Tecnológico de Monterrey

Control de Ingresos y Egresos de Vehículos en Patios de Distribución mediante Redes Neuronales Convolucionales y RFID en Raspberry Pi con Unidad Hailo-8L

Avance 3 Baseline

Profesora Titular: Ing. Grettel Barceló Alonso, PhD

Equipo #20

- A01794188 - Francisco Xavier Bastidas Moreno
- A01794653 - Raúl Jesús Coronado Aguirre
- A01794327 - Juan Sebastián Ortega Briones

13 de octubre de 2024

¿Qué algoritmo (individual, NO ensambles) se puede utilizar como baseline para predecir las variables objetivo? 3
¿Se puede determinar la importancia de las características para el modelo generado? 3
¿El modelo está sub/sobreajustando los datos de entrenamiento? 4
¿Cuál es la métrica adecuada para este problema de negocio? 5
¿Cuál debería ser el desempeño mínimo para obtener? 5
Conclusión: 6

¿Qué algoritmo (individual, NO ensambles) se puede utilizar como baseline para predecir las variables objetivo?

Hemos decidido utilizar el algoritmo YOLOv6n como nuestro baseline para este proyecto. Este modelo de detección de objetos es eficiente para la tarea de detectar vehículos en tiempo real usando videos capturados por cámaras de seguridad.

Razonamiento: YOLOv6n nos ofrece la velocidad necesaria para realizar detecciones en tiempo real y su arquitectura simple nos permite establecer una línea base que podremos ajustar posteriormente a medida que avancemos. al igual que esta versión del modelo de visión computacional es compatible con las capacidades de procesamiento de la Raspberry Pi.

Para mejorar el procesamiento, utilizaremos la unidad Hailo-8L, lo que aumentará significativamente la capacidad de procesamiento y nos permitirá hacer ajustes más precisos en el modelo, maximizando su eficiencia en escenarios de tiempo real.

¿Se puede determinar la importancia de las características para el modelo generado?

Si bien YOLOv6n no permite una selección explícita de características en términos tradicionales, las características que el modelo utiliza se basan en propiedades visuales como bordes, texturas y formas, que el modelo aprende automáticamente durante el entrenamiento.

Para mejorar la detección en nuestro caso, consideramos definir una Región de Interés (ROI), que nos permitirá eliminar áreas irrelevantes del video. Esto reducirá la carga de procesamiento y las falsas detecciones, optimizando el rendimiento.

¿El modelo está sub/sobreajustando los datos de entrenamiento?

Hemos observado que, en este punto, nuestro modelo está subajustado. Hemos identificado dos problemas principales:

Omite entradas y salidas: En algunos casos, el modelo no detecta correctamente cuando un coche cruza la línea, en especial si se mueve rápidamente o si hay obstáculos parciales.

Cuenta carros doblemente: Cuando un carro si se pierde temporalmente de la detección de objetos, el modelo tiende a contarlo dos veces, como si fuera un coche nuevo en vez de reconocer que solo dejo de reconocerlos por algunos frames.

Estos problemas sugieren que el modelo no está capturando todas las variaciones necesarias de movimiento de los vehículos, lo que indica la necesidad de ajustes adicionales en el seguimiento de los objetos o más entrenamiento para mejorar su precisión.

Pasos futuros pensados incluyen entrenar un modelo con imágenes de carros tomados en el ángulo y con la resolución exacta de las cámaras a utilizar para las entradas y salidas, ya que el modelo de detección de autos que yolo entrenó muy seguramente fue diseñado con carros en situaciones diferentes y ángulos a un nivel de tierra más bajo que el de nuestro análisis, este entrenamiento se puede sumar al modelo de yolo y brindar mejor rendimiento que no pierda de vista a los carros.

¿Cuál es la métrica adecuada para este problema de negocio?

Una buena métrica en este escenario sería utilizar dos tablas separadas o métricas de precisión para cada clase (entradas y salidas) de la siguiente manera:

Para las salidas:
Comparar cuántas salidas reales hubo contra cuántas salidas se predijeron.
Métricas relevantes: Precisión, Exactitud (qué tan cerca están las salidas predichas de las reales).

Para las entradas:
Comparar cuántas entradas reales hubo contra cuántas entradas se predijeron.
Métricas relevantes: Precisión, Exactitud para las entradas.

Precisión por clase: Se puede calcular la precisión para cada clase por separado, es decir, qué porcentaje de las salidas predichas realmente fueron correctas y lo mismo para las entradas.
Exactitud global: Una métrica adicional sería la exactitud total, que representa el porcentaje de entradas y salidas que fueron predichas correctamente.

¿Cuál debería ser el desempeño mínimo para obtener?

Como equipo, creemos que el desempeño mínimo aceptable para este proyecto debería estar por encima de un 80% de precisión. Esto garantizaría que la mayoría de las entradas y salidas de los vehículos se detecten correctamente, minimizando los errores de conteo doble o falsas detecciones.

Este baseline nos permitirá establecer una referencia clara para evaluar nuestros próximos pasos y mejorar el modelo a medida que avanzamos en el desarrollo.

Conclusión:

Hasta ahora, hemos realizado pruebas con diferentes archivos de video que utilizamos como muestra y observamos que nuestro modelo está subajustado, ya que no está detectando correctamente las entradas y salidas de todos los vehículos y presenta problemas de conteo doble cuando los coches pierden visibilidad temporalmente. Esto nos indica que el modelo no está capturando todas las variaciones necesarias en el comportamiento de los vehículos. Para resolver esto, planeamos mejorar el seguimiento de objetos y ajustar los parámetros del modelo para optimizar su rendimiento. Además, utilizaremos la Raspberry Pi con Hailo-8L para maximizar la capacidad de procesamiento y garantizar que el modelo sea más eficiente en escenarios complejos.

Aunque el modelo YOLOv6n parece estar funcionando muy bien, en algunos cuadros no se realizan detecciones, lo que nos obliga a diseñar un algoritmo complejo para identificar y seguir cada vehículo. En caso de que un vehículo deje de ser detectado, necesitamos guardarlo de alguna manera para que, cuando vuelva a aparecer en un cuadro, podamos intentar identificar si es el mismo vehículo. Esto es crucial para poder determinar la distancia recorrida y la dirección. Sin embargo, si no logramos identificar si es el mismo vehículo, la detección de la dirección se vuelve muy compleja. Por ello, la complejidad del algoritmo juega un papel importante en los resultados obtenidos.

Referencias

Visio.ai (2024) Computer Vision Model Performance Evaluation (Guide 2024) https://viso.ai/computer-vision/model-performance/

Gallagher, J. (2022) What is a Confusion Matrix? https://blog.roboflow.com/what-is-a-confusion-matrix/#:~:text=The%20confusion%20matrix%20is%20an,Negative%2C%20and%20True%20Negative%20boxes.