Reporte: Detección del Estado de Ánimo en Gatos mediante Imágenes (Gatección)

1. Objetivos del Proyecto

El objetivo principal de este proyecto es **desarrollar un sistema automático capaz de detectar el estado de ánimo de los gatos a partir de imágenes**, utilizando técnicas modernas de visión por computadora y aprendizaje profundo. Esto permitirá:

- **Mejorar el bienestar animal**: Identificar emociones negativas o de alerta puede ayudar a prevenir problemas de salud o estrés en los gatos.
- **Facilitar el monitoreo remoto**: Permite a dueños y veterinarios monitorear el estado emocional de los gatos sin intervención directa.
- **Aportar a la etología aplicada**: Generar herramientas accesibles para el estudio del comportamiento felino.

2. Problemas que Resuelve

- **Interpretación objetiva de emociones**: Los gatos no expresan emociones de forma tan evidente como otros animales; su lenguaje corporal y facial es sutil. Este sistema traduce esas señales en datos cuantificables que nos permitirán entender a nuestro michi.
- **Prevención de problemas de salud**: Detectar estados de molestia o alerta puede ser indicio temprano de enfermedad o incomodidad.
- **Automatización**: Reduce la necesidad de observación manual constante, ahorrando tiempo y recursos o incluso permitiéndonos identificar estados que no habíamos notado en el gato.
- **Mejor comunicación humano-animal**: Ayuda a los humanos a entender mejor a sus mascotas, fomentando una convivencia más armoniosa.

3. Procedimiento y Justificación de Implementación

a) Selección de Datos

Se utilizaron bases de datos públicas con imágenes de gatos, a las que se les realizó un **etiquetado manual** según criterios etológicos (posición de orejas, ojos, bigotes, tensión corporal). Esto permite tener un conjunto de entrenamiento específico para emociones, no solo para identificación de especie o raza.

b) Arquitectura del Modelo

Se implementó una **Red Neuronal Convolucional (CNN)** personalizada, llamada *CatMoodNET*, debido a la eficacia de las CNN para tareas de clasificación de imágenes.

- Dos bloques convolucionales con *ReLU* y *MaxPooling* para extraer características visuales relevantes.
- Tres capas densas (fully connected) para la toma de decisión final.
- Salida con 4 clases (emociones).

c) Proceso de Entrenamiento

- **Preprocesamiento y augmentación**: Redimensionado a 64x64, normalización, y augmentación con flips horizontales para mejorar la generalización.
- **Entrenamiento y validación**: Separación de datos para evitar sobreajuste y evaluar el desempeño real del modelo.
- **Optimización**: Uso de *Adam* como optimizador y *CrossEntropyLoss* como función de pérdida.

d) Evaluación y Uso

Se evaluó la precisión en entrenamiento y validación, y se implementaron funciones para predecir emociones tanto en imágenes del set como en imágenes nuevas. Se guardó el modelo entrenado para futuras predicciones sin necesidad de reentrenar.

4. Bibliotecas Utilizadas y su Rol en la Optimización

- **os** y **glob**: Para navegar por el sistema de archivos, listar rutas de imágenes y construir rutas de forma portable.
- **torch** y **torch.nn**, **torch.nn.functional**: Núcleo de PyTorch para definir modelos, capas convolucionales, funciones de activación y operaciones tensoriales.

- **torch.optim**: Proporciona optimizadores (Adam, SGD) y herramientas para ajustar tasas de aprendizaje durante el entrenamiento.
- **torch.utils.data.Dataset** y **DataLoader**: Para crear datasets personalizados y cargar datos en batches de forma eficiente, con soporte de múltiples procesos.
- **torchvision.transforms**: Para pipeline de preprocesado de imágenes (resize, flip, rotaciones, normalización) y data augmentation.
- **PIL (Python Imaging Library)**: Carga y conversión de imágenes a formato RGB compatible con PyTorch.
- **scikit-learn (train_test_split, precision_score)**:
  - `train_test_split`: División estratificada de los datos en entrenamiento y validación, garantizando representación de todas las clases.
  - `precision_score`: Cálculo de métricas por clase para evaluar la precisión individual de cada emoción.
- **collections.Counter**: Para contabilizar muestras por etiqueta y detectar desequilibrios de clases.
- **random**, **numpy** (`np`): Selección aleatoria de ejemplos para visualización y manejo de arreglos numéricos.
- **matplotlib.pyplot**: Visualización de curvas de pérdida, precisión, matrices de confusión e imágenes con sus predicciones.
- **ipywidgets** y **IPython.display.display**: Creación de un widget interactivo para subir imágenes en el notebook y mostrar predicciones en tiempo real.
- **io**: Manejo de buffers de bytes al cargar imágenes desde el widget sin guardarlas en disco.

Optimización lograda:

- *PyTorch* y *torchvision* permiten el uso eficiente de GPU, acelerando el entrenamiento.
- Las transformaciones y augmentaciones ayudan a que el modelo generalice mejor, evitando el sobreajuste en datasets pequeños.
- El uso de *Adam* como optimizador acelera la convergencia respecto a métodos clásicos como SGD.

5. Justificación de las 4 Emociones Seleccionadas

Las emociones seleccionadas no solo son representativas del comportamiento felino, sino que también tienen relevancia práctica para el bienestar y manejo del gato:

1. **Enojado**
   Justificación: Un gato enojado puede estar experimentando dolor, enfermedad o incomodidad ambiental. Detectar este estado ayuda a intervenir a tiempo para mejorar su calidad de vida.

2. **Dormido**
   Justificación: Clasificar si un gato duerme cómodamente permite evaluar su nivel de descanso y bienestar. Un sueño profundo y cómodo es signo de salud y tranquilidad, mientras que dormir en alerta puede indicar estrés.

3. **Neutral**
   Justificación: El estado neutral es el más común y representa tranquilidad y comodidad. En gatos, la neutralidad facial suele ser señal de que están sanos y no presentan emociones negativas ni de alerta. Además, los gatos no suelen mostrar "alegría" como los perros, por lo que esta clase engloba también estados positivos.

4. **Alerta**
   Justificación: Un gato en alerta puede estar percibiendo peligro o sentirse incómodo. Identificar este estado permite a los dueños o cuidadores actuar para mejorar el entorno o reducir factores de estrés.

6. Dificultades Presentadas

Durante la elaboración del proyecto, enfrentamos varios desafíos importantes:

- **Definir las emociones a clasificar**: Tomar la decisión de cómo lucían las emociones de los gatos fue complicado debido a la subjetividad de los criterios etológicos y la falta de consenso en la literatura sobre comportamiento felino.

- **Decidir la cantidad de épocas de entrenamiento**: Encontrar el balance entre un número adecuado de épocas para evitar el sobreajuste y garantizar una buena generalización fue un proceso iterativo que requirió múltiples pruebas.

- **Condensar el conjunto de datos**: La selección y condensación del conjunto de datos obtenido del *Cat dataset* de Kaggle para pruebas, entrenamiento y validación presentó dificultades, especialmente al garantizar que estuviera equilibrado y representara bien las emociones seleccionadas.

7. Conclusiones

- La combinación de técnicas de visión por computadora y aprendizaje profundo nos permitió abordar un problema complejo y poco explorado: la detección automática de emociones en gatos. Lo cuál resulta útil para aquellos que consideramos estos animales como parte de nuestra familia.
- La selección de emociones se basa en criterios etológicos y de bienestar animal, buscando un equilibrio entre lo observable y lo relevante para la salud y manejo de los gatos.
- El uso de bibliotecas modernas y una arquitectura CNN nos facilitó la implementación.