IA - Clase 3A

(ML – Machine Learning)

Aprendizaje de Máquina

Ejercicio para comparar resultados con K-NN (784D y HOG), K-Means, GMM y CNN simple.

Aprendizaje de Máquina (ML)
Ejercicio para comparar modelos

- ¿Qué significa entrenar un modelo?
  - Es el proceso de ajustar sus parámetros internos (pesos, coeficientes, centroides, etc.) para que aprenda a realizar una tarea, como clasificar letras en EMNIST.
  - Datos de entrada → imágenes 28×28 píxeles (tensores).
  - Modelo → puede ser una red neuronal, una regresión logística, un KNN, etc.
  - Parámetros → son números internos del modelo (ejemplo: pesos sinápticos en una red).
  - Objetivo → que, al darle una imagen, el modelo prediga la clase correcta (A, B, C, …).
  - Extended MNIST. Modified National Institute of Standards and Technology.

Aprendizaje de Máquina (ML)
Ejercicio para comparar modelos

- Etapas del entrenamiento
  - Inicialización.
    - El modelo arranca con pesos aleatorios o predefinidos.
  - Forward pass (predicción)
    - Se le pasa una imagen → el modelo genera una predicción (ej: cree que la letra es “C”).
  - Función de pérdida (loss) $\hat{P}$
- Compara la predicción del modelo contra la etiqueta real.
  - Ejemplo: predijo “C”
  - La etiqueta era “A”
- La pérdida mide cuán “mal” estuvo.
  - Fórmula típica:
  - Clasificación → se usa la entropía cruzada
  - Regresión → se usa el error cuadrático medio

Aprendizaje de Máquina (ML)
Ejercicio para comparar modelos

- Backward pass (retropropagación)
  - El error se propaga hacia atrás para calcular cómo ajustar cada peso.
- Actualización de parámetros.
  - Se usa un optimizador (ej. gradiente descendente, Adam, SGD) que ajusta los pesos un poquito para reducir el error.
- Iteraciones (epochs)
  - Se repite el proceso sobre todo el dataset muchas veces hasta que el modelo “aprenda”.

Aprendizaje de Máquina (ML)
Ejercicio para comparar modelos

- Ejemplos de modelos:
  - Regresión logística multinomial (Softmax)
    - Logits: $z = Wx + b, \quad W \in \mathbb{R}^{26\times 784},\ x \in \mathbb{R}^{784},\ b \in \mathbb{R}^{26}$
    - Softmax (probabilidades): $p_k = \dfrac{e^{z_k}}{\sum_{j=1}^{26} e^{z_j}}, \quad k = 1, \ldots, 26$
    - Predicción: $\hat{y} = \arg\max_k p_k$
    - Pérdida (entropía cruzada, una muestra):

      $\mathcal{L} = -\sum_{k=1}^{26} y_k \log p_k$ (con y one-hot)
  - Perceptrón Multicapa (MLP) con 1 capa oculta
    - Aplanado: $x \in \mathbb{R}^{784}$
    - Capa oculta (ReLU):

      $h = \sigma(W_1 x + b_1), \quad W_1 \in \mathbb{R}^{m\times 784},\ b_1 \in \mathbb{R}^m,\ \sigma(u) = \max(0, u)$
    - Salida (logits y softmax):

      $z = W_2 h + b_2, \quad W_2 \in \mathbb{R}^{26\times m},\ b_2 \in \mathbb{R}^{26}, p_k = \dfrac{e^{z_k}}{\sum_{j=1}^{26} e^{z_j}}$
    - Pérdida: $\mathcal{L} = -\sum_{k=1}^{26} y_k \log p_k$

Aprendizaje de Máquina (ML)
Ejercicio para comparar modelos

- Logits son los valores crudos de salida de un modelo antes de aplicar una función de activación como sigmoid o softmax.
- Son los números reales (pueden ser negativos o positivos, incluso muy grandes) que la red neuronal calcula en la última capa lineal.
- Formalmente:
  - Si la última capa de una red es una transformación lineal
  - $z=wx+b$
    - Donde w son los pesos, x es el vector de entrada y b el sesgo, entonces z son los logits.
- No son probabilidades, porque no están en el rango [0,1].
- No necesariamente suman 1 en problemas multiclase.

Aprendizaje de Máquina (ML)
Ejercicio para comparar modelos

- K-NN
  - No hay pesos ni entrenamiento clásico.
  - El modelo se guarda todos los datos de entrenamiento (imágenes + etiquetas).
  - Para clasificar una nueva imagen: Se mide la distancia (usualmente Euclídea) entre esa imagen y cada imagen de entrenamiento.
  - Se eligen los k vecinos más cercanos.
  - Se predice la clase más frecuente entre esos vecinos (mayoría).
  - Distancia Euclídea entre el vector de entrada x y un ejemplo de entrenamiento x_i

$$d(x, x_i) = \sqrt{\sum_{j=1}^{784} (x_j - x_{i,j})^2}$$

  - Conjunto de índices de los k vecinos más cercanos

$$N_k(x) = \text{argmin}_{i=1,\ldots,n}^{k} \; d(x, x_i)$$

  - Predicción por mayoria de votos $\hat{y} = \text{mode}\{y_i : i \in N_k(x)\}$

Aprendizaje de Máquina (ML)
Ejercicio para comparar modelos

- K-Means: objetivo de cuantización
  - Minimiza el error intra-cluster (distancia al centro).
  - Útil para compresión, prototipos e inicialización de GMM.
- Algoritmo de Lloyd
  - Paso E (asignación): asignar cada punto al centro más cercano.
  - Paso M (actualización): recomputar cada centro como media del cluster.
-

$$z_i = \operatorname{argmin}_{c \in \{1,\dots,K\}} \; |x_i - \mu_c|_2^2$$

$$\mu_c = \frac{1}{N_c} \sum_{i=1}^{N} x_i \, \mathbf{1}\{z_i = c\}, \quad N_c = \sum_{i=1}^{N} \mathbf{1}\{z_i = c\}$$

Aprendizaje de Máquina (ML)
Ejercicio para comparar modelos

- K-Means: convergencia y clasificación
  - El coste desciende monótonamente hasta un mínimo local.
  - Clasificación: cluster→clase por mayoría (train).
  - Predicción = clase del centro más cercano.
  - Pseudo-probabilidades: normalizar inversos de distancias a centros.

$$J^{(t+1)} \leq J^{(t)}$$

$$\hat{y}(x) = \text{argmax}_y \sum_{i=1}^{N} \mathbf{1}\{y_i = y\}\, \mathbf{1}\{z_i = z^\star(x)\}$$

$$z^\star(x) = \text{argmin}_c \ |x - \mu_c|_2^2$$

Aprendizaje de Máquina (ML)
Ejercicio para comparar modelos

- GMM: modelo y log-verosimilitud
  - Mezcla gaussiana: combinación convexa de gaussianas con pesos \(\pi_k\).
  - Parámetros: medias \(\mu_k\), covarianzas \(\Sigma_k\), pesos \(\pi_k\).

$$p(x) = \sum_{k=1}^{K} \pi_k \, \mathcal{N}(x \mid \mu_k, \Sigma_k), \quad \sum_{k=1}^{K} \pi_k = 1, \ \pi_k \geq 0$$

$$\mathcal{L}(\theta) = \sum_{i=1}^{N} \log\left(\sum_{k=1}^{K} \pi_k \, \mathcal{N}(x_i \mid \mu_k, \Sigma_k) \right)$$

Aprendizaje de Máquina (ML)
Ejercicio para comparar modelos

- GMM: algoritmo EM
  - E-step: responsabilidades \(\gamma_{ik}\) (posterior del componente).
  - M-step: actualizar \(\mu_k\), \(\Sigma_k\), \(\pi_k\) ponderando por \(\gamma_{ik}\).

$$\gamma_{ik} = \frac{\pi_k \, \mathcal{N}(x_i \mid \mu_k, \Sigma_k)}{\sum_{j=1}^{K} \pi_j \, \mathcal{N}(x_i \mid \mu_j, \Sigma_j)}$$

$$N_k = \sum_{i=1}^{N} \gamma_{ik}, \quad \mu_k = \frac{1}{N_k} \sum_{i=1}^{N} \gamma_{ik} \, x_i$$

Aprendizaje de Máquina (ML)
Ejercicio para comparar modelos

- GMM: clasificación y relación con K-Means
  - Clasificación: componente→clase por mayoría (train); sumar responsabilidades por clase (test).
  - Límite: \(\Sigma_k=\sigma^2 I,\ \sigma\to 0\) ⇒ responsabilidades duras (K-Means).

$$\text{label}(k) = \text{argmax}_y \sum_{i=1}^{N} \mathbf{1}\{y_i = y\}\, \mathbf{1}\{\text{argmax}_j\, \gamma_{ij} = k\}$$

$$p(y = c \mid x) = \sum_{k\,:\,\text{label}(k)=c} \gamma_k(x), \quad \hat{y}(x) = \text{argmax}_c\ p(y = c \mid x)$$

$$\Sigma_k = \sigma^2 I,\ \sigma \to 0 \ \Rightarrow\ \gamma_{ik} \in \{0, 1\}$$

Aprendizaje de Máquina (ML)
Ejercicio para comparar modelos

- Selección de K (BIC)
  - En no supervisado: elegir K con BIC/AIC (penalización por complejidad).
  - Para EMNIST Letters como clasificador: K=26 por semántica

$$\mathrm{BIC} = \log L - \frac{p}{2}\log N, \quad p = K\left[d + \frac{d(d+1)}{2}\right] + (K-1)$$

Aprendizaje de Máquina (ML)
Ejercicio para comparar modelos

- Clasificar un conjunto (set) de imagenes sencillas.
- Letras: quiero ingresar letras y que las clasifique en base a la base del EMINST (Extended MNIST) que es una extensión del dataset MNIST (dígitos manuscritos).
- Formato: cada imagen es en escala de grises, 28×28 píxeles, igual que MNIST.
- Etiquetas: números enteros que representan la clase (0–61 según la variante).

Aprendizaje de Máquina (ML)
Ejercicio para comparar modelos

- Variantes:
  - EMNIST ByClass: 814,255 caracteres, 62 clases (10 dígitos + 26 mayúsculas + 26 minúsculas).
  - EMNIST ByMerge: 814,255 caracteres, 47 clases (se fusionan mayúsculas y minúsculas similares, como C/c).
  - EMNIST Balanced: 131,600 caracteres, 47 clases (subconjunto balanceado).
  - EMNIST Letters: 145,600 caracteres, 26 clases (solo letras, sin distinción mayúscula/minúscula).
  - EMNIST Digits: 280,000 caracteres, 10 clases (solo dígitos).
  - EMNIST MNIST: 70,000 dígitos (idéntico al MNIST original, sirve para consistencia).
- Trabajaremos con EMNIST Letters (26 clases, 145 600 imágenes).
- Descarga del dataset
  - datasets.EMNIST(..., download=True) baja un archivo comprimido grande (cientos de MB).
  - Una vez descargado, queda guardado en ./data y ya no se vuelve a bajar.

Aprendizaje de Máquina (ML)
Ejercicio para comparar modelos

- Descompresión / preparación de archivos
  - EMNIST viene en formato IDX comprimido (.gz).
  - torchvision lo descomprime la primera vez.
  - Esta operación de descompresión y escritura suele tardar varios minutos (pero ocurre solo una vez).
- Construcción de índices internos
  - Al inicializar el objeto EMNIST, se abren los archivos IDX y se leen los encabezados (número de imágenes, resolución, etc.).
  - Luego de la primera vez en las ejecuciones siguientes no vuelve a descargar ni descomprimir.
  - Solo abre los archivos en disco. El acceso a cada imagen es inmediato porque se hace lazy loading (on-demand).

Aprendizaje de Máquina (ML)
Ejercicio para comparar modelos

- TRAIN: Es el grupo de datos que el modelo ve para aprender.
  - Se usa para ajustar los parámetros del modelo (pesos en una red neuronal, centroides en K-means, etc.).
  - EMNIST Letters: train = 124 800 imágenes.
  - Cada imagen tiene su etiqueta (1–26).
- TEST: grupo separado, que el modelo no vio nunca durante el entrenamiento.
  - Se usa para medir qué tan bien generaliza el modelo a datos nuevos.
  - EMNIST Letters: test = 20 800 imágenes.
  - Tienen el mismo formato que las de train, pero distintas instancias manuscritas.
- Si entreno y evalúo con el mismo conjunto, el modelo podría memorizar (overfitting) y dar una falsa sensación de que funciona bien.
  - Con test, verifico que realmente entiende los patrones y no solo repitió lo visto.

Aprendizaje de Máquina (ML)
Ejercicio para comparar modelos

- Utilizamos datase EMNIST
  - Letters
- Objetivo: comparar K-NN (784D y HOG), K-Means, GMM y una CNN simple.
- Procesos para preprocesado de carácter manual (letra t):
  - umbrales, morfología, bbox, resize, centrado, deskew.
- Visualizaciones:
  - matriz de confusión, vecinos, y regiones K-NN en PCA-2D.
- Grilla de pruebas desde archivo (grid.json) y reportes automáticos.

Aprendizaje de Máquina (ML)
Ejercicio para comparar modelos

- Dataset: EMNIST Letters
  - 26 clases: A..Z (mayús/minús fusionadas).
  - Etiquetas 1..26 ↔ A..Z.
- Corrección de orientación: rotar –90° + espejo horizontal (solo para el dataset).
- Muestreo estratificado:
  - mismo nº de ejemplos por letra
  - evita UndefinedMetricWarning
- División típica: ~15–25k train, ~5–8k test.

Aprendizaje de Máquina (ML)
Ejercicio para comparar modelos

- Preprocesado del bitmap (28×28 estilo EMNIST)
- Convertir a escala de grises (L), auto-invertir si el fondo es claro (letra clara/fondo negro).
- Umbral adaptativo (ventana y offset) u Otsu para hallar bbox; padding asimétrico para no cortar la ‘t’.
- Morfología: opening (quita ruido), closing (cierra cortes), dilatación horizontal leve (refuerza travesaño).
- Resize manteniendo aspecto (lado mayor=20) + ancho mínimo; centrar en 28×28; deskew con límites + recentrado por COM.

Aprendizaje de Máquina (ML)
Ejercicio para comparar modelos

- K-NN en 784D (baseline)
  - Característica: vector de 28×28 = 784 (intensidades normalizadas).
  - K vecinos, weights='distance' (vecinos cercanos pesan más).
  - Métrica: euclidean o cosine (más robusto a grosor de trazo).
  - Diagnóstico: vecinos más cercanos + votos por clase.

Aprendizaje de Máquina (ML)
Ejercicio para comparar modelos

- Visualizaciones para K-NN
  - Matriz de confusión (test):
    - Clases más confundidas (p. ej., T↔L/F).
- Vecinos más cercanos: inspección cualitativa de la decisión.
- Regiones K-NN (PCA-2D): solo visual para entender la frontera.
- Reportes: se guardan PNG y JSON/CSV con votos/predicciones.
- UndefinedMetricWarning: por qué aparece y cómo evitarlo
  - Ocurre si alguna clase no aparece en y_test o nunca es predicha.
  - Solución: muestreo estratificado y zero_division=0 en el reporte.
  - Aumentar cobertura por clase y ajustar K/métrica si hay clases nunca predichas.

Aprendizaje de Máquina (ML)
Ejercicio para comparar modelos

- K-NN con HOG (mejor para encontrar ‘t’ vs ‘l’)
  - HOG para 28×28: orientaciones=9, celdas 4×4, bloques 2×2, L2-Hys.
  - Captura bordes/direcciones: el travesaño de ‘t’ deja firma horizontal.
  - K-NN sobre HOG suele mejorar frente a intensidades puras (784D).

Aprendizaje de Máquina (ML)
Ejercicio para comparar modelos

- Desambiguador L↔T (post-proceso)
  - 1) Votos K-NN: si P(T) ≈ P(L) (margen pequeño),
  - 2) ‘Crossbar score’: energía horizontal en banda superior-central.
  - Si score ≥ τ y P(T) no muy inferior a P(L), reasignar L→T.
  - Parámetros típicos: τ≈1.05–1.20, margen≈0.05–0.10 (ajustar al trazo).

Aprendizaje de Máquina (ML)
Ejercicio para comparar modelos

- Grilla de pruebas (grid.json) + reportes
  - Archivo externo con experimentos: modelo (784D/HOG), K, métrica, umbrales, dilatación…
  - Cada experimento: genera bitmap_postproc.png, vecinos.png, regiones.png y report.json.
- resumen.csv con los resultados de la grilla.

Aprendizaje de Máquina (ML)
Ejercicio para comparar modelos

- K-Means (no supervisado) como clasificador
  - Entrenar con 26 clusters (A..Z).
  - Mapear cluster→clase por mayoría en train (pseudoetiquetado).
  - ‘Probabilidades’ por clase a partir de 1/distancia a centroides (normalizada). $\hat{P}$
  - Útil para explorar estructura; suele rendir menos que K-NN/CNN.

Aprendizaje de Máquina (ML)
Ejercicio para comparar modelos

- GMM (Gaussian Mixture) como clasificador
  - Entrenar con 26 componentes (full covariance).
  - Mapear componente→clase por mayoría; usar responsabilidades como prob.
  - Suaviza fronteras; puede capturar subformas (multi-modos) por letra.
  - Visualización: regiones en PCA-2D (solo visual).

Aprendizaje de Máquina (ML)
Ejercicio para comparar modelos

- GMM (Gaussian Mixture) como clasificador
  - Entrenar con 26 componentes (full covariance).
  - Mapear componente→clase por mayoría; usar responsabilidades como prob.
  - Suaviza fronteras; puede capturar subformas (multi-modos) por letra.
  - Visualización: regiones en PCA-2D (solo visual).

Aprendizaje de Máquina (ML)
Ejercicio para comparar modelos

- CNN simple (PyTorch)
  - Arquitectura: Conv(1→16)–MP–Conv(16→32)–MP–FC(64)–Softmax(26).
- Entrenar 5–15 épocas (estratificado ayuda).
- Evaluar: accuracy, matriz de confusión, top-8 probabilidades para tu bitmap.
- Regiones 2D: usar features de la penúltima capa + KNN 2D para visual.

Aprendizaje de Máquina (ML)
Ejercicio para comparar modelos

- Comparativa y recomendaciones
  - 784D vs HOG: HOG suele ganar en t/l/f y otros pares confusos.
  - K-Means/GMM: buenos para explorar; asignación por mayoría necesaria.
  - CNN: mejor performance si le dedicás más épocas y datos.
- Afinar preprocesado: adaptive offset, dilatación horizontal, ancho mínimo, deskew limitado.