace LIME es tomar una imagen ya con el preprocesamiento adecuado y la modifica varias veces, con el fin de detectar cuales son las áreas que al ser afectados modifican más la predicción. Al final del algoritmo hecho por la librería, se obtiene una imagen donde dichas regiones clave se muestran para saber qué partes de la imagen son las más importantes para la toma de decisión de una imagen en específico. Haciendo varias de estas pruebas, es que buscamos inferir y comprender acerca de la toma de decisiones en general de los modelos, analizando casos locales con LIME. Según Gupta (2022), las técnicas de interpretación de modelos como LIME son esenciales para comprender las predicciones de las CNN, especialmente en aplicaciones de clasificación de caracteres.

Finalmente, teniendo ambos modelos guardados, nos proponemos usarlos dentro de una interfaz gráfica para implementar la herramienta de práctica de hiragana. Usando una región para dibujar en una aplicación de streamlit, dicho input es procesado para poder

ser interpretado por un modelo y se le da una nota al usuario en base a las predicciones de la inteligencia artificial. Si el dibujo del usuario al ser evaluado, está dentro de las primeras 10 predicciones del modelo, entonces se le asigna un puntaje en función de la posición en la que está en la predicción. Por ejemplo, si se pide dibujar el símbolo “ko” (こ), y al dibujar ko está en la primera posición de las predicciones (la clase con mayor porcentaje) la nota es de 10/10. La nota máxima será de 10 y la mínima de 0 en el caso que el símbolo solicitado no se encuentre dentro las primeras 10 predicciones.

Datos utilizados

En ambos modelos utilizamos como fuente de datos el mismo set de imágenes de kaggle. El set de datos tiene el nombre de “Handwritten Japanese Hiragana Characters”. Este conjunto de datos contiene imágenes de los 46 caracteres de hiragana con varios ejemplares escritos a mano. Estos datos justamente fueron recolectados para tareas de reconocimiento de imágenes/caracteres mediante técnicas de aprendizaje profundo. Las características principales del set de datos son:

- **Fuente:** Kaggle (farukece/handwritten-japanese-hiragana-characters)
- **Número de clases:** 46 caracteres hiragana básicos
- **Imágenes por clase:** 100 imágenes
- **Total de imágenes:** 4,600 imágenes
- **Formato de las imágenes:** JPG
- **Dimensiones originales:** 500 × 400 píxeles
- **Tipo de imagen:** RGB, aunque visualmente en escala de grises
- **Características visuales:** Fondo blanco con trazos negros

La distribución de las clases presenta un balance perfecto para todos los caracteres de japonés. Todas las clases tienen exactamente 100 imágenes de la misma resolución. Los 46 caracteres incluidos son:

- aa (あ), chi (ち), ee (え), fu (ふ), ha (は), he (へ), hi (ひ), ho (ほ), ii (い)
- ka (か), ke (け), ki (き), ko (こ), ku (く)
- ma (ま), me (め), mi (み), mo (も), mu (む)
- na (な), ne (ね), ni (に), nn (ん), no (の), nu (ぬ)
- oo (お), ra (ら), re (れ), ri (リ), ro (ろ), ru (る)
- sa (さ), se (せ), shi (し), so (そ), su (す)
- ta (た), te (て), tsu (つ), to (と)
- uu (う), wa (わ), wo (を)
- ya (や), yo (よ), yu (ゆ)

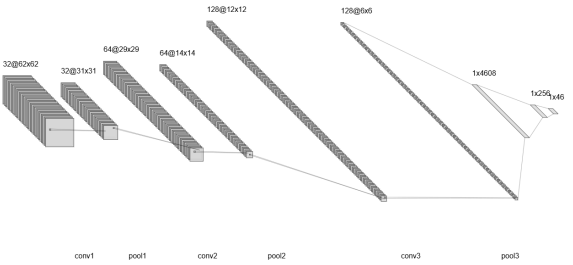
Para estos modelos se realizó una división estratificada del dataset en 3 partes distintas, manteniendo el balance de las clases. Se dividió en:

- Train 70%. 3220 Imágenes. Con un total de 70 imágenes por clase
- Validation 15%. 690 Imágenes. Con un total de 15 por clase
- Test: 15%- 690 Imágenes. Con un total de 15 por clase

Además de esto se prepararon los datos para optimizar el entrenamiento y adaptarlo para hacer una CNN. Transformamos las imágenes a un solo canal de color, aunque estuvieran en RGB, realmente al final solo estaban en blanco y negro. Además de eso las imágenes fueron escaladas de 500x400 a 64x64 utilizando un método de interpolación llamado LANCZOS para redimensionar sin perder calidad. La reducción de tamaño ayuda con la eficiencia del entrenamiento al final. Finalmente también se hace uso de una normalización para que los pixeles se normalicen del rango [0,255 al rango [0,1]. No se utilizaron técnicas de data augmentation para agregar más imágenes o modificaciones a las mismas

Modelo y Arquitectura

Modelo entrenado desde cero



Como mencionamos ya previamente, se trabajó con una Red Neuronal Convolucional

(CNN) para la clasificación de carácteres de hiragana. Las CNN son efectivas para reconocer patrones visuales, gracias a las operaciones que se realizan con las imágenes. Este algoritmo es capaz de extraer características de las imágenes mediante las capas de convolución. La arquitectura propuesta para este modelo en específico tiene un diseño secuencial con 3 bloques de capas convolucionales seguidos (con capas de max pooling intercaladas), seguidos de capas totalmente conectadas. Se realizó el modelo mediante TensorFlow y Keras. El modelo cuenta con:

1. Capa de entrada
2. Tres bloques convolucionales (Conv2D + MaxPooling2D)
3. Capa de aplanamiento (Flatten)
4. Capas densas (Dense + Dropout)
5. Capa de salida (Dense con activación Softmax)

Capa de entrada:

- Dimensiones: (64, 64, 1)
- Representa imágenes en escala de grises de 64×64 píxeles

Bloque Convolucional 1:

- Conv2D (conv1):
  - Filtros: 32
  - Tamaño del kernel: 3×3
  - Activación: ReLU
  - Salida: (62, 62, 32)
  - Parámetros: 320
- MaxPooling2D (pool1):
  - Tamaño del pool: 2×2
  - Stride: 2
  - Salida: (31, 31, 32)
  - Parámetros: 0

Esta parte sirve para detectar características básicas como bordes, líneas y contornos simples en la imagen.

Bloque Convolucional 2:

- Conv2D (conv2):
  - Filtros: 64
  - Tamaño del kernel: 3×3
  - Activación: ReLU
  - Salida: (29, 29, 64)
  - Parámetros: 18,496
- MaxPooling2D (pool2):
  - Tamaño del pool: 2×2
  - Stride: 2
  - Salida: (14, 14, 64)
  - Parámetros: 0

Este sirve en teoría para detectar características de nivel medio como curvas, formas y combinaciones de bordes.

Bloque Convolucional 3:

- Conv2D (conv3):
  - - Filtros: 128
  - - Tamaño del kernel: 3×3
  - - Activación: ReLU
  - - Salida: (12, 12, 128)
  - - Parámetros: 73,856
- MaxPooling2D (pool3):
  - - Tamaño del pool: 2×2
  - - Stride: 2
  - - Salida: (6, 6, 128)
  - - Parámetros: 0

Tiene como propósito detectar características de alto nivel específicas de los caracteres hiragana, como trazos característicos y patrones complejos.

Capa de Flatten:

- Entrada: (6, 6, 128)
- Salida: (4608,)
- Parámetros: 0

Esta capa convierte la matriz tridimensional en un vector para conectar a las siguientes capas.

Capas Fully Connected:

- Dense (dense1):
  - Neuronas: 256
  - Activación: ReLU
  - Salida: (256,)
  - Parámetros: 1,179,904
- Dropout (dropout1):
  - Tasa: 0.5 (50%)
  - Salida: (256,)
  - Parámetros: 0

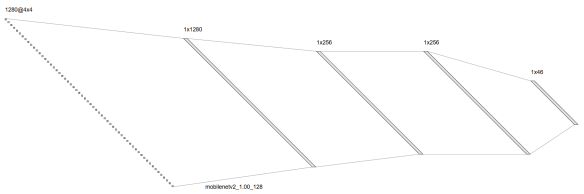
Aprender combinaciones complejas de las características extraídas. El Dropout previene el overfitting desactivando aleatoriamente el 50% de las neuronas durante el entrenamiento.

Capa de Salida:

- Dense (output):
  - Neuronas: 46
  - Activación: Softmax
  - Salida: (46,)
  - Parámetros: 11,822

En esta capa es cuando se genera probabilidad para cada una de las 46 clases de hiragana. Utilizando las función softmax es que se garantiza que la suma entre todas las probabilidades de salida sumen 1.

Modelo modificado (Transfer Learning)



Para el modelo de transfer learning se utilizó como extractor de características para la clasificación de caracteres en hiragana. En lugar de entrenar una red neuronal convolucional desde cero, se aprovechó un modelo pre entrenado en ImageNet, capaz de reconocer patrones visuales complejos gracias a sus capas de convolución y bloques residuales invertidos. MobileNetV2 funciona como una base congelada que transforma cada imagen en un mapa de características comprimido pero altamente descriptivo, evitando así el uso de capas tradicionales como MaxPooling. A continuación se presentan la estructura de las siguientes capas:

MobileNetV2:
Dimensiones:(4,4,1280)
Parámetros totales: 2,257,984
Parámetros entrenables: 0
Parámetros no entrenables: 2,257,984
Salida: 4,4,1280

Representa: Es una representación comprimida pero altamente descriptiva de la imagen, donde cada celda 4×4 contiene 1280 características aprendidas del modelo.

GlobalAveragePooling2D:
Entrada: (4,4,1280)
Parámetros: 0
Salida: 1280

Representa: Reducción de las dimensiones espaciales 4x4 de MobileNetV2, donde solo se mantiene la información esencial.

Capas Fully Connected:

Dense (dense1):
Neuronas: 256
Activación: ReLU
Entrada: 1280
Parámetros 327,936
Salida: (256)

Dropout:
tasa: 0.5
Parámetros: 0
salida: (256)

El dropout es utilizado para prevenir el overfitting desactivando aleatoriamente el 50% de las neuronas durante el entrenamiento.

Capa de salida:

Dense(output):
Neuronas:46
Activación: Softmax
Entrada: 256
Parámetros: 11,822
Salida: (46)

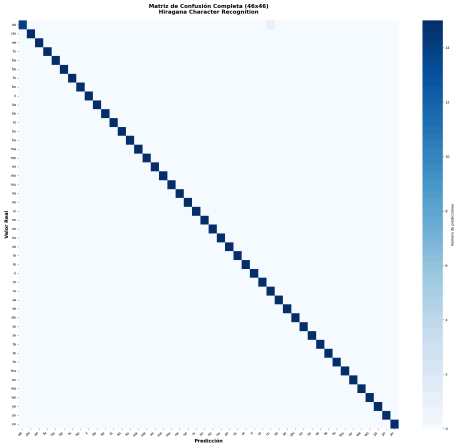
Esta capa es utilizada para generar la probabilidad para cada una de las 46 clases de hiragana. Por lo tanto su suma de todas las probabilidades debe de ser 1.

En total, el modelo resultante utiliza 2,597,742 parámetros para realizar las predicciones.

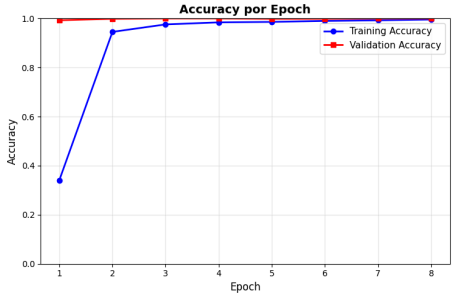
Resultados

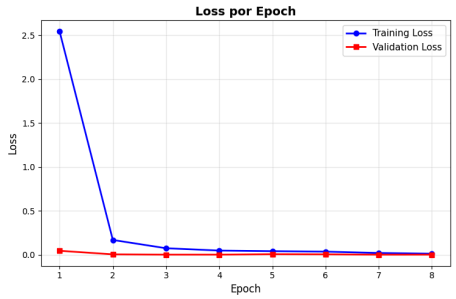
Modelo entrenado desde cero

Luego de realizar el entrenamiento del modelo con todos los datos descritos previamente, el modelo tuvo un desempeño realmente impresionante. Como resultados finales del entrenamiento hemos obtenido un Accuracy de 99.57% en el training, un 100% en validation y un 0.9986 en el test. Estas métricas son sorprendentes y al obtener tales resultados graficando la matriz de confusión queda de la siguiente forma:

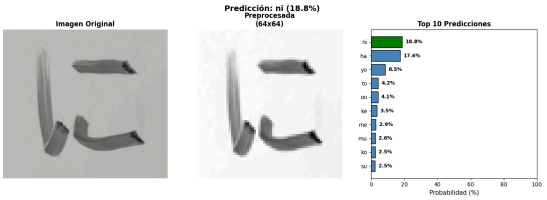


La matriz está representada con un mapa de calor, ya que al ser tantas clases es difícil determinar y distinguir los números dentro de la matriz. El eje vertical es el valor real y el eje horizontal es el valor predicho. Se puede ver claramente una línea recta en la diagonal, producto de la alta precisión del modelo. Solo se confundió 1 vez, prediciendo “a” como “ru”. Evaluando las curvas de aprendizaje se puede ver también lo rápido que el modelo logró aprender los patrones.





La diferencia mínima entre validation y training indican un buen desempeño y generalización. Y por último, los resultados de hacer explainable AI con el modelo entrenado fueron los siguientes. Para comprobar el funcionamiento del modelo también se puso a prueba con otras imágenes fuera del set de datos, por la siguiente imagen fue hecho a mano en un pizarrón por nosotros:



Finalmente para el modelo también hicimos el análisis con LIME de explainable AI. La idea de realizar varias pruebas fue identificar qué factores ayudan en la toma de decisión local para comprender mejor la toma de decisión en general del modelo CNN. En estas imágenes principalmente comparamos símbolos similares y de imágenes no necesariamente presentes en el set de datos. Las distintas partes de la imagen describen las regiones críticas y cuales son las que están apoyando o bien contradiciendo la predicción. En color rojo se ven las regiones las que apoyan y en tonos azules las que contradicen. Mientras más intenso el color, quiere decir que es más fuerte el apoyo o contradicción. A continuación se muestran distintas imagenes que fueron analizadas con LIME.

Imagen de prueba obtenida del set de datos:

- Etiqueta real: ni
- Predicción del modelo: ni
- Regiones: 5

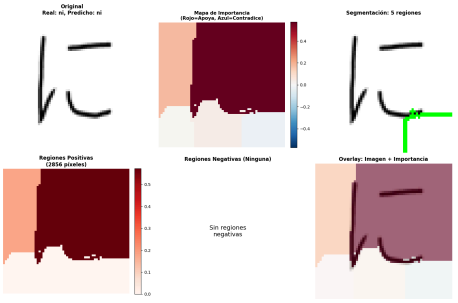


Imagen recortada y obtenida de internet:

- Etiqueta real: ni
- Predicción del modelo: ni
- Regiones: 7

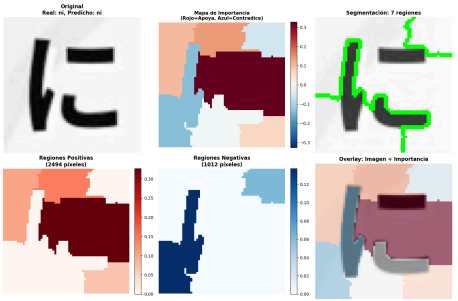
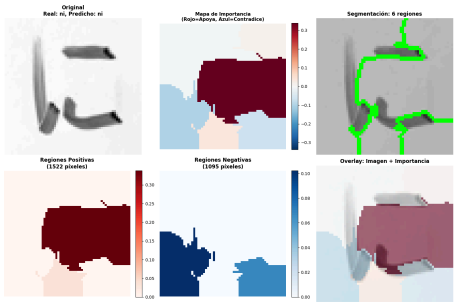


Imagen tomada de un trazo en un pizarrón hecho a mano por nosotros:

- Etiqueta real: ni
- Predicción del modelo: ni
- Regiones: 6



Podemos ver claramente que el modelo ha sido capaz de clasificar de forma correcta el símbolo “ni” sin importar que fuera una imagen del set de datos, internet o a mano. También, ya que el símbolo es un poco complejo y similar a otros, parecen haber regiones que contradicen y otras que apoyan.

Imagen obtenida de internet:

- Etiqueta real: ta
- Predicción del modelo: ta
- Regiones: 6

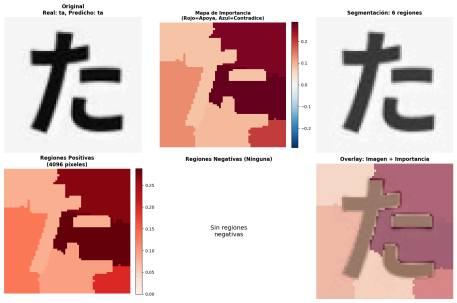


Imagen obtenida de internet:

- Etiqueta real: ko
- Predicción del modelo: ta
- Regiones: 4

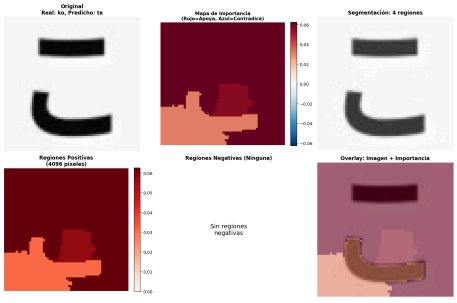


Imagen obtenida de internet:

- Etiqueta real: re
- Predicción del modelo: re
- Regiones: 6

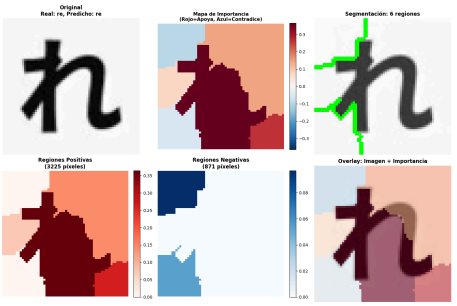


Imagen obtenida de internet:

- Etiqueta real: ne
- Predicción del modelo: ne
- Regiones: 7

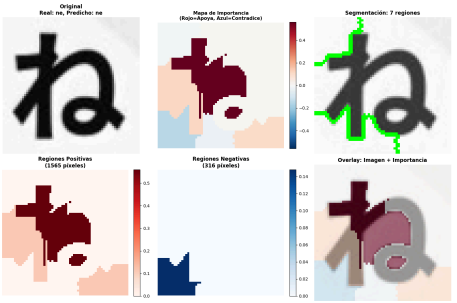
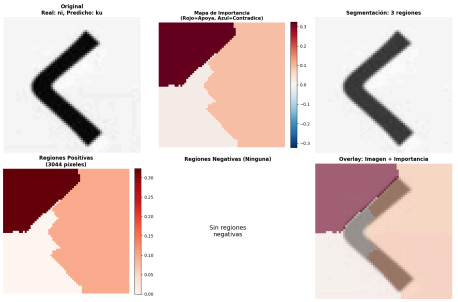


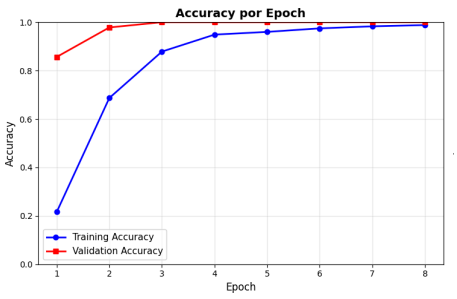
Imagen obtenido de internet:

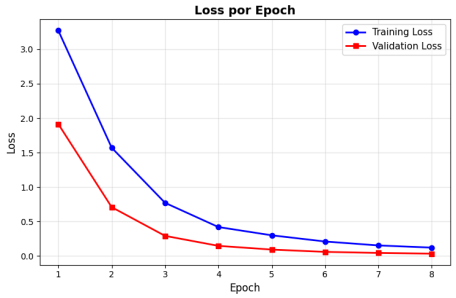
- Etiqueta real: ku
- Predicción del modelo: ku
- Regiones: 3



Modelo entrenado con transfer learning

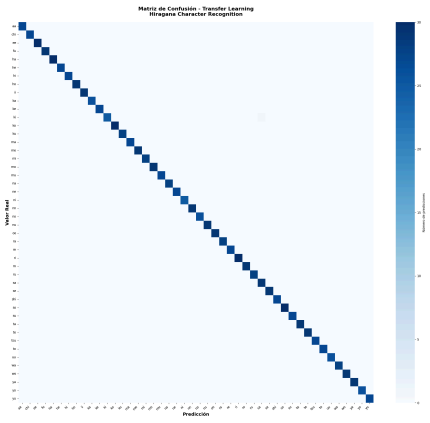
Luego de realizar..



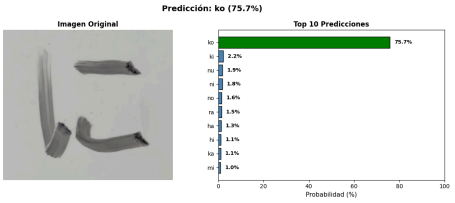


Se puede interpretar que el modelo aprendió con rapidez y generaliza adecuadamente al conjunto de validación. En las primeras épocas, la accuracy de entrenamiento pasa de 0.2 a casi 0.9, mientras que la de validación aumenta de 0.86 hasta prácticamente 1.0 y luego se mantiene estable. Al mismo tiempo, la loss de entrenamiento y validación desciende de forma continua y la curva de validación no presenta incrementos, por lo que no hay una señal clara de sobreajuste.

En comparación con la gráfica del modelo entrenado desde cero, aquí el incremento de accuracy es más gradual y consistente de época en época, lo que refleja un proceso de aprendizaje más estable gracias al uso de transfer learning.

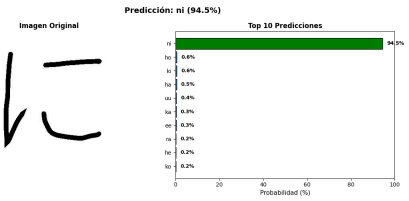


De acuerdo con la matriz de confusión, se observa que el modelo presenta un desempeño muy consistente en la mayoría de las clases, ya que la gran mayoría de las predicciones se concentran sobre la diagonal principal. Esto indica que, en casi todos los casos, el carácter real coincide con la clase predicha. Aunque existen algunas confusiones puntuales entre ciertas clases, la proporción de errores por categoría se mantiene por debajo como se observaba en la siguiente figura



Alfabeto Ko que predijo

Como se observa en la siguiente figura, el modelo de transfer learning no logró predecir correctamente el carácter. El modelo asignó una probabilidad del 75.7% a una clase que en realidad sólo corresponde al 1.9%, como se muestra en la imagen. Esto podría deberse a que la línea del lado izquierdo tiene muy poca opacidad, lo que hace que el modelo pierda esa característica y termine confundiendo el símbolo con Ko, ya que ambos se parecen visualmente.



A diferencia de la imagen anterior, en esta el carácter tiene mayor opacidad dentro de la escala de blanco y negro, por lo que el modelo de transfer learning sí logra predecir correctamente que el alfabeto corresponde a Ni. A continuación, se presentará el análisis con LIME dentro del enfoque de Explainable AI para identificar qué factores o características considera el modelo y entender en qué situaciones podría confundirse o no.

Nuevamente, se realizó el análisis a través de LIME de explainable AI. La idea de realizar varias pruebas fue identificar cómo en el caso anterior qué factores ayudan en la toma de decisión local para comprender cuál fue la mejor decisión en general del modelo de Transfer

Learning. De las imágenes tomadas se observan algunos símbolos similares y de imágenes no presentes del set de datos de prueba y entrenamiento.

En color rojo se ven las regiones las que apoyan y en tonos azules las que contradicen. Mientras más intenso el color, quiere decir que es más fuerte el apoyo o contradicción. A continuación se muestran distintas imágenes que fueron analizadas con LIME.

Imagen tomada de un trazo en un pizarrón hecho a mano por nosotros:

- Etiqueta real: ni
- Predicción del modelo: ko
- Regiones: 13

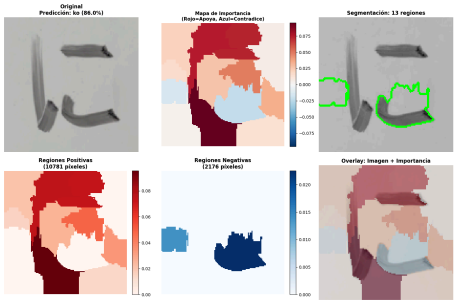


Imagen de prueba obtenida del set de datos:

- Etiqueta real: ni
- Predicción del modelo: ni
- Regiones: 15

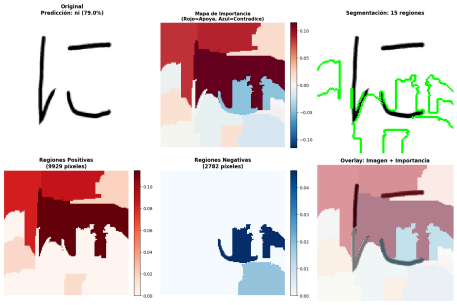


Imagen obtenida de internet:

- Etiqueta real: ni
- Predicción del modelo: ko
- Regiones: 13

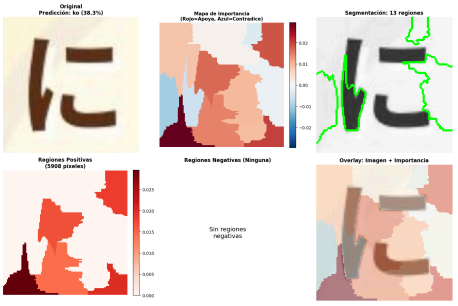


Imagen obtenida de internet:

- Etiqueta real: ko
- Predicción del modelo: ko
- Regiones: 13

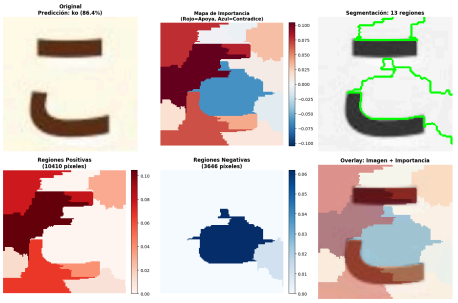


Imagen obtenida de internet:

- Etiqueta real: ku
- Predicción del modelo: shi
- Regiones: 5

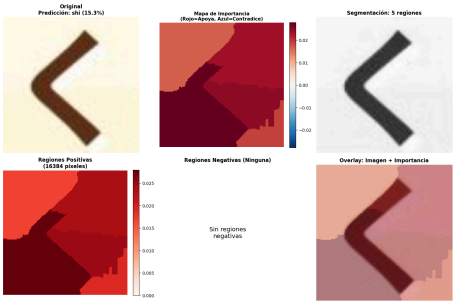


Imagen obtenida de internet:

- Etiqueta real: ta
- Predicción del modelo: chi
- Regiones: 10

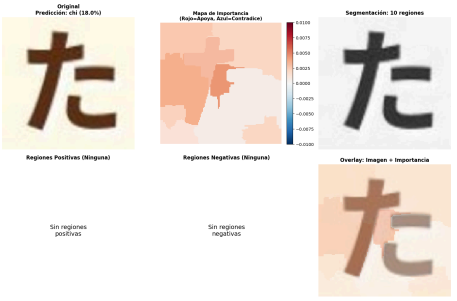


Imagen obtenida de internet:

- Etiqueta real: ne
- Predicción del modelo: ne
- Regiones: 14

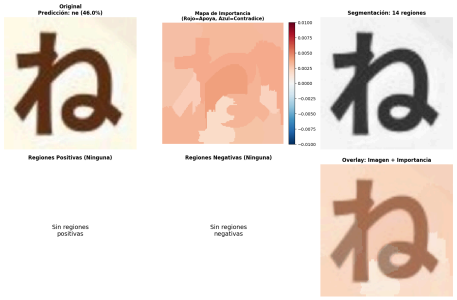


Imagen obtenida de internet:

- Etiqueta real: re
- Predicción del modelo: nn
- Regiones: 11

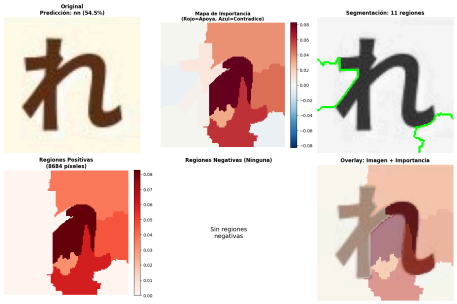
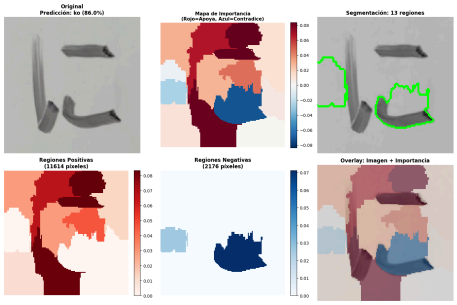


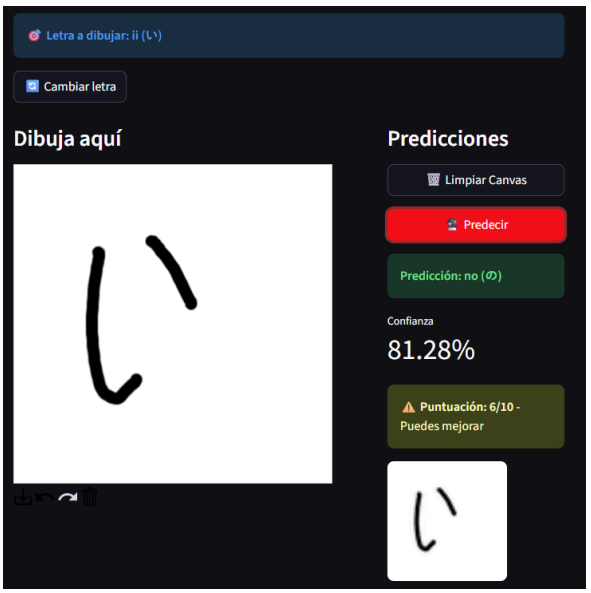
Imagen tomada de un trazo en un pizarrón hecho a mano por nosotros:

- Etiqueta real: ni
- Predicción del modelo: ko
- Regiones: 13



Aplicación de los modelos

La idea de este proyecto era crear un sistema que sea usado como una herramienta para practicar escritura de las letras japonesas en hiragana. El sistema selecciona una letra y el usuario debe de dibujar lo mejor que pueda esta letra.



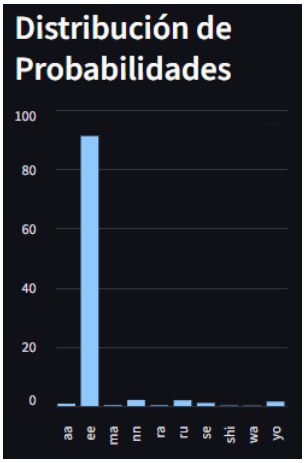
Para hacer uso de los modelos de forma User friendly, se hizo uso de la librería Streamlit y sus widgets para una integración estable. Este programa corre localmente y tiene las siguientes características:

CNN entrenado desde 0

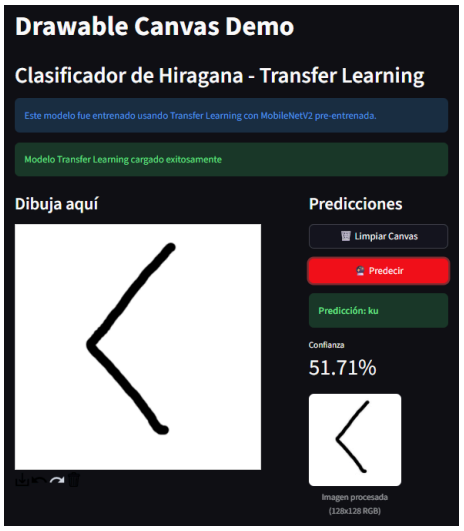
Aproximadamente como se ven las estadísticas con el modelo.

Tabla de top 10 predicciones

Gráfico de probabilidad comparado a otras



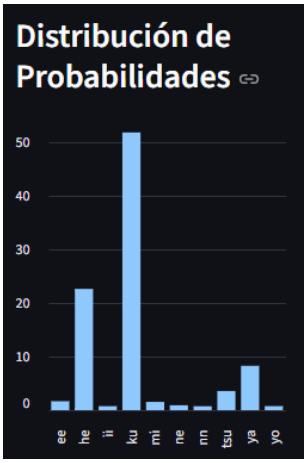
Transfer learning



Aproximadamente como se ven las estadísticas con el modelo.

Tabla de top 10 predicciones

Gráfico de probabilidad comparado a otras



Discusión

Al final los resultados en términos de entrenamiento fueron altamente satisfactorios. En el caso de la CNN que entrenamos llegando a tener un accuracy perfecto en el set de validación y teniendo una precisión de 99% en el caso del training y del test. En primera instancia al ver los resultados del modelo fue un poco preocupante que haya tenido una precisión tan alta, ya que conseguir un modelo perfecto por lo general es bastante difícil y en el caso de métricas absurdamente altas puede haber overfitting. Pero analizando el hecho que la diferencia entre validation y test es mínima (además de la matriz de confusión) en teoría el modelo está generalizando bien al momento de predecir. Es importante mencionar que aunque se hayan dado muy buenas métricas, realmente se tienen pocos datos de cada clase. Analizando las distribuciones de los datos para training, validation y testing, realmente hay pocos datos para cada clase, siendo 15 imágenes para

testing por clase al final. El hecho que el modelo tenga tan buena precisión realmente puede ser porque tiene pocas oportunidades para cometer errores y las imágenes puede que no varíen tanto unas de otras.

Entonces, para corroborar el desempeño del modelo, decidimos experimentar con otras imágenes de internet y también un ejemplar hecho a mano, antes de probar el modelo en la aplicación para dibujar. Probando una imagen con el símbolo “ni” (に), el modelo 1 de CNN fue capaz de identificar el símbolo dibujado a mano, pero profundizar más luego probamos con varios símbolos que poseen características prácticamente idénticas ta, ko, ni, re y ne (た、こ、に y ね、れ) por ejemplo, son algunos de los símbolos con mayores similitudes dentro del alfabeto. Probamos dichos símbolos a la vez con LIME, para poder comprender cómo es que se está realizando la toma de decisión por parte del modelo y evaluar al mismo tiempo la capacidad de predicción con imágenes fuera del set de datos. Los resultados fueron bastante satisfactorios nuevamente, de las 9 imágenes completamente nuevas, la predicción fue correcta para 8 de estas. Lo cuál en cierta forma es de esperar considerando con cuál fue la que cometió el error. De las 3 imágenes que usamos para probar に, la imagen fue clasificada exitosamente en todos los casos. Sin importar que fuera de internet, del set de datos o bien a mano.

Observando las secciones de la imagen podemos ver que la imagen de internet fue la que tuvo mayor cantidad variedad de secciones y con esto comenzamos e identificar cómo es que el modelo toma sus decisiones. Como tal el modelo está captando relaciones complejas y no solo trazos. Por lo que se ve en las secciones vistas por lime, analiza el espacio entre los caracteres también. Curiosamente, en el caso de Ni (に), la parte más importante parece ser la parte derecha, especialmente en el espacio entre la línea inferior y superior del lado derecho. Del lado izquierdo se puede ver también que hay una parte del símbolo que no apoya la decisión, esto tiene sentido ya que otros símbolos como ke (け) tiene exactamente el mismo trazo. El modelo no evalúa la imagen completamente, sino que toma en cuenta los trazos y las cosas que rodean a dichos trazos. En el caso del símbolo ta (た), la predicción fue exitosa nuevamente y realmente no hay regiones que contradigan la decisión. Pero en el caso de ko (こ), la predicción fue incorrecta, se predijo que era ta (た) el modelo. Analizando esto hace bastante sentido que el modelo se haya confundido, tomando en cuenta que el símbolo está contenido dentro de “ta”. De hecho el análisis de LIME es prácticamente idéntico al de “ta”, en una región en específico. Seguramente “ko” también era parte de las posibles predicciones, pero esto nos indica que en efecto el modelo no es perfecto y con símbolos similares puede cometer errores.

Luego también se tiene la comparación de de ne (ね) y re (れ) símbolos prácticamente idénticos que difieren prácticamente en el final del trazo. A pesar de la similitud, el modelo clasificó ambos de forma correcta. En ambos casos el lado izquierdo es de color azul, lo cuál confirma la teoría de decisión, el modelo detecta trazos que son similares a los de otros símbolos y toma en cuenta distintas regionse de la imagen para tomar decisiones. Podemos ver que un región clave en ambos casos es justo el centro de la imagen, dependiendo del símbolo hay un trazo o no allí. Por lo que el modelo puede diferenciarlos. Por último quisimos incluir también un símbolo fácil de clasificar, ku (く). Podemos decir que es sencillo para el model, principalmente porque es único y no hay otros que tengan un trazo similar. De hecho, se puede ver que al analizarlo con LIME, la figura tiene menos regiones. Las letras más simples, tienen menos regiones ya que es más fácil identificar los trazos.

DISCUSIÓN SOBRE EL TRANSFER LEARNING

Los resultados obtenidos con el modelo de transfer learning mostraron algunos casos de confusión. Durante el entrenamiento, la época con la mejor precisión alcanzó 99.71% en training y 99.92% en precisión. Este comportamiento puede ser preocupante, ya que valores tan altos en ambas curvas podrían indicar indicios de overfitting. Sin embargo, al observar las gráficas por épocas, tanto la precisión como la pérdida en validación se

mantienen estables y cercanas a valores óptimos. Además, la matriz de confusión muestra una fuerte correlación entre las predicciones y sus clases reales, lo cual sugiere que el modelo sí está generalizando adecuadamente.

Por otro lado, el dataset del alfabeto sólo contenía 100 imágenes por clase. Luego de dividir en 70% para entrenamiento, 15% para validación y 15% para pruebas, cada partición queda 15 imágenes por clase en validación y pruebas. Esto limita la capacidad del modelo para aprender variabilidad real. Además, las imágenes no incluían diferentes escalas de tonos grises o fondos que puedan afectar al modelo, lo cual puede afectar la capacidad del modelo para distinguir correctamente ciertos caracteres.

Para evaluar la capacidad de generalización fuera del dataset original, se decidió probar imágenes externas del alfabeto hiragana, incluyendo imágenes obtenidas en internet e incluso dibujos hechos a mano por el equipo. A través de las pruebas y utilizando Explainable AI, se observó que símbolos como ni (に) y ko (こ) tienden a confundirse cuando están hechos a mano. Esto podría deberse a que, al dibujar, el trazo puede no tener un contraste adecuado, haciendo que partes del símbolo se mezclen con el color del fondo. En LIME se observa que el modelo, en ocasiones, toma el fondo como característica relevante.

También se identificaron dificultades para distinguir entre símbolos como ku (く) y he (へ) en imágenes provenientes de distintas fuentes. Su forma es muy similar, y en los mapas de probabilidad se observa que el modelo asigna varias clases con probabilidades cercanas a 15.3%, escogiendo finalmente la clase con mayor probabilidad aunque no sea claramente dominante. Situaciones similares ocurren con chi (ち) y ta (た).

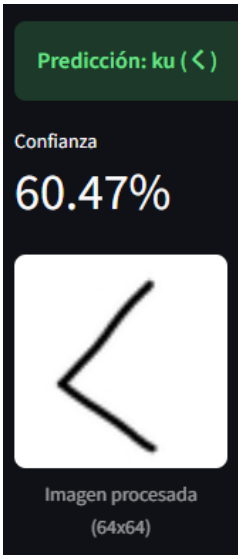
Otro caso interesante es la confusión entre nn (ん) y re (れ). En LIME se aprecia que el modelo asigna mucha importancia visual al trazo final de cada símbolo, el cual es muy parecido en ambas clases. Esto contribuye directamente a la confusión observada.

El modelo puede equivocarse al reconocer algunos símbolos cuando estos presentan características muy similares entre sí, como formas, tamaños o estilos gráficos parecidos. Estas similitudes pueden hacer que el modelo confunda un símbolo con otro, especialmente si las diferencias son sutiles o si el símbolo aparece con variaciones visuales. Sin embargo, mientras más claras y distintas sean las características del símbolo, mayor será la precisión del modelo al identificarlo correctamente ya que así fueron entrenadas los modelos.

DISCUSIÓN SOBRE LA APLICACIÓN DE LOS MODELOS

Durante la integración del modelo CNN y del modelo basado en transfer learning dentro de la interfaz de usuario, se observaron una serie de comportamientos interesantes que permiten evaluar no solo el rendimiento cuantitativo de los modelos, sino también su utilidad práctica en un entorno real. Uno de los hallazgos más relevantes es que ambos modelos resultan ser especialmente estrictos en sus condiciones de entrada al momento de realizar una predicción. En otras palabras, su desempeño se ve afectado de manera notable cuando el usuario dibuja una letra que no se ajusta a las características presentes en los datos utilizados durante el entrenamiento.

En la práctica, esto se manifiesta en que si la letra dibujada está descentrada, demasiado grande, demasiado pequeña, deformada o presenta trazos que podrían asemejarse parcialmente a otra sílaba, la capacidad del modelo para identificarla correctamente disminuye. Incluso cuando los modelos aciertan, la confianza del resultado puede ser sorprendentemente baja. Un ejemplo de este caso ocurrió al predecir la sílaba “く” (ku), donde el modelo entregó una predicción correcta pero con una confianza de apenas 60.47%, a pesar de tratarse de un carácter relativamente simple en su forma.



Este fenómeno puede explicarse por la sensibilidad inherente de los modelos a las condiciones del entrenamiento: si el conjunto de datos estaba compuesto por imágenes bien centradas, normalizadas y consistentes, los modelos tienden a “esperar” entradas similares. Por lo tanto, cualquier variación significativa en las proporciones, el trazado o la posición afecta su capacidad de generalizar, lo que se traduce en una baja probabilidad asignada a la categoría correcta.

Sin embargo, lejos de ser únicamente una limitación, este comportamiento estricto puede convertirse en una ventaja pedagógica dentro de la interfaz. La sensibilidad de los modelos funciona, en la práctica, como un sistema de retroalimentación inmediata para el usuario: obliga a la persona que está aprendiendo a escribir hiragana a dibujar los caracteres de manera más precisa, clara y consistente. De este modo, la herramienta no solo clasifica, sino que también fomenta la correcta caligrafía japonesa, ayudando al usuario a identificar errores de trazo o proporción.

En resumen, la manera en que los modelos reaccionan ante variaciones en la escritura evidencia su dependencia en la calidad del dato y revela oportunidades para futuras mejoras, como implementar algoritmos de preprocesamiento más robustos (centrado automático, normalización del tamaño, eliminación de ruido) o ampliar el conjunto de entrenamiento con ejemplos más diversos. A la vez, esta misma característica permite que la interfaz tenga un valor adicional en el aprendizaje, convirtiendo la sensibilidad del modelo en una forma eficiente de guía para quienes comienzan a practicar hiragana.

Recomendaciones y mejoras

- Aumentar datos al implementar transformaciones aleatorias durante el entrenamiento como rotaciones, zoom, desplazamientos leves y distorsiones elásticas que simulan variaciones naturales en la escritura. Estas técnicas pueden mejorar significativamente la robustez y generalización del modelo sin necesidad de recolectar más datos reales, lo cual es especialmente valioso dado el tamaño moderado del dataset actual.
- Otra solución a la falta de precisión en entornos reales es directamente conseguir más imágenes por clase. pasar de 100 a 200-500 imágenes por ejemplo. De esta manera hay más imágenes tanto para entrenamiento como para test.
- Complementar la métrica de accuracy con Precision, Recall y F1-Score por clase para identificar caracteres problemáticos. En este caso no implementamos estas métricas en las mediciones debido al alto desempeño, pero también son métricas importantes para la medición de rendimiento de modelos de clasificación.
- En general, aunque LIME sea una herramienta útil, el estudio se podría ver beneficiado por la implementación de más técnicas de análisis de modelos de caja negra. Técnicas como Grad-CAM (Gradient-weighted Class Activation Mapping) e Integrated Gradients para obtener múltiples perspectivas sobre qué regiones de la imagen son importantes pueden ser útiles también.
- Extender el modelo para reconocer katakana y kanji básicos, o desarrollar un sistema que reconozca palabras completas utilizando RNN o

Transformers. Un proyecto que involucra mayor complejidad pero que sería muy útil en situaciones donde se necesite ayuda para traducción por ejemplo.

- Complementar el software de práctica de hiragana con una aplicación completa relacionada al aprendizaje del idioma japonés. Mejorando la app para dar feedback sobre los trazos, poner caracteres de referencia y demás opciones que mejoran la experiencia de usuario.

Conclusiones

En este estudio se comprobó que las redes convolucionales son altamente efectivas para reconocer caracteres de hiragana escritos a mano. El modelo entrenado desde cero alcanzó un desempeño excepcional y mostró una gran capacidad de generalización, incluso con datos externos. El modelo de transfer learning también presentó resultados sólidos, aunque fue más susceptible a variaciones en la opacidad y en la forma del trazo.

Los modelos desarrollados permitieron demostrar que es posible construir un sistema funcional para apoyar el aprendizaje del alfabeto hiragana mediante técnicas de visión por computadora. Tanto la CNN como el modelo de transfer learning lograron identificar patrones esenciales de los caracteres, aunque la CNN mostró mayor estabilidad frente a variaciones externas.

El análisis con LIME brindó una comprensión más profunda del proceso de clasificación, permitiendo visualizar cómo los modelos interpretan elementos específicos del trazo y por qué ocurren ciertas confusiones. Esto no solo valida la solidez del enfoque, sino que también orienta la mejora futura del sistema.

La aplicación final confirma que estos modelos pueden utilizarse de manera efectiva en un entorno educativo, proporcionando retroalimentación inmediata que ayuda al usuario a mejorar su caligrafía y familiarizarse con las formas correctas de los caracteres.

Referencias

Handwritten Japanese Hiragana characters. (2025, August 8). Kaggle. https://www.kaggle.com/datasets/farukece/handwritten-japanese-hiragana-characters/data

School, C. J. L. (2025, October 28). *Guide to Japanese writing system: Kanji, Hiragana, and Katakana*. Coto Japanese Academy. https://cotoacademy.com/japanese-writing-system-kanji-hiragana-and-katakana-explained/

Staff, G. G. N. (2024, May 10). Alfabeto japonés y sus diferentes tipos. *Go! Go! Nihon*. https://gogonihon.com/es/blog/alfabeto-japones/

Gupta, M. (2022, 8 noviembre). Interpreting CNNs using LIME. Medium. https://medium.com/data-science-in-your-pocket/interpreting-cnns-using-lime-5a0bfb225708