# Memoria parcial de clasificación de artefactos arqueológicos




UNIVERSIDAD
COMPLUTENSE
MADRID

**UNIVERSIDAD COMPLUTENSE DE MADRID**

Facultad de Ciencias Matemáticas

Asignatura: Taller de tecnomatemática

Profesor: Antonio López Montes

**Autores:** Mario Bermejo Cuervo, Vanessa Stefani Carnero Olivares, Jesús Gutiérrez Díaz y Aroa Ortiz Lominchar

Abril de 2026

**Resumen**

El seguiente artículo se divide en dos bloques fundamentales. El primero compara tres arquitecturas ligeras para clasificación de artefactos arqueológicos: una CNN directa sobre imagen, una MLP alimentada por un VAE y una KAN también alimentada por un VAE. El segundo estudia una estrategia de *transfer learning* con `EfficientNet-B0` y compara dos posibles cabezas finales, una MLP y una KAN. En ambos casos la base experimental es el `cssl_dataset`.

# Índice general


**1. Modelos ligeros: CNN, VAE + MLP y VAE + KAN** **2**
- 1.1. Objetivo del experimento . . . 2
- 1.2. Dataset utilizado . . . 2
  - 1.2.1. Experimentos analizados . . . 3
- 1.3. Arquitecturas evaluadas . . . 3
  - 1.3.1. CNN directa sobre imagen . . . 3
  - 1.3.2. VAE compartido para MLP y KAN . . . 3
  - 1.3.3. Clasificador VAE + MLP . . . 4
  - 1.3.4. Clasificador VAE + KAN . . . 4
- 1.4. Entrenamiento . . . 4
- 1.5. Resultados obtenidos . . . 4
  - 1.5.1. Resultados sobre drawings . . . 4
  - 1.5.2. Resultados sobre photos . . . 5
  - 1.5.3. Comparación visual resumida . . . 6
- 1.6. Conclusión del primer bloque . . . 6

**2. Transfer learning con EfficientNet y comparación entre MLP y KAN** **7**
- 2.1. Objetivo y planteamiento . . . 7
- 2.2. Dataset y configuración experimental . . . 7
- 2.3. Arquitectura comparada . . . 7
- 2.4. Entrenamiento . . . 8
- 2.5. Resultados cuantitativos . . . 8
- 2.6. Análisis visual de ejemplos . . . 9
- 2.7. Discusión del segundo bloque . . . 12
- 2.8. Conclusión general . . . 12

# Capítulo 1

# Modelos ligeros: CNN, VAE + MLP y VAE + KAN

## 1.1. Objetivo del experimento

El objetivo de este bloque es comparar tres formas distintas de resolver la misma tarea de clasificación:

- una arquitectura visual directa, especializada en explotar la estructura espacial de la imagen (CNN);
- una arquitectura densa que trabaja sobre una representación latente comprimida (VAE + MLP);
- una arquitectura alternativa a la MLP, también sobre espacio latente, pero con capas KAN basadas en expansiones RBF (VAE + KAN).

La hipótesis de partida era que el espacio latente aprendido por el VAE podía facilitar la clasificación en problemas visuales modestos, y que una cabeza KAN podría capturar relaciones no lineales más expresivas que una MLP convencional.

## 1.2. Dataset utilizado

Los experimentos se han realizado sobre el conjunto `cssl_dataset`, en la tarea de clasificación por forma contenida en:

`cssl_dataset/experiments/regular_shape/shapes/train_sets`

El dataset contiene dos modalidades:

- `photos`: fotografías de artefactos.
- `drawings`: dibujos o representaciones esquemáticas de esos artefactos.

Las clases usadas en este bloque son diez:

`ankh, anthropomorphic, bands, beetle, bird, circles, cross, ibex, lion, sa`

En todos los casos las imágenes se leen en escala de grises y se redimensionan a 256×256. Durante entrenamiento se aplica una aumentación ligera basada en *horizontal flip* con probabilidad 0,5 y una rotación aleatoria de hasta 8°. Tras ello, la entrada se normaliza con media 0,5 y desviación típica 0,5.

### 1.2.1. Experimentos analizados

En esta memoria se recogen los resultados que ya entrenados y almacenados en un módulo anterior llamado `CNN_ARTEFACTOS` que está diseñado como versión ligera del proyecto. Este, por defecto limita el número de ejemplos a `400` en entrenamiento y `200` en validación/test para reducir el coste computacional.

Tabla 1.1: Configuración de los experimentos analizados.

| Modalidad | Split | Tamaño original | Muestras usadas | Épocas observadas |
|---|---|---|---|---|
| Drawings | `50-50 / experiment_0` | 507 train / 513 test | 400 train / 200 test | 20 |
| Photos | `50-50-q / experiment_0` | 129 train / 513 test | 129 train / 200 test | 50 |

De esta manera, la comparación principal es válida dentro de cada experimento, pero no debe interpretarse como una comparación directa entre dibujos y fotografías, ya que cambian tanto la modalidad como el split.

## 1.3. Arquitecturas evaluadas

### 1.3.1. CNN directa sobre imagen

La arquitectura `SmallArtefactCNN` recibe la imagen en escala de grises y la procesa directamente mediante cuatro bloques convolucionales. El número de canales evoluciona como:

$$1 \to 16 \to 32 \to 64 \to 96.$$

Cada bloque utiliza dos convoluciones $3 \times 3$, normalización `BatchNorm2d` y activación `ReLU`, con operaciones de `MaxPool2d` entre bloques. Al final se aplica una agregación global mediante `AdaptiveAvgPool2d` y una pequeña cabeza densa,

$$96 \to 64 \to 10.$$

Esta red sirve como *baseline*, ya que conserva la estructura espacial de la imagen durante casi todo el *pipeline* y aprende directamente patrones locales de borde, textura y forma.

### 1.3.2. VAE compartido para MLP y KAN

Las arquitecturas MLP y KAN no clasifican la imagen original directamente. Antes de clasificar, ambas pasan por un VAE ligero. Su encoder está formado por tres capas convolucionales estriadas, con progresión

$$1 \to 16 \to 32 \to 48.$$

Sobre esa representación se estiman $\mu$ y $\log \sigma^2$, se aplica la reparametrización gaussiana y, finalmente, un decoder con convoluciones transpuestas intenta reconstruir la imagen de entrada. El encoder produce un mapa latente de tamaño $1 \times 32 \times 32$, equivalente a 1024 características tras aplicar *flatten*. La idea es sustituir la imagen cruda por una representación más compacta antes de clasificar.

### 1.3.3. Clasificador VAE + MLP

La red `SmallArtefactMLP` toma el latente $32 \times 32$, lo aplana y lo clasifica con una MLP de tres capas densas:

$$1024 \to 256 \to 128 \to 10.$$

Las dos capas ocultas usan activación `ReLU` y `Dropout` de 0.25 y 0.15, respectivamente. En consecuencia, la tarea visual se transforma en una clasificación sobre un vector latente aprendido por el VAE.

### 1.3.4. Clasificador VAE + KAN

La red `SmallArtefactKAN` comparte el mismo VAE anterior, pero sustituye la MLP por una cabeza de clasificación basada en capas KAN:

$$1024 \to 96 \to 48 \to 10.$$

Las dos capas ocultas son `RBFKANLayer`, separadas por activaciones `SiLU` y un `Dropout` de 0.15. Cada capa KAN combina una parte lineal convencional con una expansión no lineal basada en funciones base radiales sobre una rejilla aprendible. La motivación es disponer de una no linealidad más estructurada que la de una capa densa estándar.

## 1.4. Entrenamiento

Los tres modelos se entrenan con optimizador `Adam`, tasa de aprendizaje inicial $10^{-3}$, `weight decay` de $10^{-4}$ y `batch size` de 16. En todos los casos se conserva como *checkpoint* final el estado correspondiente a la mejor `val_acc`.

La función de pérdida no es exactamente la misma para todos los modelos:

$$\mathcal{L}_{\text{CNN}} = \mathcal{L}_{\text{CE}} \tag{1.1}$$

$$\mathcal{L}_{\text{MLP/KAN}} = \mathcal{L}_{\text{CE}} + 0{,}35\,\mathcal{L}_{\text{recon}} + 0{,}02\,\mathcal{L}_{\text{KL}} \tag{1.2}$$

donde $\mathcal{L}_{\text{CE}}$ es la entropía cruzada de clasificación, $\mathcal{L}_{\text{recon}}$ es el error de reconstrucción del VAE y $\mathcal{L}_{\text{KL}}$ es la regularización KL del espacio latente.

Esto tiene una consecuencia importante a la hora de interpretar las figuras: la accuracy sí es comparable entre modelos, pero la loss total no debe compararse de forma absoluta entre CNN y los modelos con VAE, porque en MLP/KAN incorpora términos extra que la CNN no tiene.

## 1.5. Resultados obtenidos

### 1.5.1. Resultados sobre drawings

La comparación principal más limpia es la de `drawings / 50-50 / experiment_0`, donde se entrenaron las tres arquitecturas en el mismo contexto experimental.

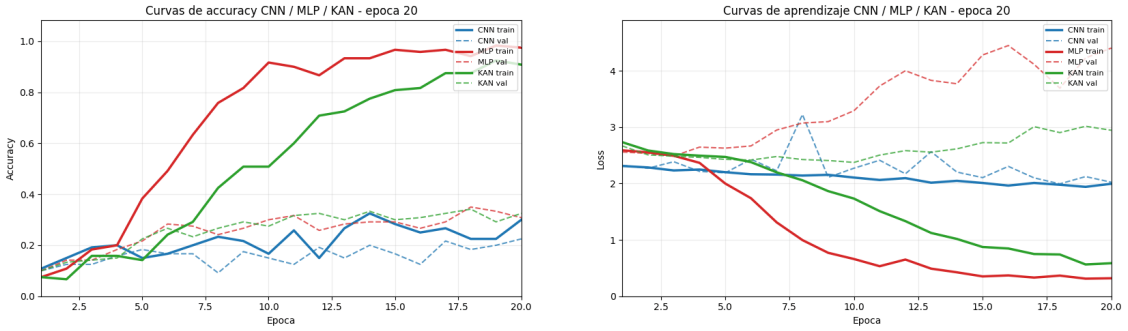
Tabla 1.2: Mejor resultado de validación sobre `drawings / 50-50 / experiment_0`.

| Modelo | Mejor época | Train acc. | Val acc. | Gap train-val |
|---|---|---|---|---|
| CNN | 20 | 0.3000 | 0.2250 | 0.0750 |
| MLP | 18 | 0.9417 | 0.3500 | 0.5917 |
| KAN | 18 | 0.8750 | 0.3417 | 0.5333 |

La lectura de estos resultados es clara:

- la MLP obtiene la mejor accuracy de validación (0.3500);
- la KAN queda muy cerca (0.3417);
- la CNN queda bastante por debajo (0.2250).

En este escenario, trabajar sobre un espacio latente aprendido por el VAE resulta más eficaz que clasificar el dibujo directamente con una CNN pequeña. Sin embargo, tanto la MLP como la KAN muestran un *gap* grande entre train y validación, lo que sugiere sobreajuste. La MLP logra una validación ligeramente mejor, pero también sobreajusta algo más que la KAN.




(a) Curvas de accuracy.

(b) Curvas de loss total.

Figura 1.1: Evolución del entrenamiento en el experimento con dibujos. Las curvas se han extraído de los GIFs generados por el propio módulo `CNN_ARTEFACTOS`.

### 1.5.2. Resultados sobre photos

Como complemento, también se dispone de un segundo experimento sobre `photos / 50-50-q / experiment_0`. En este caso la cantidad de entrenamiento es mucho menor, ya que el split original solo contiene 129 muestras de train.

Tabla 1.3: Mejor resultado de validación sobre `photos / 50-50-q / experiment_0`.

| Modelo | Mejor época | Train acc. | Val acc. | Gap train-val |
|---|---|---|---|---|
| CNN | 50 | 0.5659 | 0.2400 | 0.3259 |
| MLP | 44 | 0.8605 | 0.2000 | 0.6605 |
| KAN | 24 | 0.2248 | 0.1500 | 0.0748 |

En fotografías el patrón cambia:

- la CNN es la mejor (0.2400);
- la MLP queda por debajo (0.2000);
- la KAN es la peor (0.1500).

Aquí parece que el VAE no consigue construir una representación latente suficientemente robusta para mejorar la clasificación. Una explicación posible es que, con tan pocas fotografías de entrenamiento, la compresión previa del VAE introduce una dificultad adicional en lugar de ayudar. La CNN, aunque modesta, aprovecha mejor la estructura espacial directa de la imagen.

### 1.5.3. Comparación visual resumida

La Figura 1.2 resume la mejor `val_acc` observada para cada modelo en los dos escenarios disponibles.

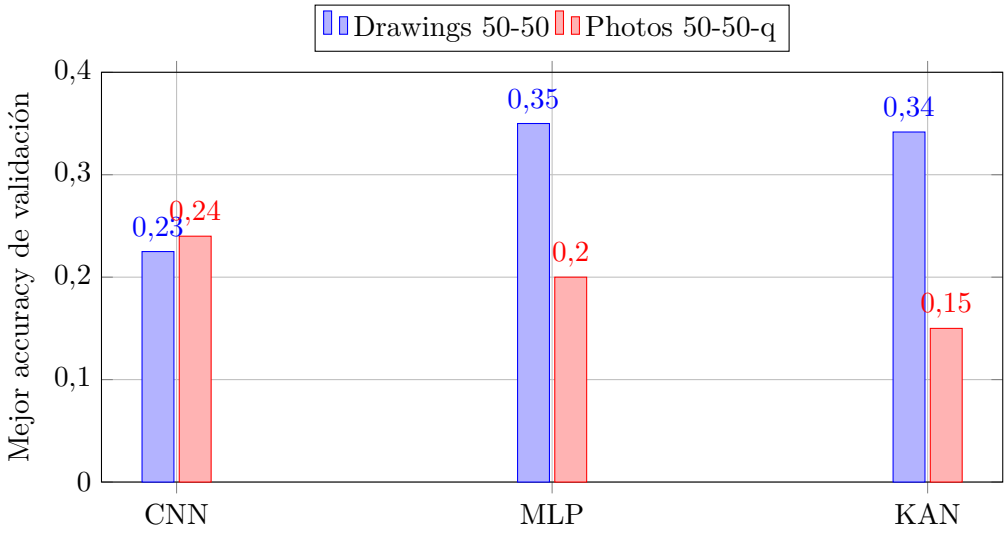



Figura 1.2: Resumen de la mejor accuracy de validación por modelo y escenario.

## 1.6. Conclusión del primer bloque

De los resultados anteriores se pueden extraer varias conclusiones relevantes:

1. No existe un ganador universal. El comportamiento depende de la modalidad y del tamaño efectivo del entrenamiento.
2. En dibujos, el espacio latente del VAE sí ayuda. Tanto MLP como KAN superan claramente a la CNN.
3. En fotografías, la CNN es más estable. Cuando el conjunto de entrenamiento es pequeño, la ruta directa sobre imagen parece más robusta que introducir un VAE previo.
4. KAN no mejora a MLP en los experimentos realizados. En dibujos (`drawings`) queda muy cerca, pero no la supera; en fotografías queda claramente por debajo.
5. MLP y KAN muestran sobreajuste. Los gaps entre accuracy de entrenamiento y de validación son altos, sobre todo en la MLP.

En consecuencia, la conclusión experimental de esta primera fase no es que KAN sea mejor que MLP, sino que la utilidad de la cabeza de clasificación depende del tipo de representación y del régimen de datos. En nuestro caso, la KAN es una alternativa interesante, pero en este proyecto no han demostrado una ventaja consistente frente a una MLP convencional cuando se trabaja con modelos ligeros y con un VAE intermedio.

De todos modos, ningún clasificador ha sido capaz de obtener una precisión considerada buena, ya que todas ellas se encuentran por debajo del umbral de 0.40.

# Capítulo 2

# Transfer learning con EfficientNet y comparación entre MLP y KAN

## 2.1. Objetivo y planteamiento

En el segundo bloque se abandonan las arquitecturas ligeras del primer experimento y se pasa a un esquema de *transfer learning*. La idea es aprovechar un *backbone* visual preentrenado, `EfficientNet-B0`, y comparar dos maneras de especializar su bloque final: una cabeza MLP convencional y una cabeza KAN. A diferencia del bloque anterior, aquí la entrada no es una sola imagen, sino un par formado por una fotografía y un dibujo del mismo artefacto.

## 2.2. Dataset y configuración experimental

El experimento se ha realizado sobre el split `80-20 / experiment_0` del mismo problema de clasificación por forma del `cssl_dataset`. Se mantienen las diez clases ya utilizadas en el primer bloque, pero cada muestra se construye emparejando una imagen de `photos` y una de `drawings` con el mismo `itemUUID`. De este modo, el modelo dispone simultáneamente de información fotográfica y de información esquemática del mismo objeto.

El conjunto emparejado contiene 817 pares para entrenamiento y 203 pares para validación/test. Las entradas se redimensionan a $224 \times 224$ y se procesan con pesos preentrenados de ImageNet, lo que permite reutilizar representaciones visuales generales antes de especializarlas al dominio arqueológico.

## 2.3. Arquitectura comparada

Ambas variantes comparten exactamente el mismo *backbone*. La fotografía y el dibujo se introducen por separado en una `EfficientNet-B0` compartida, de forma que se obtienen dos *embeddings* en el mismo espacio latente. A continuación, ambos *embeddings* se fusionan concatenando cuatro componentes: el embedding de la fotografía, el embedding del dibujo, su diferencia absoluta y su producto elemento a elemento.

Sobre esa representación fusionada se construye la clasificación final. En la variante MLP, la fusión pasa por una cabeza densa convencional antes de la capa de salida. En la variante KAN, el *backbone* y la fusión se mantienen, pero las capas ocultas de la cabeza final se sustituyen por capas `RBFKANLayer`. Por tanto, la comparación es bastante limpia, ya que el cambio principal se concentra en el bloque final de decisión.

## 2.4. Entrenamiento

El entrenamiento se realiza en dos fases. En una primera fase se congela el *backbone* y se entrena solo la cabeza nueva; después se desbloquea el último bloque de `EfficientNet` para hacer un ajuste fino. En ambos modelos se usa `batch size` 8, *label smoothing*, regularización por `dropout`, *scheduler* de tasa de aprendizaje y *early stopping*. Además de la accuracy directa de clasificación, también se evalúa una métrica híbrida que combina la salida del clasificador con distancias a prototipos de clase en el espacio de *embeddings*.

Aunque la filosofía general es la misma, las configuraciones finales no son idénticas. La variante MLP se lanzó con un máximo de 60 épocas, 10 épocas de congelación y `dropout` 0.45, mientras que la variante KAN se lanzó con un máximo de 50 épocas, 8 épocas de congelación, `dropout` 0.35 y 4 nudos por capa KAN. En ambos casos el entrenamiento terminó antes del máximo previsto por acción del *early stopping*.

## 2.5. Resultados cuantitativos

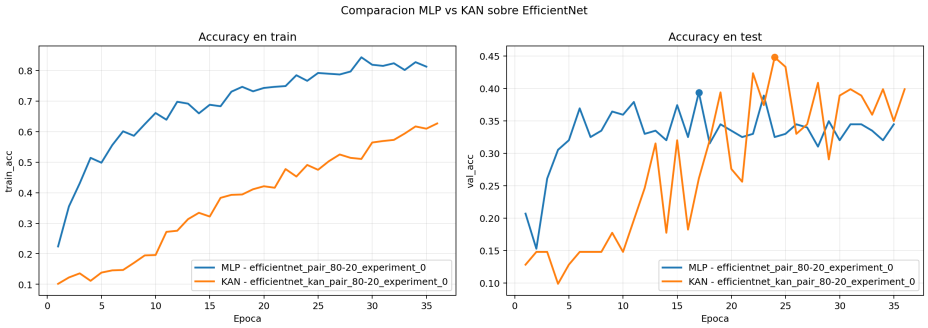
La Tabla 2.1 resume los resultados principales de la comparación.

Tabla 2.1: Comparación entre `EfficientNet + MLP` y `EfficientNet + KAN`.

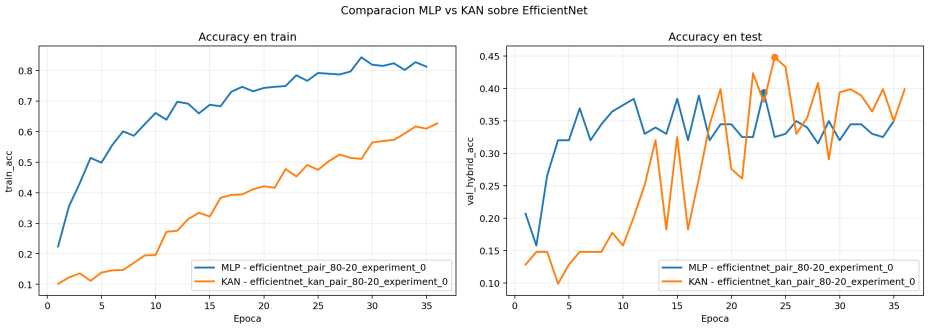
| Modelo | Épocas entrenadas | Mejor época | Train acc. | Val acc. | Val hybrid acc. | Val loss |
|---|---|---|---|---|---|---|
| EfficientNet + MLP | 35 | 17 | 0.7307 | 0.3941 | 0.3892 | 2.2306 |
| EfficientNet + KAN | 36 | 24 | 0.4908 | 0.4483 | 0.4483 | 1.6986 |

El resultado principal es que la variante KAN supera a la variante MLP en esta comparación. La mejora en accuracy de validación es de aproximadamente 5.4 puntos porcentuales, al pasar de 0.3941 a 0.4483. La misma tendencia aparece en la métrica híbrida, donde KAN vuelve a situarse por encima.

Además, la variante KAN alcanza esos resultados con una accuracy de entrenamiento claramente menor en su mejor época. Esto sugiere un comportamiento más contenido y una mejor generalización. En cambio, la cabeza MLP aprende con mayor facilidad el entrenamiento, pero no convierte esa ventaja en una mejora de validación. La diferencia de `val_loss` también favorece a KAN, lo que refuerza la idea de una separación más estable entre clases en el conjunto de validación.




(a) Curvas de `train_acc` y `val_acc`.




(b) Curvas de `train_acc` y `val_hybrid_acc`.

Figura 2.1: Comparación de las curvas de aprendizaje para las dos cabezas finales sobre `EfficientNet`.

Las curvas de la Figura 2.1 apoyan esta lectura. La cabeza MLP alcanza niveles de entrenamiento más altos, pero sus curvas de validación se estabilizan antes y en un nivel inferior. La cabeza KAN progresa de forma más gradual y consigue una validación final más alta y más consistente.

## 2.6. Análisis visual de ejemplos

Para complementar la comparación numérica se generaron mosaicos de ejemplos del conjunto de test. Las Figuras 2.2 y 2.3 permiten inspeccionar cualitativamente el comportamiento de ambas variantes sobre ocho pares representativos.

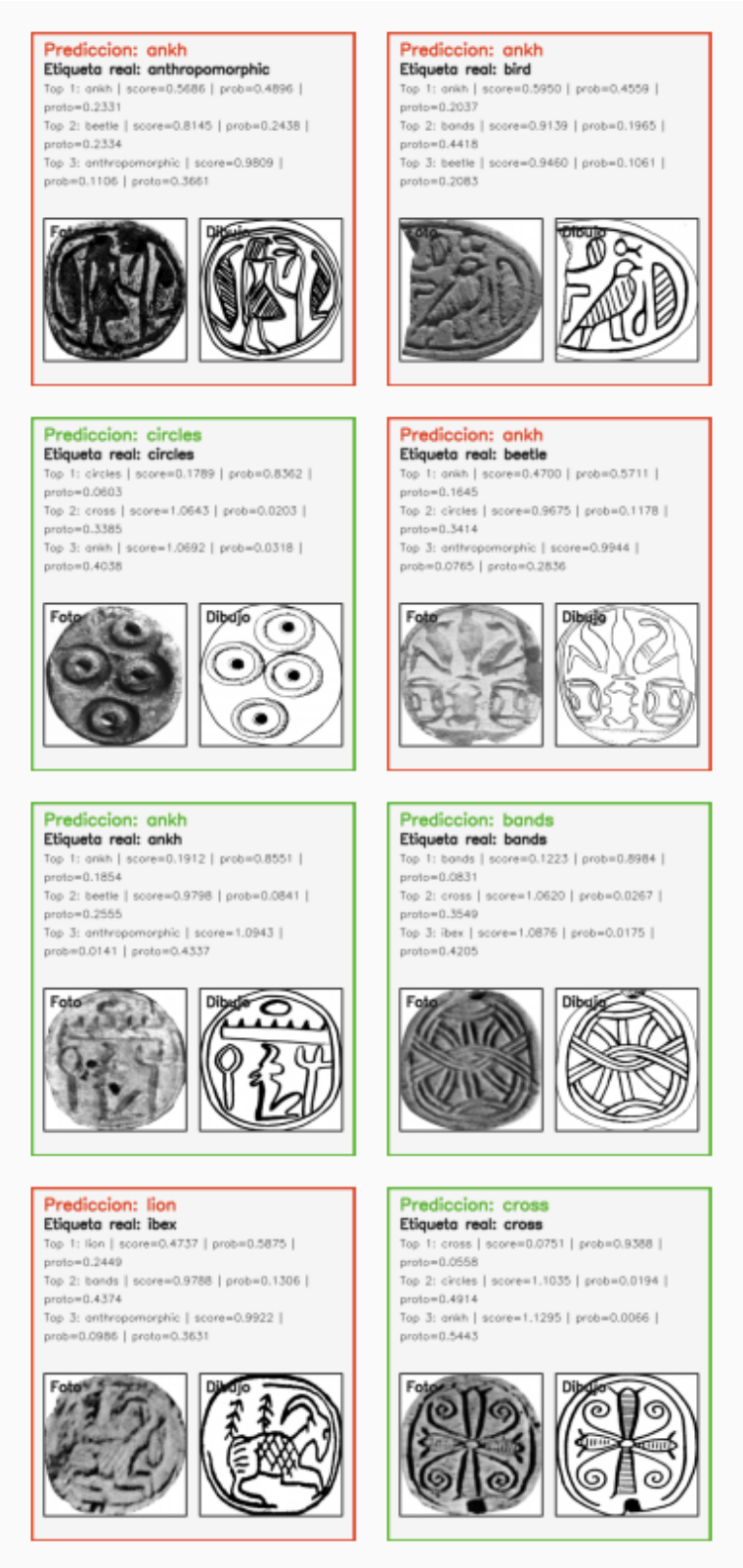



Figura 2.2: Resumen visual de `EfficientNet + MLP`.

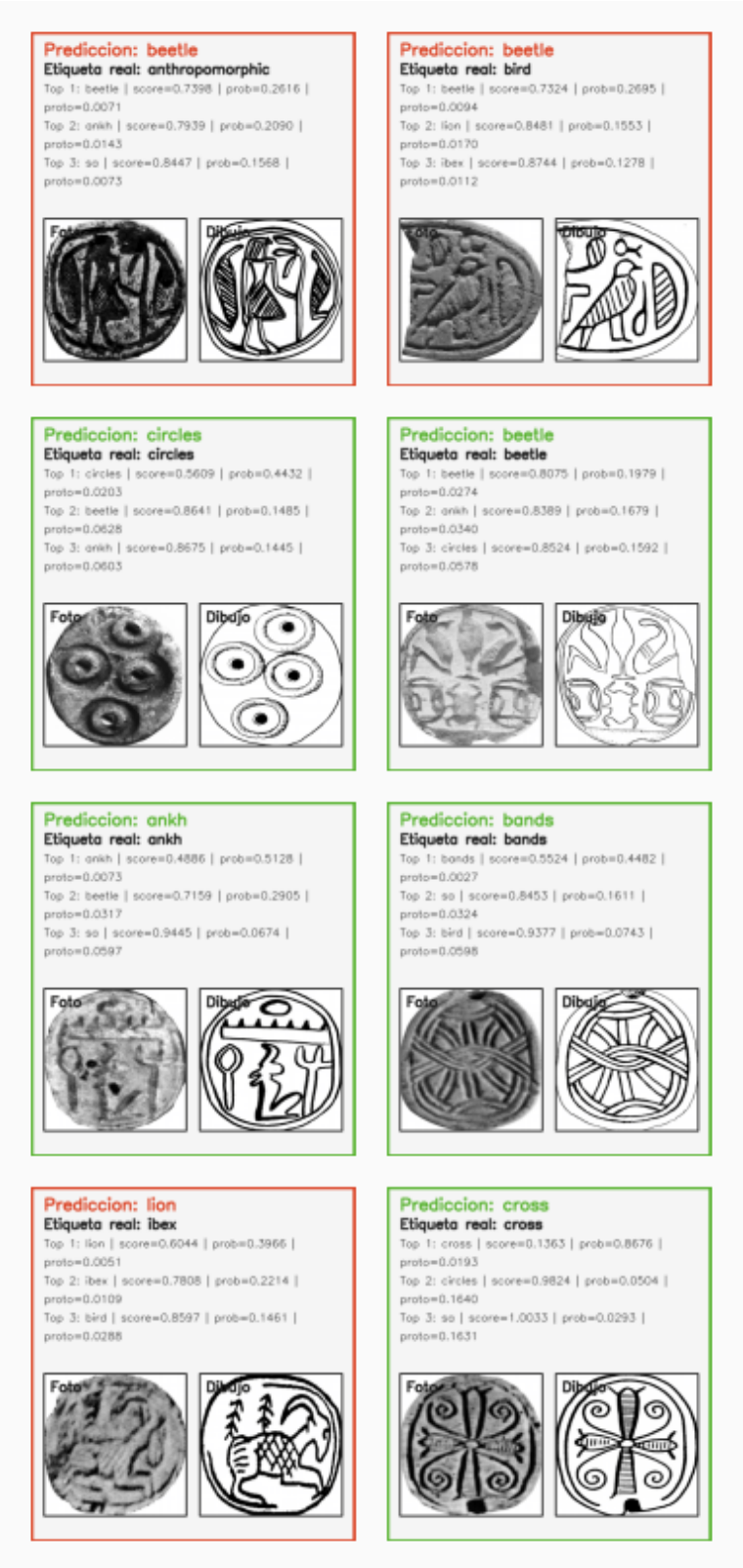



Figura 2.3: Resumen visual de `EfficientNet + KAN`.

Estos mosaicos no sustituyen a la evaluación cuantitativa, pero ayudan a interpretar el tipo de errores. En la selección mostrada, la variante MLP acierta 4 de 8 ejemplos, mientras que la variante KAN acierta 5 de 8. Ambos modelos comparten algunas confusiones semánticas, por ejemplo entre `ibex` y `lion`, lo que sugiere que ciertas formas siguen siendo ambiguas incluso cuando se combinan fotografía y dibujo.

Sin embargo, también se aprecian diferencias cualitativas. La variante KAN corrige al menos uno de los fallos que el modelo MLP cometía en esta misma selección, como el ejemplo cuyo artefacto real es `beetle`. En cambio, ambos siguen fallando en algunos casos donde la similitud geométrica entre clases es alta, especialmente en ejemplos con rasgos compartidos o con información visual menos discriminativa.

## 2.7. Discusión del segundo bloque

La comparación sobre `EfficientNet` cambia la lectura obtenida en el primer bloque. Cuando se parte de un *backbone* potente y preentrenado y solo se modifica el bloque final de clasificación, la cabeza KAN sí muestra una ventaja empírica frente a la MLP. Esto sugiere que la expresividad adicional de KAN puede ser más útil cuando el extractor de características ya proporciona *embeddings* ricos y relativamente estables.

Dicho de otro modo, en el primer bloque la limitación principal parecía estar en la representación. En el segundo bloque, esa representación la aporta `EfficientNet`, de modo que la diferencia entre MLP y KAN se vuelve más visible. En este contexto, la KAN parece aprovechar mejor el espacio de *embeddings* fusionado foto+dibujo y produce una frontera de decisión más adecuada para el problema.

## 2.8. Conclusión general

Los dos bloques del trabajo llevan a una conclusión matizada. En modelos ligeros, especialmente cuando interviene un VAE y el volumen de datos es reducido, la KAN no ha mostrado una mejora consistente sobre la MLP. Sin embargo, al pasar a un escenario de *transfer learning* con `EfficientNet-B0`, la situación cambia: manteniendo fijo el *backbone* y variando solo la cabeza final, la variante KAN obtiene mejores resultados que la variante MLP tanto en accuracy directa como en la métrica híbrida.

En conjunto, esto sugiere que las capas KAN no deben evaluarse aisladamente, sino en función de la calidad de la representación sobre la que operan. En este proyecto, su ventaja aparece con mayor claridad cuando trabajan sobre *embeddings* de alto nivel extraídos por un *backbone* preentrenado. Por ello, la comparación `EfficientNet + MLP/KAN` constituye un resultado más favorable para KAN que el observado en el bloque inicial de arquitecturas ligeras.