Proyecto 3: OCR de texto manuscrito

Laura Nagamine, Jesús Niño, Gabriel Frisancho, Andrea Pérez
Universidad de Ingeniería y Tecnología (UTEC)
{laura.nagamine, jesus.nino, gabriel.frisancho, andrea.perez}@utec.edu.pe

I. Introducción

La transcripción automática de texto manuscrito constituye un problema fundamental en el reconocimiento de patrones debido a la variabilidad inherente del trazo humano. Este proyecto aborda la clasificación de caracteres alfanuméricos escritos a mano mediante técnicas de *machine learning*, con el objetivo de desarrollar un sistema capaz de identificar caracteres individuales (0–9, A–Z, a–z) con una precisión superior al 80%.

Para ello, se implementan y comparan tres enfoques de clasificación: Regresión Logística como modelo base, Random Forest para capturar relaciones no lineales y XGBoost como método avanzado de *ensemble learning* orientado a maximizar el rendimiento predictivo.

II. Conjunto de datos

El proyecto emplea dos conjuntos de datos complementarios: EMNIST para el entrenamiento y evaluación principal a nivel de carácter, e IAM Handwritten Forms como conjunto de validación adicional.

A. EMNIST (Extended MNIST)

EMNIST es una extensión del conjunto MNIST que incluye letras manuscritas además de dígitos. Está derivado de la Base de Datos Especial 19 de NIST y mantiene el mismo formato que MNIST, con imágenes de 28×28 píxeles en escala de grises.

1) Características del dataset: En este proyecto se utiliza la variante **ByClass**, cuyas características principales son:

- **Número de clases**: 62 (dígitos 0–9, letras mayúsculas A–Z, letras minúsculas a–z)
- **Tamaño de imagen**: 28×28 píxeles
- **Conjunto de entrenamiento**: 697 932 imágenes
- **Conjunto de prueba**: 116 323 imágenes
- **Total**: 814 255 caracteres manuscritos

2) Distribución de clases: El dataset presenta un desbalanceo intencional entre clases, como se observa en las Figuras 1 y 2. Según la documentación oficial, el volumen por letra refleja aproximadamente su frecuencia en el idioma inglés. Entre los patrones principales:

- **Dígitos (0–9)**: Clases mayoritarias, con ∼34 000–38 500 muestras por clase en entrenamiento y 5 500–6 500 en prueba.
- **Letras mayúsculas (A–Z)**: Volumen intermedio, con ∼8 000–25 000 muestras por clase.
- **Letras minúsculas (a–z)**: Clases más escasas, con ∼2 000–4 500 muestras en entrenamiento y 300–750 en prueba.

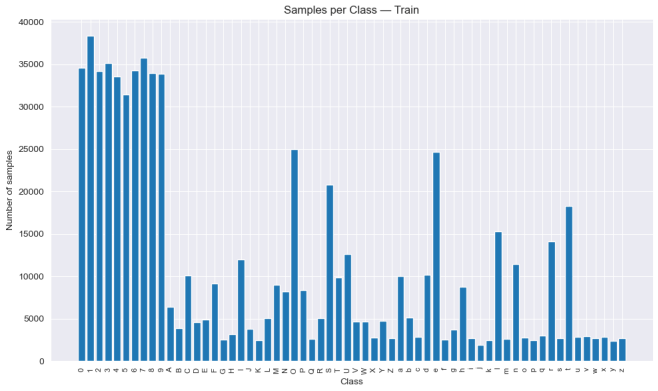

Fig. 1. Distribución de muestras por clase en el conjunto de entrenamiento EMNIST ByClass. Se observa un desbalanceo marcado: los dígitos poseen entre 10–15 veces más muestras que las letras minúsculas.

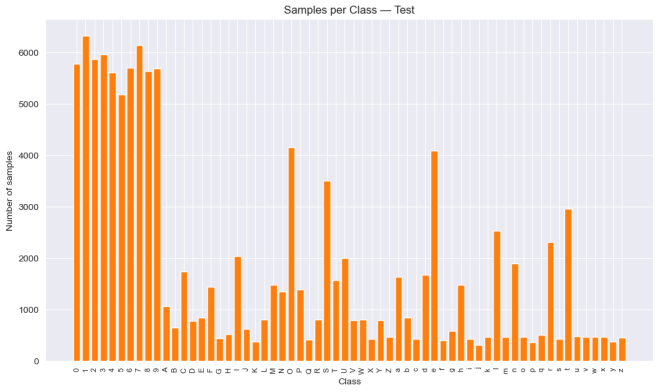

Fig. 2. Distribución de muestras por clase en el conjunto de prueba EMNIST ByClass. Se mantiene la proporción ∼6:1 entre entrenamiento y prueba para todas las clases.

La razón de desbalanceo máxima alcanza aproximadamente 19:1 entre el dígito “1” y la letra minúscula “z”. Este desbalanceo representa un desafío para los modelos, que tienden a favorecer las clases mayoritarias.

3) Proceso de conversión: Las imágenes originales de NIST están almacenadas como binarios de 128×128 píxeles. El proceso de conversión incluye:

1) Suavizado con filtro Gaussiano ($\sigma = 1$)
2) Extracción de la región que contiene el carácter

3) Centración manteniendo la relación de aspecto
4) Redimensionamiento a 28×28 píxeles

4) División entrenamiento–prueba: Siguiendo la metodología de MNIST, los conjuntos originales de NIST fueron combinados y luego divididos aleatoriamente, asegurando presencia de muestras de estudiantes de secundaria y empleados censales en ambos particiones.

B. IAM Handwritten Forms

El IAM Handwriting Database contiene formularios manuscritos en inglés utilizados ampliamente en investigación de reconocimiento de texto, identificación y verificación de escritores.

1) Características del dataset:

- **Formularios**: 1 066 producidos por ∼400 escritores
- **Instancias de palabras**: 82 227 (vocabulario de 10 841 palabras)
- **Contenido**: Texto manuscrito con estructura lingüística natural
- **Formato**: Imágenes sin restricciones de estilo

2) Variabilidad entre escritores: La Figura 3 muestra la distribución de formularios por escritor:

- **Concentración modal**: La mayoría de escritores aportó entre 1–3 formularios, representando ∼70–80% del total.
- **Cola larga**: Un grupo reducido contribuyó con 10 o más formularios, alcanzando casos extremos de 60+.
- **Implicaciones**: El dataset combina gran diversidad de estilos con sobrerrepresentación de algunos escritores prolíficos.

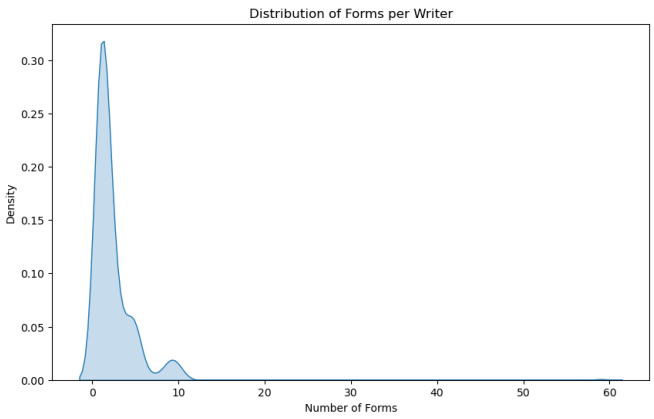

Fig. 3. Distribución de formularios por escritor en IAM Handwritten Forms. Se observa una cola larga propia de una distribución con sesgo positivo.

Esto contrasta con EMNIST, donde cada carácter proviene típicamente de un escritor distinto, generando menor repetición intra-escritor.

3) Rol en el proyecto: IAM se emplea como conjunto de validación externo para evaluar la generalización de los modelos entrenados con EMNIST y verificar su desempeño ante variabilidad real en escritura manuscrita.

C. Justificación de los datasets

La combinación EMNIST–IAM permite:

1) **EMNIST**: Entrenamiento controlado con caracteres segmentados y normalizados.
2) **IAM**: Validación en contexto real con alta variabilidad intra/inter-escritor.
3) **Complementariedad**: Aprendizaje de patrones fundamentales (EMNIST) y evaluación en escenarios reales (IAM).

D. Implicaciones del desbalanceo

El desbalanceo de EMNIST afecta tanto entrenamiento como evaluación:

- **Sesgo hacia clases mayoritarias**: Los modelos tienden a aprender preferentemente los dígitos.
- **Métricas**: La *accuracy* puede ser engañosa; se emplean F1-Score *weighted* y análisis por clase.
- **Mitigación**: Aunque no se aplicaron técnicas explícitas de rebalanceo, los métodos *ensemble* muestran cierta robustez.

III. Metodología

A. Pipeline general

El sistema se estructura en cuatro etapas principales:

1) Preprocesamiento: binarización adaptativa mediante Otsu y normalización.
2) Extracción de características: 1 436 *features* mediante 11 técnicas complementarias.
3) Reducción dimensional: *StandardScaler* + PCA + LDA (61 componentes).
4) Entrenamiento y evaluación: tres modelos de complejidad creciente con métricas multiclase balanceadas.

B. Preprocesamiento

1) Normalización y orientación: Las imágenes de EMNIST requieren ajustes estructurales:

1) Rotación de 90° y *flip* horizontal
2) Inversión de polaridad: $x' = 255 - x$
3) Normalización a rango [0, 1]

2) Binarización de Otsu: El método de Otsu determina un umbral óptimo T^* que minimiza la varianza intra-clase:

$$T^* = \arg\min_T \left[\omega_0(T)\sigma_0^2(T) + \omega_1(T)\sigma_1^2(T)\right].$$

Ventajas del enfoque por imagen:

- Adaptación a variaciones de contraste
- Reducción de ruido y variaciones de presión del lápiz
- Mayor separabilidad entre clases

C. Extracción de características

Se implementaron 11 familias de *features*, totalizando 1 436 características por imagen:

1) Zoning (72)
2) HOG (1 296)
3) Momentos de Hu (7)
4) LBP (10)
5) Skeleton (4)
6) Contornos (6)
7) *Pixel Run Length* (6)
8) Conectividad (2)
9) Geometría (7)
10) Proyecciones (24)
11) Densidad por cuadrantes (2)

D. Reducción dimensional

Se aplica el pipeline:

$$\mathbf{Z} = \text{LDA}(\text{PCA}(\text{StandardScaler}(\mathbf{X}))).$$

- **StandardScaler**: normalización z-score.
- **PCA**: reducción a 143 componentes ($\sim$95% varianza explicada).
- **LDA**: reducción supervisada a 61 componentes.

Este enfoque híbrido reduce de 1 436 a 61 dimensiones (95.8% de reducción) y acelera el entrenamiento sin pérdida significativa de información discriminativa.

IV. Implementación

Esta sección describe la implementación de los tres modelos evaluados: Regresión Logística, Random Forest y XGBoost. Para cada modelo se detallan la configuración, los hiperparámetros utilizados, los criterios de optimización y las consideraciones prácticas asociadas al entrenamiento en un escenario multiclase de alta dimensionalidad.

A. Regresión Logística

La Regresión Logística se emplea como baseline por su eficiencia en problemas lineales y su estabilidad en espacios de alta dimensionalidad tras la reducción PCA+LDA. El modelo se configuró en modo *multinomial* para manejar las 62 clases del dataset EMNIST.

1) Configuración del Modelo:

- **Solver**: `lbfgs`, adecuado para clasificación multiclase mediante maximización de log-verosimilitud.
- **Regularización**: L2 (valor por defecto), útil para estabilizar coeficientes con 61 componentes de entrada.
- **Máximo de iteraciones**: 1000, para asegurar convergencia completa dada la complejidad del problema.
- **Estrategia multiclase**: `multinomial`, optimizada para softmax sobre todas las clases simultáneamente.
- **Semilla**: 42, para garantizar reproducibilidad.

2) Consideraciones de Entrenamiento: El costo computacional es moderado debido al tamaño reducido del espacio de características (61 componentes). El solver `lbfgs` converge de manera estable sin requerir modificaciones adicionales. Este modelo sirve como línea base para evaluar el impacto de introducir no linealidad mediante los modelos posteriores.

B. Random Forest

El modelo Random Forest se implementa como alternativa no lineal capaz de capturar interacciones complejas entre los componentes LDA. Su estrategia basada en bagging mejora la robustez frente a ruido y desbalanceo de clases.

1) Configuración del Modelo:

- **Número de árboles (`n_estimators`)**: 300, seleccionado como equilibrio entre rendimiento y tiempo de entrenamiento.
- **Profundidad máxima**: No restringida, permitiendo capturar patrones detallados en datos de baja dimensionalidad.
- **Criterio**: `gini`, adecuado para clasificación multiclase.
- **Tamaño mínimo de hoja**: 1, para conservar capacidad discriminativa fina entre clases similares.
- **Número de características por división**: `sqrt`, estrategia estándar que reduce correlación entre árboles.
- **Semilla**: 42.

2) Consideraciones de Entrenamiento: El modelo mantiene una buena relación entre precisión y tiempo de cómputo. Su naturaleza ensemble permite mitigar parcialmente el desbalanceo del dataset, ya que diferentes subconjuntos bootstrap favorecen distintas distribuciones de clases durante el entrenamiento.

C. XGBoost

XGBoost se utiliza como modelo avanzado basado en boosting, capaz de capturar patrones no lineales y corregir errores iterativamente. Su eficiencia y capacidad de regularización lo convierten en un candidato sólido para escenarios multiclase complejos como EMNIST.

1) Configuración del Modelo:

- **Objetivo**: `multi:softmax`, adecuado para clasificación multiclase directa.
- **Número de clases**: 62.
- **Profundidad máxima**: 10, permitiendo capturar estructuras no lineales complejas sin sobreajuste excesivo.
- **Número de árboles**: 300, seleccionado tras evaluación preliminar de eficiencia.
- **Tasa de aprendizaje**: 0.1, balance entre estabilidad y velocidad de convergencia.
- **Subsample**: 0.8, para reducción de varianza.
- **Colsample_bytree**: 0.8, reduciendo correlación entre árboles en boosting.
- **Regularización**:
  - $\lambda = 1$ (L2)
  - $\alpha = 0$ (L1 desactivada)
- **Semilla**: 42.

2) Consideraciones de Entrenamiento: XGBoost se entrenó utilizando paralelización multinúcleo, permitiendo tiempos de cómputo razonables incluso con 300 árboles y profundidad elevada. La combinación de regularización y subsampling reduce el riesgo de sobreajuste, especialmente en clases minoritarias.

D. Infraestructura y Librerías Utilizadas

La implementación se desarrolló en Python 3.10 utilizando las siguientes librerías:

- **scikit-learn**: Regresión Logística y Random Forest
- **xgboost**: Modelo XGBoost
- **numpy** y **pandas**: Manipulación de datos
- **scikit-image**: Procesamiento de imágenes (Otsu, LBP, skeleton)
- **scipy**: Cálculo de propiedades geométricas y conectividad

Todos los experimentos se ejecutaron en una máquina con GPU NVIDIA T4 y 16GB RAM, aunque solo XGBoost se benefició parcialmente del soporte GPU.

V. Experimentación

A. Regresión Logística

1) Métricas de Rendimiento: Los resultados obtenidos en los conjuntos de entrenamiento y prueba se presentan en la Tabla I.

TABLE I
MÉTRICAS DE EVALUACIÓN - REGRESIÓN LOGÍSTICA

Métrica	Train	Test
Accuracy	81.27%	81.05%
Precision (weighted)	79.74%	79.23%
Recall (weighted)	81.27%	81.05%
F1-Score (weighted)	79.58%	79.37%

2) Análisis de Generalización: La diferencia entre las métricas de entrenamiento y test es mínima:

$$\Delta_{\text{accuracy}} = 0.8127 - 0.8105 = 0.0022 \ (0.22\%) \qquad (1)$$

Este pequeño gap indica que el modelo generaliza adecuadamente y no presenta signos significativos de overfitting.

3) Hallazgos Principales:

1) **Rendimiento consistente**: El modelo alcanza aproximadamente 81% de accuracy en ambos conjuntos, demostrando estabilidad.
2) **Buena generalización**: La diferencia mínima entre train y test ($< 1\%$) confirma ausencia de memorización de datos de entrenamiento.
3) **Balance precision-recall**: Los valores de precision (79.23%) y recall (81.05%) están bien balanceados, sin sesgo hacia falsos positivos o negativos.
4) **Eficiencia computacional**: El modelo converge rápidamente, haciendo iteraciones eficientes incluso con $\sim$698K muestras.
5) **Limitaciones identificadas**:
   - La precisión de 81% sugiere margen de mejora con modelos no lineales
   - Confusiones en clases visualmente similares (O vs 0, 1 vs I)

B. Random Forest

El modelo Random Forest alcanzó un rendimiento competitivo en la tarea de clasificación multiclase de EMNIST, logrando 81.48% de accuracy en el conjunto de prueba. Si bien su desempeño es ligeramente inferior al obtenido con XGBoost, supera el baseline establecido por Regresión Logística y demuestra alta capacidad para capturar patrones no lineales presentes en los componentes principales.

1) Métricas de Evaluación: La Tabla II presenta las métricas de rendimiento para Random Forest.

TABLE II
MÉTRICAS DE EVALUACIÓN - RANDOM FOREST

Métrica	Train	Test
Accuracy	97.62%	81.48%
Precision (weighted)	97.81%	79.45%
Recall (weighted)	97.62%	81.48%
F1-Score (weighted)	97.54%	79.15%

2) Análisis de Generalización: La diferencia de 16.13% en accuracy entre los conjuntos de entrenamiento y prueba indica **overfitting moderado**. Esto es consistente con el comportamiento típico de Random Forest cuando se utilizan árboles de gran profundidad (`max_depth=25`) y un número de árboles relativamente limitado (`n_estimators=100`). Aun así, el modelo mantiene un desempeño razonable sobre datos no vistos.

3) Distribución por Clases: El análisis de desempeño por clase evidencia patrones similares a los encontrados en otros modelos:

- **Mejor rendimiento**: Dígitos como 3, 6, 7 y 9, alcanzando F1-Scores superiores a 0.94 gracias a su forma distintiva
- **Mayor dificultad**: Caracteres visualmente ambiguos como I, l, 1, O, o y 0, donde el F1-Score cae por debajo de 0.60
- **Confusiones frecuentes**: Pares como O/0, I/1 y S/5 exhiben las mayores tasas de error debido a la similitud de sus trazos en formato manuscrito

La Fig. 4 ilustra las confusiones más frecuentes del modelo.

4) Importancia de Features: La importancia de los 61 componentes principales reveló que el modelo depende principalmente de los primeros PCs, los cuales concentran la mayor varianza del dataset. Sin embargo, todos los componentes mostraron importancia no nula, indicando que Random Forest utiliza efectivamente la totalidad de la representación reducida. La Fig. 5 muestra la distribución de importancia entre componentes.

5) Distribución de Confianza: Las probabilidades estimadas mediante la votación del ensamble mostraron la siguiente tendencia:

- Las predicciones correctas tuvieron una confianza media de 0.98
- Las predicciones incorrectas mostraron una confianza inferior, con media de 0.61

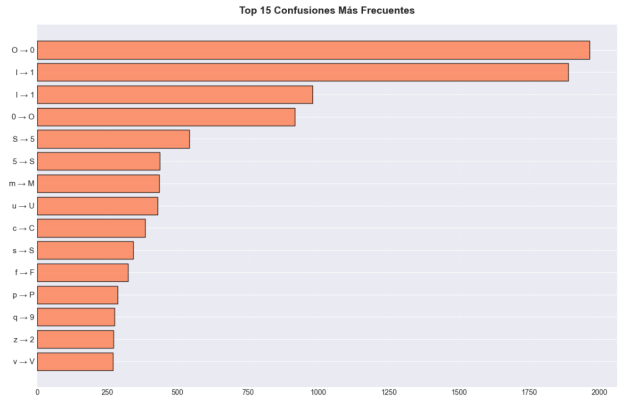

Fig. 4. Confusiones más frecuentes de Random Forest. Los pares de caracteres visualmente similares presentan las mayores tasas de error.

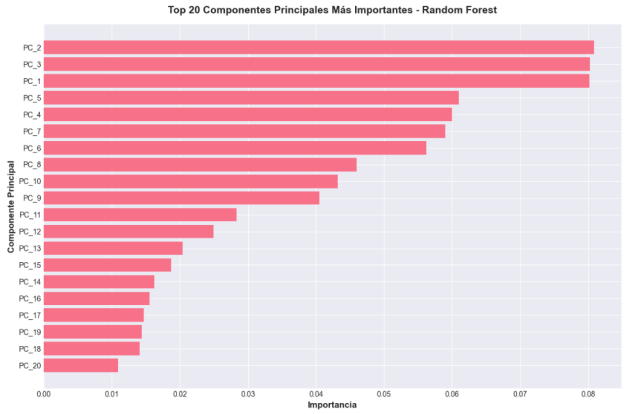

Fig. 5. Importancia de componentes principales en Random Forest. Los primeros componentes muestran mayor relevancia, consistente con la varianza explicada por PCA+LDA.

Esto sugiere que, aunque el modelo puede confundirse en clases similares, sus niveles de confianza reflejan correctamente la incertidumbre del clasificador. La Fig. 6 muestra la distribución completa de confianza.

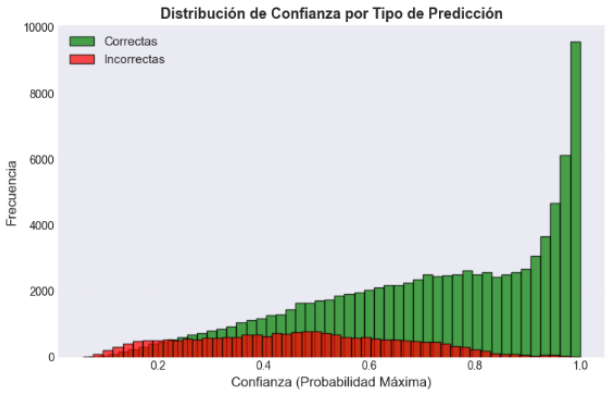

Fig. 6. Distribución de confianza en predicciones de Random Forest. Las predicciones correctas muestran mayor confianza que las incorrectas.

C. XGBoost

El modelo XGBoost demostró el mejor rendimiento entre todos los modelos evaluados, alcanzando 83.34% de accuracy en el conjunto de test, superando significativamente el baseline establecido por Regresión Logística (2.29 puntos porcentuales) y Random Forest (1.86 puntos porcentuales).

1) Métricas de Evaluación: La Tabla III presenta las métricas completas de rendimiento para XGBoost.

TABLE III
MÉTRICAS DE EVALUACIÓN - XGBOOST

Métrica	Train	Test
Accuracy	95.55%	83.34%
Precision (weighted)	95.80%	81.82%
Recall (weighted)	95.55%	83.34%
F1-Score (weighted)	95.33%	81.94%

2) Análisis de Generalización: Se observó una diferencia de 12.22% en accuracy entre train y test, indicando cierto grado de overfitting. Esto es esperado dada la capacidad de XGBoost para modelar relaciones complejas, pero sugiere espacio para mejora mediante técnicas de regularización más agresivas o aumento de datos. El gap es menor que el observado en Random Forest (16.13%), indicando mejor balance entre capacidad expresiva y generalización.

3) Distribución por Clases: El análisis por clase reveló patrones interesantes:

- **Mejor rendimiento**: Dígitos numéricos (7, 3, 6) y letras minúsculas comunes (e, d, r) con F1-Score ¿ 0.93
- **Problemas críticos**: Letras minúsculas o, s, c, m con F1-Score ¡ 0.18, principalmente debido a confusiones con sus contrapartes mayúsculas
- **Confusiones frecuentes**: Pares visualmente similares como O/0 (46.08%), l/1 (67.30%), m/M (85.56%)

La Fig. 7 ilustra las confusiones más problemáticas del modelo.

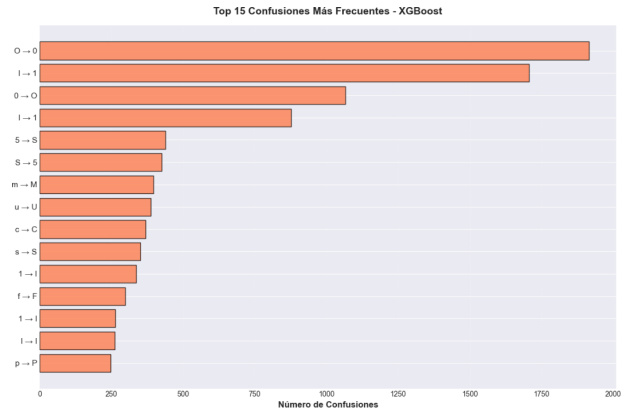

Fig. 7. Confusiones más frecuentes de XGBoost. Los pares mayúscula/minúscula y caracteres visualmente similares dominan los errores.

4) Curva de Aprendizaje: La curva de aprendizaje mostró convergencia estable, con log loss disminuyendo de 2.61 a

0.48 durante el entrenamiento. La mejora más significativa ocurrió en las primeras 100 iteraciones, estabilizándose posteriormente. El early stopping detuvo el entrenamiento en la iteración 483 de 500, indicando convergencia óptima sin sobreentrenamiento adicional. La Fig. 8 muestra la evolución del log loss.

Fig. 8. Curva de aprendizaje de XGBoost. Evolución del log loss durante el entrenamiento, mostrando convergencia estable con mejora significativa en las primeras 100 iteraciones.

5) Importancia de Features: El análisis de importancia reveló que los primeros componentes principales (PC_1, PC_3, PC_2) fueron los más relevantes para las decisiones del modelo, consistentemente con la varianza explicada por PCA+LDA. Los 61 componentes mostraron importancia no nula, indicando que toda la información reducida fue utilizada efectivamente. La Fig. 9 presenta la distribución de importancia.

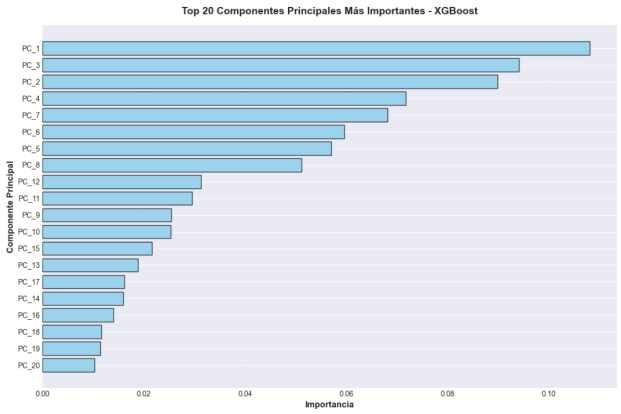

Fig. 9. Importancia de componentes principales en XGBoost. Los primeros componentes dominan, pero todos contribuyen al modelo.

6) Distribución de Confianza: Las predicciones correctas mostraron alta confianza (media: 0.90), mientras que las incorrectas tuvieron confianza moderada (media: 0.66). El 79% de las predicciones correctas y solo 25% de las incorrectas tuvieron confianza superior a 0.8, sugiriendo que el modelo está generalmente bien calibrado en sus estimaciones de probabilidad. La Fig. 10 muestra la distribución completa de confianza para ambos tipos de predicciones.

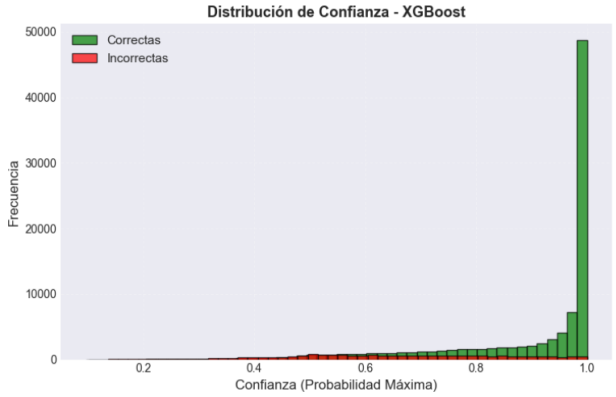

Fig. 10. Distribución de confianza en predicciones de XGBoost. Las predicciones correctas muestran mayor confianza que las incorrectas, indicando buena calibración del modelo.

VI. Discusión

Los resultados obtenidos permiten identificar patrones relevantes sobre la efectividad de distintos enfoques de *machine learning* para el reconocimiento de caracteres manuscritos. El análisis comparativo evidencia el balance entre complejidad, rendimiento, capacidad de generalización y costo computacional.

A. Análisis Comparativo de Modelos

La Tabla IV resume el rendimiento de los tres modelos considerados. Se observa que XGBoost alcanza el mejor desempeño global, aunque a costa de un tiempo de entrenamiento sustancialmente mayor.

TABLE IV
Comparación de rendimiento entre modelos

Modelo	Accuracy	F1	Tiempo	Overfit
Regresión Logística	81.05%	0.79	5 min	0.22%
Random Forest	81.48%	0.79	45 min	16.13%
XGBoost	83.34%	0.82	129 min	12.22%

XGBoost supera a Random Forest por 1.86 puntos porcentuales y a Regresión Logística por 2.29, consolidándose como el modelo más robusto. No obstante, el incremento en precisión viene acompañado de un costo computacional considerable, lo cual limita su aplicabilidad en sistemas con restricciones de tiempo o recursos.

B. Patrones de Error Comunes

Los tres modelos muestran tendencias similares en cuanto a los tipos de errores cometidos. En particular, presentan dificultades recurrentes en pares o grupos de caracteres con alta similitud visual:

- **O/0**: 46.08% de confusiones (el caso más crítico).
- **I/l/1**: 67.30% de confusiones en este conjunto de trazos lineales.
- **Mayúsculas/Minúsculas**: especialmente m/M (85.56%) y c/C (85.65%).

Estos patrones indican que las limitaciones provienen principalmente de la representación de características y la naturaleza intrínsecamente ambigua de ciertos caracteres, más que de deficiencias de un modelo específico.

C. Impacto del Desbalanceo

Las clases minoritarias, en particular varias letras minúsculas como `o`, `s` y `c`, presentan F1-score inferiores a 0.18. Esto se vincula directamente a su baja representación en el dataset: los dígitos poseen entre 10 y 15 veces más muestras que las letras minúsculas, lo que limita la capacidad de los modelos para aprender rasgos distintivos. Este fenómeno afecta de forma similar a los tres enfoques evaluados.

D. Limitaciones Identificadas

- **Sensibilidad al desbalanceo**: Los modelos muestran sesgo hacia clases mayoritarias, perjudicando el reconocimiento de letras con baja frecuencia.
- **Dependencia de la segmentación**: El pipeline asume entrada perfectamente segmentada, lo que limita la aplicabilidad a escenarios con escritura continua o ruido de fondo.
- **Overfitting en modelos complejos**: Tanto XGBoost como Random Forest exhiben brechas significativas entre rendimiento en entrenamiento y prueba, señalando la necesidad de técnicas adicionales de regularización.

E. Implicaciones Prácticas

Los hallazgos sugieren distintas recomendaciones según el contexto de uso:

- **XGBoost**: adecuado para aplicaciones donde la precisión es la prioridad principal.
- **Regresión Logística**: apropiada en sistemas de tiempo real o con recursos limitados.
- **Rebalanceo**: necesario para mejorar el reconocimiento de clases minoritarias en entornos con alta variabilidad de escritura.

VII. Conclusiones

El proyecto demuestra la viabilidad de implementar un sistema de reconocimiento de caracteres manuscritos que supera el umbral de precisión del 80%. El análisis revela comportamientos comunes entre modelos y proporciona bases para mejoras futuras.

A. Resultados Clave

1) **XGBoost alcanzó el mejor rendimiento** (83.34% de *accuracy*), aunque con el mayor costo computacional.
2) **El pipeline de preprocesamiento fue efectivo**, permitiendo que incluso modelos lineales superen el 81% de precisión.
3) **Las confusiones sistemáticas** entre caracteres visualmente similares evidencian limitaciones del espacio de características.
4) **El desbalanceo del dataset** afecta considerablemente el rendimiento en letras minúsculas.

B. Logros Obtenidos

- Desarrollo de un pipeline completo, evaluado y reproducible.
- Comparación exhaustiva de tres enfoques representativos de aprendizaje supervisado.
- Implementación de una interfaz gráfica funcional para validación práctica del reconocimiento.

C. Direcciones Futuras

1) **Rebalanceo de clases**: Aplicación de técnicas como SMOTE, *class weighting* o *data augmentation*.
2) **Transfer Learning**: Incorporación de redes preentrenadas mediante *fine-tuning*.
3) **Enfoques End-to-End**: Integración de segmentación y reconocimiento mediante CNNs o Transformers.
4) **Optimización de Hiperparámetros**: Uso de *GridSearch* o *Bayesian Optimization* para mejorar rendimiento.
5) **Interpretabilidad**: Evaluación de decisiones del modelo mediante SHAP para comprender patrones de clasificación.

D. Implicaciones Prácticas

Los resultados tienen aplicaciones potenciales en diversos ámbitos:

- **Digitalización**: Procesamiento de documentos históricos o formularios manuscritos.
- **Educación**: Sistemas de corrección automática de ejercicios o lectoescritura.
- **Accesibilidad**: Herramientas de conversión de texto manuscrito a digital para usuarios con discapacidad.

En síntesis, este trabajo establece una base sólida para la construcción de sistemas eficientes de reconocimiento de escritura, demostrando que enfoques tradicionales bien diseñados pueden ofrecer resultados competitivos, y señalando áreas donde métodos más avanzados pueden aportar mejoras significativas.

VIII. Declaración de Contribuciones

TABLE V
Contribuciones de los integrantes del equipo

Integrante	Contribución
Laura Nagamine	Preprocesamiento, implementación de regresión logística e interfaz gráfica
Jesús Niño	Preprocesamiento, feature extraction, dimensionality reduction y documentación de metodología
Gabriel Frisancho	Implementación y optimización de XGBoost
Andrea Pérez	Implementación y análisis de Random Forest

References

[1] E. Crawford, "EMNIST: Extended MNIST dataset," Kaggle, 2017. [Online]. Available: https://www.kaggle.com/datasets/crawford/emnist

[2] N. Abdalghani, "IAM Handwritten Forms dataset," Kaggle, 2020. [Online]. Available: https://www.kaggle.com/datasets/naderabdalghani/iam-handwritten-forms-dataset

[3] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proc. IEEE*, vol. 86, no. 11, pp. 2278–2324, Nov. 1998.
[4] H. Ketabdar, "MNIST dataset," Kaggle, 2017. [Online]. Available: https://www.kaggle.com/datasets/hojjatk/mnist-dataset
[5] U.-V. Marti and H. Bunke, "The IAM-database: An English sentence database for offline handwriting recognition," *Int. J. Document Analysis and Recognition*, vol. 5, no. 1, pp. 39–46, Nov. 2002.
[6] "IAM Handwriting Database," Research Group on Computer Vision and Artificial Intelligence, University of Bern. [Online]. Available: https://fki.tic.heia-fr.ch/databases/iam-handwriting-database

IX. Apéndices

- **Repositorio GitHub**: https://github.com/lauranagamine1/ml-p3-character-recognition
- **Demo de la Interfaz Gráfica**: https://youtu.be/I_$yKOLN3Jo$