Resumen ejecutivo - Proyecto final Aprendizaje automático 4
Detección de personas y mascarillas faciales usando redes CNN

El proyecto se estructura de manera progresiva y sistemática en torno a un backbone convolucional personalizado y cabezas de decisión adaptadas a las tareas de regresión, para detección de rostro de personas, y de clasificación, para identificar si están usando mascarillas faciales o no. Además, compara el impacto de diferentes técnicas de data augmentation, se exploran aproximaciones de transfer learning con modelos pre-entrenados y se realiza un ajuste de hiperparámetros para mejorar el rendimiento y estabilidad de las redes.

En primer lugar, mencionaremos características puntuales del enfoque utilizado para el entrenamiento y validación de los modelos:

1. Metodología de entrenamiento y optimización

- Ajuste por lotes: Ciclos de entrenamiento por épocas y lotes de imágenes.
- Optimizador AdamW: Ajustado con weight decay
- Scheduler *CosineAnnealingLR*: Reducción de learning rate progresiva.
- Recorte de gradientes *clip_grad_norm_*: Evitar explosiones de gradientes.
- Loss robusto de regresión: SmoothL1Loss con parámetro β ajustable + Distance IoU
- Loss combinada: Clasificación (Cross_entropy) + Regresión con pesos ponderados.
- Early stopping: Paciencia entre 5 a 10 épocas.
- Regularización: Weight decay, dropout, batch normalization, augmentations, etc.
- Guardado de mejor modelo: Checkpoint basado en métrica de monitoreo.

2. Metodología de evaluación, validación y monitoreo

- Métricas: Loss, IoU (regresión), Accuracy (clasificación), Score combinado.
- Validación cruzada: Split 80/20 con seed fijo para reproducibilidad.
- Monitoreo: Gráficas de pérdida y métricas por época.
- Sanity checks: Validación en batches para detectar problemas.
- Visualización: Predicciones sobre imágenes de validación con bounding boxes.

A continuación, presentaremos más detalles de las acciones realizadas para llegar a las conclusiones y entregables del proyecto:

1. Set-up de clases y funciones

- Se definió una clase MaskDataset, la cual carga un directorio de imagenes, re ajusta a un tamaño deseado y las normaliza con media y desviacion propios del dataset, o de ImageNet. Ademas, ajusta los cuadros delimitadores y los escala de 0 a 1.
- De igual manera, se definió una clase AugmentedDataset la cual toma un Dataset de pytorch y le realiza aumentaciones y transformaciones especificadas por el usuario.
- Se define una clase Trainer, la cual, gestiona el entrenamiento y validación de un modelo de CNN de 2 cabezas, calculando pérdidas y métricas (loss, accuracy, IoU),

aplicando optimización y scheduler, controlando early stopping y guardando los mejores checkpoints.
- Se definen diferentes arquitecturas de extractores de características (backbone) y cabezas de decisión para ser probadas y evaluadas. Estas arquitecturas se crean como clases dinámicas, las cuales ajustan la arquitectura de la red en base a los parámetros de la clase.
- Se desarrollaron toda clase de funciones para distintos objetivos, como mostrar las imágenes originales con sus cuadros delimitadores y etiquetas, funciones para evaluar el rendimiento y predicciones de un modelo entrenado, funciones para realizar gráficas útiles, entre otros.

2. Pruebas de arquitectura de la red

- Partimos de una arquitectura base, con un backbone sencillo y una cabeza de decisión poco profunda. Este modelo es nuestro baseline.
- Probamos en total 3 tipos de backbone, uno ligero de 2 capas convolucionales, uno “base” y uno robusto con más de 5 capas convolucionales, pooling adaptativo y todas las buenas prácticas como normalizar los net-inputs y bloques dropout.
- Probamos también, 3 tipos de cabezas de decisión, una “base”, y 2 enfoques más robustos, uno con ambas cabezas con 3 capas fully connected, y otro con más capas convoluciones de una dimensión para la cabeza de regresión.

3. Optimización de hiperparametros

- La metodología de tuneo de hiperparametros se lleva a cabo con el paquete Optuna, dividiendo el tuneo en la optimización de hiperparametros de entrenamiento y tuneo de hiperparametros de arquitectura. El metodo utilizado es una optimización multi-objetivo (maximizar IoU y Accuracy simultáneamente) en un numero fijo de intentos (entre 30 a 50 trials)
- Ambos estudios devuelven un frente de Pareto con configuraciones que ofrecen diferentes trade-offs entre IoU y ACC. De esta manera se escogen los mejores hiperparametros con la mejor arquitectura hallada anteriormente.

4. Técnicas de aumento de datos (Data augmentation)

- Haciendo uso de nuestro wrapper externo (Intersection over Union), evaluamos 3 enfoques de transformaciones y aumento de datos utilizando el mejor modelo hallado hasta el momento:
    - Light: solo HorizontalFlip.
    - Medium: flip, rotación leve, brillo/contraste.
    - Strong: flip, rotación, brillo/contraste, desenfoque y saturación.

5. Pruebas con Transfer Learning

- Se integraron 3 backbones pre-entrenados diferentes (Resnet18, VGG16, Resnet50) reemplazandolos en nuestra arquitectura y realizamos pruebas con nuestras 2 cabezas de decisión robustas.

- Para realizar el ajuste de los modelos con backbones pre-entrenados, descongelamos unicamente las 2 últimas capas del backbone y realizamos un fine-tunning con nuestro dataset.
- Al modelo con mejores resultados, le aplicamos un tuneo de hiperparametros con optuna para hallar el modelo con el mejor rendimiento posible.

A continuación, mencionaremos nuestras **conclusiones finales**:

- La tarea de **clasificación binaria** (uso o no de tapabocas) mostró un desempeño consistente incluso con arquitecturas de baja complejidad, lo que evidencia que el problema puede resolverse de manera efectiva con representaciones latentes de dimensión moderada.
- En contraste, la **regresión de bounding boxes** resultó ser el componente más desafiante, requiriendo arquitecturas más profundas y la incorporación de funciones de pérdida especializadas (SmoothL1, IoU) para mejorar la precisión de localización.
- El **data augmentation** ligero y medio contribuyó positivamente a la capacidad de generalización del modelo, mientras que las transformaciones intensivas tendieron a degradar la calidad de las predicciones de las cajas, reflejando un balance crítico entre robustez y estabilidad.
- El **ajuste de hiperparámetros mediante Optuna** confirmó la relevancia de factores como learning rate, weight decay, box_loss_w, el tamaño de imagen y parámetros arquitecturales en el desempeño global, mostrando que la optimización sistemática es determinante en el rendimiento final.
- El **transfer learning** con modelos preentrenados mejoró de manera consistente las métricas de evaluación respecto a los backbones desarrollados desde cero. Entre las alternativas probadas, **ResNet18** alcanzó los mejores resultados tanto en IoU como en precisión de clasificación, además de acelerar la convergencia del entrenamiento.
- La limitada disponibilidad de imágenes y la reducida variabilidad en condiciones de iluminación, ángulos de captura y densidad de personas constituyen factores críticos que pueden comprometer la capacidad de generalización del modelo en escenarios reales más complejos.
- En conjunto, el enfoque desarrollado se consolida como una **pipeline modular y flexible**, capaz de integrar distintas estrategias de preprocesamiento, augmentation, backbones y cabezas de decisión, lo que garantiza una base sólida para la mejora continua del sistema de detección y clasificación de tapabocas.