Apartado 2: Partiendo del desarrollo, librerías y tecnologías utilizadas en el apartado 1. Plantear cuales serían los pasos necesarios para entrenar un modelo de detección con categorías no existentes en los modelos preentrenados. Los puntos en los que centrar la explicación son:

1. Pasos necesarios a seguir.
    - En este caso, la librería utilizada es ultralytics que proveé de los modelos YOLO, los cuales a lo largo de las versiones han mejorado mucho en cuanto a tiempo de detección y eficacia y además en la forma de entrenarlos.
      Esta librería ofrece entrenamientos con buenos resultados, rápidamente y con una menos cantidad de imágenes (300 como mínimo según la documnetación).
      Pese a ello, es importante tener en cuenta diferentes aspectos:
      - Análisis y estudio de las clases a detectar y su entorno de detección. Por ejemplo, no es lo mismo detección de células cancerígenas en imágenes médicas que detección de personas en imágenes tomadas por una cámara.
      Por ello es importante conocer la naturaleza de la imagen y dependiendo de ella decidir cómo se va a hacer la adquisición de datos (imágenes tomadas por el cliente o web scrapping)
    - La creación de la base de datos es sumamente importante, es necesario que el Dataset esté balanceado y que exista variedad dentro de las imágenes y lo óptimo es tener un set de entrenamiento, uno de validación y otro de test divididos en porcentajes de 80, 10, 10% respectivamente.
    - Un estudio de posibles técnicas de preproceso en base a la naturaleza de las imágenes es importante para conseguir una mejor extracción de características de la imagen y una mejor detección en el futuro.
    - Una vez creada la base de datos y decididas las técnicas de preprocesado se procederían al entrenamiento del modelo. En el caso de YOLO, la librería ofrece una forma rápida de generar una estructura de base de datos estándar para entrenar estos modelos.
      En caso de otros modelos, habría que crear una clase de Python en la que se leyera y se preparan las imágenes con sus respectivas anotaciones, de forma que el modelo las pudiera entender.
    - A continuación, habría que crear un bucle de entrenamiento, donde se extrajera de forma recurrente información de la base de datos y se la hiciera pasar por el modelo y con una función de pérdida minimizar la diferencia entre las predicciones y los resultados originales.
    - A lo largo de los experimentos que realizarían para obtener un buen entrenamiento, lo ideal es hacer un estudio de los hyperparámetros del modelo y optimizar el proceso de mejora de estos HP con técnicas como Grid Search, Random Search, o Baby-sitting. Personalmente, mi opción para entrenar un modelo desde 0 es hacer bloques de experimentos con random search, habiendo definido unos rangos de selección en los HP y poco a poco con Babysitting ir reduciendo los rangos hasta encontrar la combinación óptima.
    - Como último paso, es importante parametrizar los resultados, por ejemplo: La precisión de las detecciones del modelo, la minimización del modelo y latasa de acierto de la clase predicha (accuracy).

2. Descripción de posibles problemas que puedan surgir y medidas para reducir el riesgo.
    - En caso de que el modelo no consiguiera llegar a unos resultados óptimos de detección y de generalización, habría que estudiar las gráficas y decidir diferentes técnicas de mejora. Por ejemplo: el caso más común es que los modelos tiendan al Overfitting, por lo cual no son capaces de generalizar correctamente. En este caso estudiaría mejores técnicas de extracción de características de las imágenes durante el preprocesado de las mismas y también técnicas de aumentado de la base de datos (transformaciones de la imagen etc)
    - También sería eficaz incrementar el tamaño de la base de datos y revisar si las imágenes realmente son variadas en el interior.
    - Finalmente estudiaría la profundidad del modelo y sobre todo de las características de las capas del clasificador, ya que se podrían ajustar mejor las cantidades de canales extraídos en esas capas así como los términos de regularización que pueden facilitar una mejor convergencia del modelo.
3. Estimación de cantidad de datos necesarios, así como de los resultados, métricas, esperadas.
    - Dependiendo de si el modelo está preentrenado o no la cantidad de imágenes necesaria es muy diferente. En el caso de los preentrenados una menor cantidad de imágenes es necesaria, con una base de datos bien creada con unas 500-1000 imágenes serían necesarias para unos buenos resultados. En el caso de los entrenados desde scratch lo ideal serían bases de datos mucho mayores. Pero siempre es posible preentrenar los modelos con bases de datos open source y luego hacer un finetuning con una base de datos personal lo cual reduciría mucho la cantidad de imágenes, similar al caso anterior dependiendo del tipo de objeto a detectar.
    - En cuanto a las métricas, se espera una minimización de la función de perdida estable y suave, sin picos. Las métricas utilizadas más comunes en detección de objetos son IoU y MAP con diferentes thresholds.
4. Enumeración y pequeña descripción (2-3 frases) de técnicas que se pueden utilizar para mejorar el desempeño, las métricas del modelo en tiempo de entrenamiento y las métricas del modelo en tiempo de inferencia.
    - En el caso de reducir los tiempos de inferencia, utilizaría técnicas de compresión del modelo, como por ejemplo convertirlo a ONNX.
    - Para reducir los tiempos durante el entrenamiento utilizaría técnicas de transfer learning como: escoger modelos preentrenados en detección de objetos y hacer un finetunning. Otra opción es coger los pesos de una red entrenada en una tarea similar e introducirlos en tu modelo, lo que hace un precalentamiento del modelo y ayuda a una convergencia más rápida.

Apartado 3 (Opcional): Exponer como se adaptaría el caso 2 a un procesamiento en el borde. Los puntos en los que centrar la explicación son:

1. Pasos necesarios a seguir.
    - Es importante conocer el tipo de hardware con el que se va a contar y dependiendo de la cantidad de memoria y el procesador etc escoger la mejor técnica.
    - En este caso, partiendo de un modelo ya entrenado estudiaría técnicas de compresión del modelo, como ONNX (ya comentado anteriormente), entrenamientos con Quantization o incluso técnicas de transferencia de

conocimiento a modelos más pequeños con Distillation.

2. Descripción de posibles problemas que puedan surgir y medidas para reducir el riesgo.
   - Pueden surgir problemas como que el modelo no reduzca su memoria lo suficiente y no pueda ser ejecutado dentro del hardware, pero para ello es importante el estudio previo de donde se va a ejecutar el modelo.
   - En técnicas como distillation y quantization es muy importante ejecutarlas correctamente ya que Quantization afecta dirécamente a las distribuciones de los pesos y puede dañarlas y Distillation afecta a la forma de minimización de la función de pérdida y una mala decisión en la fórmula con la que ejecutar la técnica puede hacer que el nuevo modelo no aprenda correctamente.
3. Estimación de cantidad de datos necesarios así como de los resultados, métricas, esperadas.
   - En este caso, optaría por entrenar modelos grandes capaces de obtener muy buenos resultados y aplicar técnicas de transfer learning a modelos inferiores. Esto facilita mucho el entrenamiento de modelos menores.
4. Enumeración y pequeña descripción (2-3 frases) de técnicas que se pueden utilizar para mejorar el desempeño, las métricas del modelo en tiempo de entrenamiento y las métricas y tiempo de ejecución en tiempo de inferencia.
   - Como se ha mencionado anteriormente intentaría estudiar compresiones del modelo como Onnx, TensorRT, OpenVino, Caffe, etc… dependiendo de los requisitos del hardware también.