Entrenar un modelo de detección con categorías no existentes en los modelos preentrenados

Entrenar un modelo de detección de categorías que no existen en los modelos pre-entrenados requeriría primero de disponer de datos etiquetados para poder entrenar al modelo, pero vamos paso por paso tal y como yo lo plantearía.

Pasos necesarios

Definición del objetivo

Es importante definir de forma clara cuáles son las nuevas categorías que debe detectar el modelo. Debe quedar bien explicado el contexto y el caso de uso, para que así se pueda hacer un estudio preliminar que pueda determinar la posibilidad o imposibilidad de detectar lo que se pide.

No siempre es posible conseguir los resultados deseados entrenando o *finetuneando* un modelo, por lo que entender esto antes de dedicar más esfuerzos a la tarea en concreto es importante. Hay que gestionar las expectativas antes de encontrar problemas a largo plazo que pudieran haber sido detectados en esta fase desde el principio.

Recolección de datos.

Los datos siguen siendo el corazón de la Inteligencia Artificial, sea generativa o no. Al igual que es importante definir bien los objetivos, también es de vital importancia definir los datos.

Se debe tener en cuenta la accesibilidad de los datos deseados y el volumen disponible existente. No es lo mismo contar con centenas de imágenes que contar con millones. El nuevo conjunto de datos debe ser representativo y amplio, de buena calidad y variado, y lo más cercano posible a su uso final, para que así el modelo final tenga buenos resultados.

Siempre es bueno intentar contactar con el cliente o usuario final para hablar de posibles problemas que ellos puedan encontrar con las imágenes con las que hipotéticamente fuéramos a trabajar.

Preparación de los datos.

Los datos deben estar ya etiquetados o han de ser etiquetados manualmente. Como estamos hablando de imágenes, necesitaremos herramientas de anotación especializadas que permitan usar una interfaz para resaltar los puntos a clasificar dentro de cada imagen. Por supuesto, estos tipos de datos tienen su formato especializado.

En mi experiencia personal he trabajado con el formato COCO y con herramientas de anotación de datos como PAWLS, que te permiten ir imagen por imagen resaltando los puntos a clasificar.

Configuración del modelo

Elegir una arquitectura de modelo base, como YOLOv8, y ajustar la configuración para incluir las nuevas categorías.

Tendríamos que tocar la configuración del modelo para añadir el nuevo número de clases, así como otros hiperparámetros relevantes.

Entrenamiento del modelo

Se deberá cargar el modelo utilizando el framework deseado y monitorizar las métricas de rendimiento adecuadas. En la página de ultralytics encontramos una amplia explicación de métricas que podríamos utilizar para nuestro modelo.

Evaluación del modelo

Si hemos dividido nuestros datos en *train, test* y *evaluation* podemos proceder a evaluar nuestros datos en el conjunto designado. Esta parte es de vital importancia para ver como funciona nuestro modelo en datos que no debería haber visto durante el entrenamiento.

Problemas posibles y soluciones probables

Es posible que nos enfrentemos a problemas importantes a la hora de entrenar estos modelos.

Un problema por ejemplo podría ser que **no tuviéramos suficientes imágenes** para las nuevas categorías.

Hablando de nuevo de los modelos de YOLO, es el acrónimo de *You Only Look Once* y aseguran un buen rendimiento en escenarios con pocos datos. Por si acaso, yo me aseguraría de tener 1.000 imágenes por cada nueva categoría y, si fuese necesario, utilizar técnicas de data augmentation.

Otro problema que podríamos encontrar son las **anotaciones incorrectas**, las cuales podrían degradar significativamente el rendimiento del modelo. Para afrontar este problema

tendríamos que realizar revisiones de calidad de las anotaciones o utilizar herramientas de etiquetado con verificación cruzada.

Podríamos también enfrentar problemas de **desbalance de clases**. Si algunas categorías están subrepresentadas, el modelo puede no aprender a detectarlas adecuadamente. Utilizar técnicas de balanceo, como por ejemplo sobremuestreo de clases minoritarias o submuestreo de las mayoritarias podría ayudar.

Otro problema es encontrar la **configuración óptima de hiperparámetro**s para conseguir una curva de aprendizaje óptima. Este problema es complicado de analizar y requiere mucha experimentación con el dataset para poder enfrentarlo.

Estimación de datos necesarios y métricas esperadas

- Cantidad de datos:
    - Entrenamiento: Al menos 1.000 imágenes por categoría.
    - Validación y prueba: 200-300 imágenes por categoría cada uno.
- Métricas: Como comentaba no he trabajado mucho en problemas de Computer Vision, pero aplicaría métricas como las del apartado de YOLO.

Técnicas para mejorar el desempeño y las métricas del modelo

1. **Fine tuning:** Utilizar modelos pre entrenados y ajustar con los nuevos datos. Esto permite aprovechar características aprendidas de un gran conjunto de datos y adaptarlas a las nuevas categorías.
2. **Data Augmentation:** Aumentar la diversidad del conjunto de datos. Esto ayuda a mejorar la generalización del modelo.
3. **Regularización:** Aplicar técnicas de regularización como dropout y L2 regularization para evitar el *overfitting*.
4. **Hyperparameter Tuning:** Realizar una búsqueda sistemática de los mejores hiperparámetros (learning rate, batch size, etc.) utilizando métodos como grid search o random search.
5. **Ensemble Methods:** Combinar las predicciones de múltiples modelos para mejorar la robustez y la precisión del sistema de detección.
6. **Optimización para la inferencia:** Utilizar técnicas como la cuantización y el pruning para reducir el tamaño del modelo y aumentar la velocidad de inferencia sin sacrificar significativamente la precisión. Esto es especialmente útil para desplegar el modelo en dispositivos con recursos limitados aunque suele conllevar pérdida de rendimiento.