Modelado I

Adrián Matute, Pablo Martínez, Osvaldo Del Valle, Andres Callealta, Jorge Martínez
Tecnológico de Monterrey Campus Querétaro, México

1. Seleccionar técnica de modelado

En este proyecto, evaluamos dos arquitecturas de deep learning diseñadas para la detección de objetos. Finalmente, seleccionamos YOLOv8 debido a su balance superior entre precisión, velocidad y capacidad de adaptación a diversas condiciones de iluminación, como las presentes en imágenes de día y de noche. Sin embargo, antes de llegar a esta decisión, se consideró Faster R-CNN como otra alternativa que vamos a probar, igualmente reconocida por su alta precisión en tareas de detección de objetos.

1.1 Justificación de Selección

YOLOv8 es una arquitectura moderna que hereda y mejora las características de sus predecesoras, como YOLOv5. Destaca por su:

1. Backbone: Basado en CSPNet (Cross Stage Partial Networks), optimizado para extraer características relevantes y retener detalles en las imágenes.
2. Neck: Utiliza PANet (Path Aggregation Network), optimizado para combinar información de diferentes resoluciones, mejorando la detección de objetos pequeños.
3. Head: Genera las predicciones finales, como coordenadas, clases y puntuación de confianza. Es altamente flexible para adaptarse a tareas específicas, como detección de objetos o segmentación de instancias.
4. Diseño Anchor-Free: A diferencia de arquitecturas basadas en anclas, YOLOv8 elimina esta dependencia, simplificando el modelo y mejorando la detección de objetos pequeños y en múltiples escalas.
5. Data Augmentation Avanzada: Integra técnicas como mosaico de imágenes y recortes aleatorios, ayudando a manejar variaciones de iluminación y posiciones de los objetos.
6. Función de Pérdida Optimizada: Combina pérdidas de posición, clasificación y presencia de objetos para un entrenamiento más estable y eficiente.

Estas características convierten a YOLOv8 en una solución ideal para un entorno donde se requiere velocidad, eficiencia computacional y precisión, como en nuestro caso, donde monitoreamos vacas en filas de espera para el ordeño en condiciones variables.

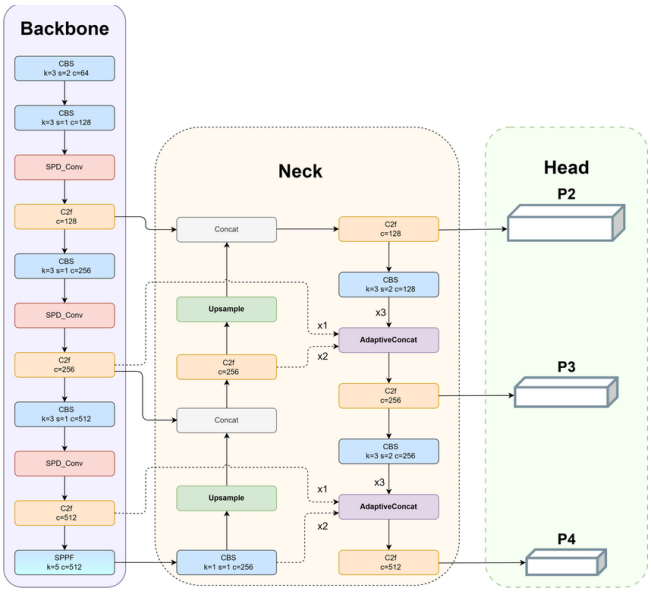

Figura 1 Arquitectura YOLOv8 [1].

1.2 Faster R-CNN: Alternativa Evaluada

Faster R-CNN es una arquitectura ampliamente reconocida por su alta precisión en tareas de detección de objetos. Su enfoque basado en propuestas de regiones (RPN) le permite refinar las predicciones de manera precisa, lo que lo hace ideal para entornos donde la precisión tiene prioridad sobre la velocidad. Sin embargo, al intentar implementarla en este proyecto durante la primera iteración, no logramos completar el ciclo de entrenamiento y evaluación debido a varios problemas fundamentales:

- Requerimientos de Hardware:

  Faster R-CNN demanda una capacidad computacional considerable debido a la complejidad de su arquitectura. El hardware

disponible no era adecuado para procesar los datos de manera eficiente, lo que resultó en tiempos de entrenamiento excesivamente largos e incluso interrupciones durante el proceso.

- Manejo Inadecuado del Dataset:

  El dataset utilizado no estaba completamente preparado para las necesidades específicas del modelo.

  Había una distribución desbalanceada de las clases y la presencia de outliers no fue controlada.

  El etiquetado contenía inconsistencias, lo que redujo la calidad de los datos para el entrenamiento.

  Las imágenes presentaban condiciones mixtas de iluminación, lo que afectó la capacidad del modelo para generalizar, especialmente en imágenes nocturnas o con sombras intensas.

  Debido a las limitaciones de hardware, la falta de preparación del dataset y la configuración inicial inadecuada del modelo, no fue posible completar una iteración de entrenamiento funcional con Faster R-CNN en esta fase. Como resultado, no se generaron métricas ni resultados que pudieran ser analizados y comparados con YOLOv8. Esta falta de resultados marcó un punto crítico para enfocar esfuerzos en resolver estas limitaciones en futuras iteraciones.

Aunque Faster R-CNN ofrece ventajas en términos de precisión en entornos controlados, seleccionamos YOLOv8 debido a:

- Su velocidad, que permite detecciones en tiempos recurrentes.
- Su eficiencia en hardware limitado
- Su capacidad de manejar condiciones variadas de iluminación y densidad de objetos.

Planes para abordar estos problemas en futuras iteraciones

En iteraciones futuras, planeamos retomar el uso de Faster R-CNN con una preparación más exhaustiva para maximizar su potencial. Esto incluye:

Preparación del Dataset:

- Identificación y eliminación de outliers y datos irrelevantes.
- Corrección de etiquetas incompletas o incorrectas para garantizar un etiquetado preciso.

Optimización de la Configuración del Modelo:

- Ajuste de hiper parámetros clave, como el tamaño del batch, la tasa de aprendizaje y número de épocas , para mejorar la estabilidad del entrenamiento.

Recursos Computacionales:

- Utilización de hardware más potente, como GPUs especializadas o instancias en la nube, para permitir un entrenamiento más eficiente y evitar interrupciones.

2. Supuestos de Modelado

- *Calidad del Dataset:*
  - Las imágenes están correctamente etiquetadas y tienen resolución suficiente para identificar vacas.
  - Las condiciones de iluminación representan adecuadamente las condiciones diurnas y nocturnas.
- *Formato del Dataset:*
  - El dataset incluye imágenes con variaciones de densidad (vacas muy pegadas una de otra y en movimiento) para garantizar la robustez del modelo.
- *Supuestos de Modelos:*
  - Los modelos están pre-entrenados en datasets generales como COCO, lo que asume que las características básicas como formas y bordes están bien representadas.
  - El modelo YOLOv8 se beneficiara de data augmentation para mejorar su

capacidad de generalización

3. Diseño de pruebas

La detección de objetos por medio de múltiples bounding boxes presenta 3 resultados que utilizaremos para medir la precisión de la detección.

- Las cajas predichas, coordenadas del centro de la caja, ancho y alto.
- Las clases a las que pertenece cada objeto para dichas cajas, en nuestro caso es únicamente *vaca.*
- La confianza que tiene el modelo de que la clase predicha.

3.1 Métricas de evaluación

Para evaluar la calidad y validez del modelo en la tarea de detección y conteo de vacas en la fila de espera, se utilizaron las siguientes métricas clave:

1. ***Precisión:***
    a. La proporción de verdaderos positivos entre todas las predicciones de clases, nos muestra que tan propenso es el modelo a generar falsos positivos.
    b. **Resultado esperado**: Superior a 85% para garantizar un conteo confiable en imágenes diurnas y nocturnas.
2. ***Recall (Sensibilidad):***
    a. La proporción de verdaderos positivos sobre los positivos esperados, mostrando la habilidad del modelo de predecir todos los objetos presentados.
    b. **Resultado esperado**: Superior al 90%, para minimizar vacas no detectadas.
3. ***mAP@50 y mAP@50-95:***
    a. mAP@50: Precisión media para un umbral de intersección sobre unión (IoU) del 50%.
    b. mAP@50-95: Precisión media en múltiples umbrales de IoU (50%-95%).
    c. **Resultado esperado**:
        i. mAP@50: Superior al 95% en condiciones diurnas.
        ii. mAP@50-95: Superior al 80% para condiciones mixtas.

Estas métricas permiten calcular un porcentaje de error y observar cómo el modelo se ajusta a los objetivos planteados, tanto en términos de detección como de conteo.

3.2 División del Dataset

El dataset total de 8,087 imágenes fue dividido cuidadosamente para asegurar una representación balanceada de condiciones diurnas y nocturnas:

1. Porcentaje de división:
    - 80% para entrenamiento: El modelo ajusta sus parámetros para mejorar la detección de vacas.
    - 20% para validación: Evalúa el rendimiento del modelo en imágenes no vistas previamente.

3.3 Plan de entrenamiento, pruebas y evaluación

3.3.1 Entrenamiento:

- Modelo base: Se utilizó Transfer Learning con un modelo pre entrenado en el dataset COCO, para aprovechar las características generales aprendidas.
- Épocas: Se realizaron hasta 100 iteraciones en los modelos iniciales para evaluar la convergencia.
- Métricas monitoreadas:
    - Pérdida de las cajas (bounding boxes).
    - Pérdida de clasificación.
    - Distribución de Focal Loss (DFL).

3.3.2 Validación y pruebas:

- El conjunto de validación fue utilizado para calcular las métricas de precisión, recall y mAP en imágenes no vistas previamente.
- Se generaron gráficos de pérdida y métricas para observar la estabilidad y el rendimiento del modelo durante las épocas de entrenamiento.

3.3.3 Optimización y ajustes:

- Tasa de aprendizaje: Ajustada iterativamente para garantizar una convergencia eficiente.
- Tamaño de lote: Modificado según los recursos de hardware disponibles.

- Parada temprana: Implementada para detener el entrenamiento si no se observaron mejoras significativas en las métricas de validación.

3.3.4 Resultados Esperados y Actividades de Prueba

1. Conteo de vacas en la fila de espera:
    - Prueba: Evaluar la precisión del modelo para identificar la cantidad exacta de vacas en imágenes.
    - Condiciones: Considerar situaciones con vacas solapadas o en movimiento.
    - Métricas clave: Precisión y recall superiores al 85%.

2. Análisis de aglomeraciones para identificar cuellos de botella:
    - Prueba: Detectar patrones de aglomeración en secuencias de imágenes y evaluar cambios en la densidad de vacas en el área de espera.
    - Evaluación: Se utilizaron métricas de recall en series temporales para determinar momentos de alta congestión.

3.3.4 Justificación del Diseño de Test

El diseño de test asegura que las métricas elegidas sean representativas de los objetivos planteados:

- Precisión y Recall: Directamente alineadas con la capacidad del modelo para cumplir con el conteo de vacas.
- mAP: Garantiza que las detecciones sean consistentes a través de múltiples umbrales, lo que es crucial para minimizar errores en el análisis de aglomeraciones.

Estas pruebas validan que el modelo cumple con los requisitos técnicos y de negocio, maximizando su capacidad para detectar patrones útiles para la toma de decisiones en el sistema de ordeño.

4. Ajuste de parámetros

En nuestro proyecto modificaremos dos tipos de grupos de parámetros para cada modelo que generemos, las cuales son:

- Configuración de entrenamiento
    - Épocas: número de iteraciones que entrenará nuestro modelo.
    - Tamaño de lote: Número de imágenes por cada lote.
    - Tasa de aprendizaje: Controla el tamaño del paso de cada actualización de optimización.
    - Optimizadores: SGD o Adam, entre otros.
- Funciones avanzadas
    - Parada temprana: Para el entrenamiento si no hay mejora en un límite de épocas.
    - Weight Decay: Es una técnica de regularización para evitar el overfitting.
    - Scheduler: Es una técnica para que la tasa de aprendizaje se ajuste durante el entrenamiento.

Esta primera prueba del modelo, se entrenó con los valores por default del Yolov8n.

Parámetro	**Valor**
Épocas	100
Lote	16
Weight Decay	0.0005
Taza de Aprendizaje	0.01

5. Descripción de Modelo

El primer modelo se entrenó con el 80% del dataset (6,470 imágenes), contemplando las dos condiciones (noche y día) en la que trabajará este primer modelo, el 20% restante se usó para validación (1,617 imágenes).

- El modelo tiene una arquitectura eficiente para detección rápida de objetos.
- Generaliza bien en condiciones de luz diurna y nocturna.
- Presenta robustez en detecciones de vacas en movimiento o pegadas una de otra.

6. Evaluación del Modelo

6.1 Resultados de métricas

Métrica	**Valor**
Precisión	95%
Recall	95%

mAP50	96%
mAP@50-95	84%

6.1.1 Limitaciones

- El modelo llega a encadenar diferentes partes de la vaca, y esto genera que cada parte encajonada represente una vaca.
- El modelo algunas veces predice que el ser humano es una vaca, y en la noche hay momentos que no puede identificar si es una vaca.
- El modelo en se equivoca en un 16% en predecir una caja adecuada para un objeto identificado como vaca en un rango de 50% a 95% de confianza.

6.1.2 Dificultades

- **Tiempo de entrenamiento:** El modelo se tardó en entrenar 1.57 hr por 100 épocas.
- **Capacidad de Hardware:** El uso de servidores como Colab o Kaggle en sus planes sin costo fue demasiado lento y poco confiable para el entrenamiento. Esto nos llevó a tener que correr el entrenamiento de manera local.

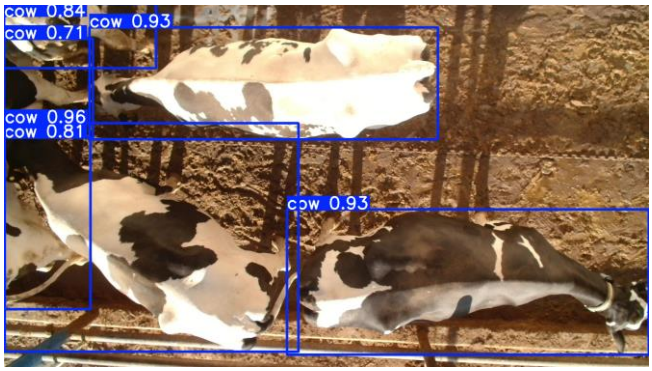

Figura 2 Resultados de la detección.

6.2 Desempeño del Modelo Híbrido

El modelo híbrido YOLOv8 logró un balance excelente entre precisión y calidad con condiciones mixtas:

- Precisión: 95.14%
- Recall: 95%
- mAP@50: 96%
- mAP@50-95: 84%

Ventajas:

- Capacidad de generalización en entornos mixtos (dia/noche).
- Robustez en la detección de vacas en movimiento o muy pegadas una de otra.

6.4 Impacto en el Negocio:

1. **Conteo de Vacas:**
   - Precisión del 95% asegura un monitoreo confiable.
2. **Aglomeraciones:**
   - Recall del 95% permite identificar con precisión patrones de densidad y flujo.

6.5 Conclusión de la Evaluación

El modelo YOLOv8 ha demostrado ser una solución eficiente y robusta para el conteo de vacas y análisis de aglomeraciones en condiciones diurnas y nocturnas. Su capacidad para generalizar en entornos mixtos y su precisión del 95% lo convierten en una herramienta valiosa para optimizar los procesos en el sistema de ordeño.

7. Referencias

[1] Small object detection based on YOLOv8 in UAV perspective disponible en: https://www.researchgate.net/figure/Improving-the-network-structure-of-yolov8_fig1_383213316