Segmentación semántica de imágenes en tiempo real para la conducción autónoma

Antonyjuan Miranda Yquira
Departamento de Ciencias de la Computación
Universidad Católica San Pablo
Arequipa, Perú
antonyjuan.miranda@ucsp.edu.pe

***Abstract*—La segmentación semántica de imágenes es una tarea de gran importancia para sistemas de conducción autónoma. Esta tarea es desafiante debido a la exigencia tanto de su precisión como de su velocidad. Sin embargo, muchos enfoques que buscan mejorar la velocidad de inferencia sacrifican la precisión de sus predicciones debido a que no fueron diseñados específicamente para esta tarea. El enfoque de dos ramas busca lidiar con esta descompensación dividiendo la extracción de características espaciales y semánticas entre sus ramas, logrando un gran balance entre velocidad y precisión. Por ese motivo, este trabajo busca mostrar el funcionamiento de dos modelos basados en el enfoque de dos ramas que obtengan las mejores velocidades, como son BiSeNet V2 y Fast-SCNN. Así mismo se busca analizar cuáles son las diferencias entre estos modelos para poder hacer una comparación entre sus estrategias y se realiza la implementación de ambos buscando comprobar cuál de los dos modelos funciona mejor. Se muestran los resultados originales de ambos modelos en donde BiSeNet V2 logra una mejor velocidad con 156 FPS y una mejor precisión con 72.6% mIoU en comparación de Fast-SCNN con una velocidad de 123.5 FPS y una precisión de 68.0% mIoU. Esto debido a ciertas técnicas que BiSeNet V2 aplica como el no uso de operaciones de "omitir conexiones", la agregación guiada bilateral y una estrategia de entrenamiento de refuerzo. Finalmente, se muestran los resultados obtenidos en la implementación de ambos modelos en este trabajo, los cuales se muestran inferiores en los dos casos debido al bajo poder computacional usado en este trabajo y mostrando así la poca aplicabilidad de estos modelos en sistemas de bajos recursos, pero mostrando un mejor rendimiento el Fast-SCNN sobre el BiSeNet V2 en este tipo de sistemas.**

***Keywords*—Segmentación semántica, tiempo real, alta resolución, tiempo de inferencia, precisión de predicción, enfoque de dos ramas, sistemas de bajos recursos.**

I. Introducción

La segmentación semántica de imágenes es una tarea importante de la visión computacional en donde se busca etiquetar densamente cada pixel de una imagen según su categoría de objeto y dar como resultado una imagen con regiones significativas que no se superponen, como se muestra en la Fig. 1. Mediante esta tarea se puede obtener información exhaustiva como la clasificación de objetos, obtención de formas o localización espacial, las cuales ayudan a comprender mejor un entorno. Lograr esto en tiempo real permite su aplicación en áreas de investigación que requieren capacidades de visión artificial de alta gama. Una de estos campos es la conducción autónoma, en donde los vehículos autónomos deben comprender el entorno que los rodea; es decir, deben ser capaces de reconocer otros automóviles, peatones, carriles de carretera, señales de tráfico o semáforos. Los sistemas de conducción autónoma requieren hacer la segmentación semántica en tiempo real debido a su necesidad de tomar decisiones en intervalos precisos [12].

Muchas técnicas de visión computacional y aprendizaje máquina han sido propuestas, pero el uso del aprendizaje profundo en los métodos de segmentación semántica ha logrado un gran progreso, mejorando su rendimiento en términos de precisión, y convirtiéndose en una solución segura [7].

Sin embargo, estos métodos desarrollados todavía no son prácticos para una segmentación semántica en tiempo real, ya que buscan alcanzar precisiones más altas con un número cada vez mayor de parámetros y módulos complejos, y generando por ende un gran costo computacional. Esto repercute en un tiempo de inferencia grande, lo cual es una seria limitante para la aplicabilidad en tiempo real de estos métodos de segmentación semántica [7].

En ese sentido, es un reto atractivo para los investigadores desarrollar nuevas arquitecturas que no solo demanden un buen rendimiento con una baja energía y memoria, sino también una buena velocidad de inferencia, logrando hacer un balance entre precisión y velocidad. Para esto se basan en diferentes técnicas o enfoques como reducir la resolución de las imágenes de entrada, modificar las convoluciones dentro de la red, utilizar redes de dos o más ramas, o hacer una mezcla de canales. De todos estos, el enfoque de dos ramas es uno que fue diseñado

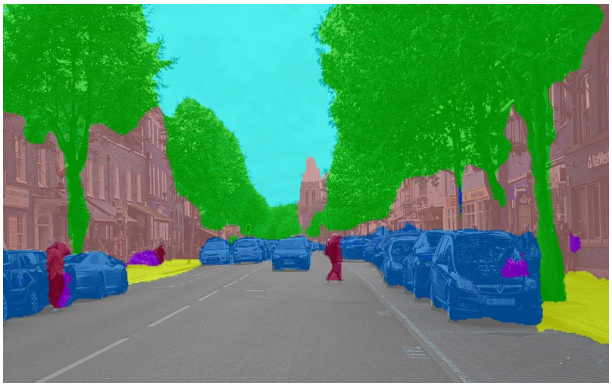

Fig. 1. Ejemplo de segmentación semántica

específicamente como una solución para el problema de la segmentación semántica en tiempo real y tiene la capacidad de obtener un balance entre velocidad y precisión.

Por tal motivo este trabajo busca abortar métodos prometedores que utilicen el enfoque de dos ramas y logren una segmentación semántica en tiempo real de una manera eficiente, es decir, que tenga una mejor velocidad. Para esto se escogerá los métodos BiSeNet V2 [15] y el Fast-SCNN [2] que muestran un buen desempeño con respecto a su velocidad y que utilizan el mismo enfoque con el fin de poder realizar un análisis entre ambas. Además, se realizará la implementación de los dos métodos a fin de poner realizar una comparación entre los resultados obtenidos en este trabajo y los resultados de los trabajos originales.

El resto del trabajo está organizado de la siguiente manera: un breve resumen de los trabajos relacionados más relevantes en esta área es presentado en la sección 2. En la sección 3 se hace una descripción detallada de las dos técnicas a implementar y un análisis de sus metodologías. En la sección 4 se muestran los resultados obtenidos de la implementación de ambos modelos y una comparación entre estos. Y finalmente en la sección 5 se describen las conclusiones del trabajo.

II. Trabajos Relacionados

A pesar de que los modelos de segmentación semántica basados en aprendizaje profundo han logrado una gran precisión en los últimos años, aún se busca que estos modelos requieran un menor tiempo de inferencia, lo cual es vital para aplicaciones como la conducción autónoma [12]. A continuación, se presenta los modelos actuales de segmentación semántica en tiempo real, agrupados en función de los enfoques que utilizaron.

En los métodos que usan redes de dos ramas, el contexto global se aprende con una entrada de resolución reducida en una rama profunda, mientras que las características espaciales se aprenden con una entrada de resolución completa en una rama poco profunda. SS [13] utiliza una rama para recibir la imagen y otra rama para recibir la imagen a mitad de su resolución. ContextNet [14] con la primera rama logra una segmentación precisa a baja resolución, mientras que con la segunda combina una subred a alta resolución para proporcionar resultados de segmentación detallados. Fast-SCNN propone un módulo llamado "aprender a reducir la muestra" para calcular características de bajo nivel para múltiples ramas en paralelo. BiSeNet [6] presenta un módulo de fusión de características y un módulo de refinamiento de la atención para filtrar las funciones de cada etapa. La evolución del BiSeNet, BiSeNet V2 cuenta con una capa de agregación guiada para fusionar características extraídas de las ramas de detalle y semántica.

Algunos métodos utilizan más de dos ramas en sus diseños para lograr una arquitectura liviana que combina con la posibilidad de aprendizaje de representación de características de bajo y alto nivel. FasterSeg [16] emplea la búsqueda de arquitectura neuronal y propone una regularización de latencia desacoplada y fina para equilibrar la compensación entre alta velocidad de inferencia y baja precisión. ESNet [17] consta de un módulo de unidad de convolución factorizada en paralelo con convoluciones paralelas de múltiples ramas, convoluciones dilatadas de múltiples ramas y convoluciones puntuales. ShelfNet18 [18] está compuesto de múltiples ramas codificador-decodificador. ICNet [5] propone un marco para guardar operaciones en múltiples resoluciones y cuenta con una unidad de fusión de funciones en cascada.

Otros métodos buscan modificar la forma en que funcionan las convoluciones en la arquitectura de red para aumentar la velocidad. ESPNet [19] introduce un módulo convolucional, llamado "pirámide espacial eficiente". ESPNet V2 [20] introduce una pirámide espacial extremadamente eficiente la cual sustituye las convoluciones puntuales por convoluciones grupales puntuales. ESSGG [21] utiliza convoluciones separables en profundidad y convoluciones agrupadas. DBANet [4] utiliza un módulo que usa convoluciones asimétricas en profundidad y convoluciones dilatadas, el cual extrae información combinada local y contextual, y reduce el número de parámetros. DFANet [3] se centra en la agregación profunda de características utilizando varias rutas de codificación interconectadas para agregar contexto de alto nivel a las características codificadas. ShuffleSeg [22] utiliza convoluciones agrupadas en el codificador para mejorar el rendimiento. HarDNet [23] utiliza la convolución separable en profundidad. ERFNet [24] constituye una capa que presenta conexiones residuales y convoluciones factorizadas.

Para el caso de la mezcla de canales, muchos métodos utilizaron este enfoque en donde a diferencia de una convolución estándar, donde cada canal de entrada está asociado con un solo canal de salida, una convolución de grupo adquiere datos de diferentes grupos de entrada. Esta técnica fue introducida con ShuffleNet [25] y también es usada por ShuffleSeg y LEDNet [26] para aumentar la velocidad de inferencia y su eficiencia.

La reducción de resolución temprana busca reducir el procesamiento de grandes imágenes de entrada, para así obtener pequeños conjuntos de mapas de características. ENet [27] introduce la reducción temprana para obtener operaciones de baja latencia en su arquitectura codificador-decodificador.

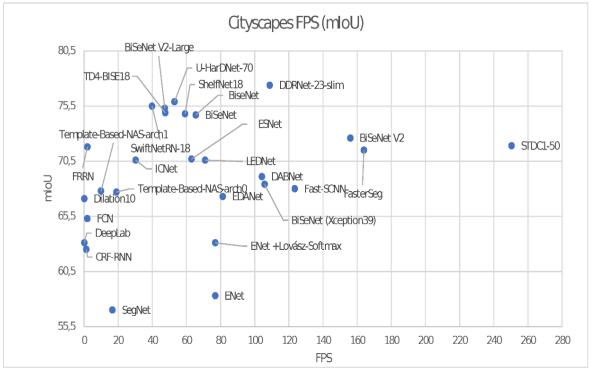

Fig. 2. Rendimiento en términos de FPS en relación con mIoU para modelos del estado de arte en el conjunto de datos Cityscapes. Imagen extraida de [12].

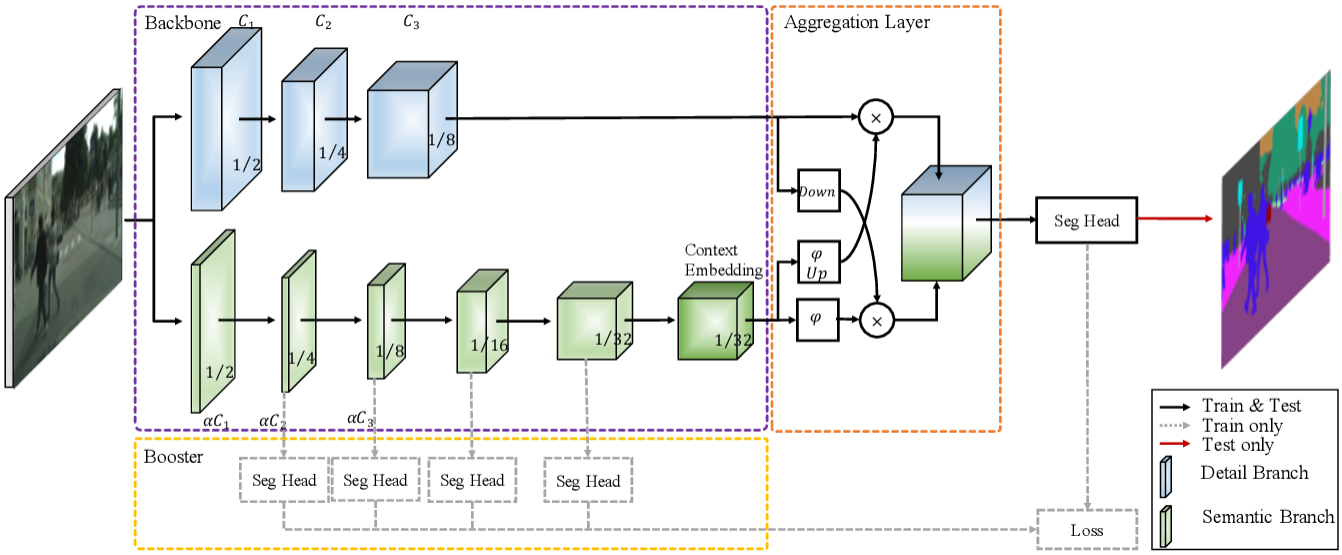

Fig. 3. Vista general de la arquitectura de BiSeNet V2, compuesta principalmente de 3 componentes: la columna vertebral de dos vias en el cuadro punteado de purpura, la capa de agregacion en el cuadro naranja, y parte de refuerzo en el cuadro amarrillo. La columna vertebral de dos vias tiene un rama de detalle (los cubos azules) y una rema semantica (los cubos verdes). Imagen extraida de [15]

EDANet [28] usa la reducción de resolución temprana y una conexión densa para mejorar la eficiencia y mantener bajo el costo computacional, así también BiSeNet V2 también incluye la reducción de resolución temprana en su red.

Finalmente, a pesar de que muchos de estos modelos logran una precisión promedio bastante buena, estos no logran esta misma medida para ciertos objetos individualmente dentro de una imagen. DSNet [7] se centró en el problema de la distribución no balanceada de los objetos en una imagen, en donde usó una unidad y arquitectura codificador-decodificador asimétrica y una función de perdida focal ponderada por objetos.

III. Técnicas a implementar

A. Preámbulo

Las arquitecturas utilizadas por los métodos para la segmentación semántica que logran una alta precisión, como son la arquitectura codificador-decodificador o la red troncal de dilatación, fueron diseñadas para un propósito de segmentación general sin importarles mucho la velocidad de inferencia o su costo computacional. Tratando de cubrir este problema, otros métodos basados en ambas arquitecturas, usan enfoques para lograr aumentar su velocidad de inferencia sacrificando la precisión de sus predicciones (como la reducción de la resolución temprana o la poda de canales). Por otro lado, la arquitectura de dos ramas, que fue diseñada específicamente para la segmentación semántica en tiempo real, logra un mejor balance entre la velocidad de inferencia y la precisión en la predicción.

El trabajo de investigación sobre el estado del arte para la segmentación semántica en tiempo real [12], muestra una gráfica en la que compara los diferentes métodos en función de su velocidad alcanzada y la precisión en su predicción para el conjunto de prueba de Cityscapes, como se muestra en la Fig. 2.

De los métodos más sobresalientes que logran un balance entre velocidad y precisión, están STDCI-50 [29], FasterSeg, BiSeNet V2, DDRNet-23-slim [30] y Fast-SCNN, de los cuales los tres últimos usan el enfoque de dos ramas y los dos primeros usan enfoques diferentes. Es por eso que para este trabajo en investigación se escogerá los trabajos de BiSeNet V2 y Fast-SCNN debido a que ambos utilizan un mismo enfoque y que son los trabajos que logran un mejores tiempos de inferencia.

Para el enfoque de dos ramas, se sabe que un largo campo receptivo es importante para aprender las complejas correlaciones entre clases de objetos. Además, la información espacial de una imagen es necesaria para preservar los límites de un objeto. Por eso en este enfoque es importante tanto la información espacial como el campo receptivo y busca sostener ambos al mismo tiempo. El modo en que los dos trabajos seleccionados varían esta principalmente en el modo en el que juntan ambos caminos para lograr la predicción final. Además, las capas que conforman ambos caminos son diferentes entre métodos.

BiSeNet V2 consta de una rama de detalle, con canales de alta capacidad y capas pocas profundas con un pequeño campo receptivo para capturar detalles de bajo nivel y generar una representación de características de alta resolución. También consta de una rama semántica, con canales de baja capacidad y capas profundas con gran campo receptivo para obtener un contexto semántico de alto nivel. La rama semántica es liviana debido a la reducción de la capacidad del canal y usa una estrategia de reducción de resolución rápida. Adicionalmente consta con una capa de agregación guiada para mejorar las conexiones mutuas y fusionar ambos tipos de representación

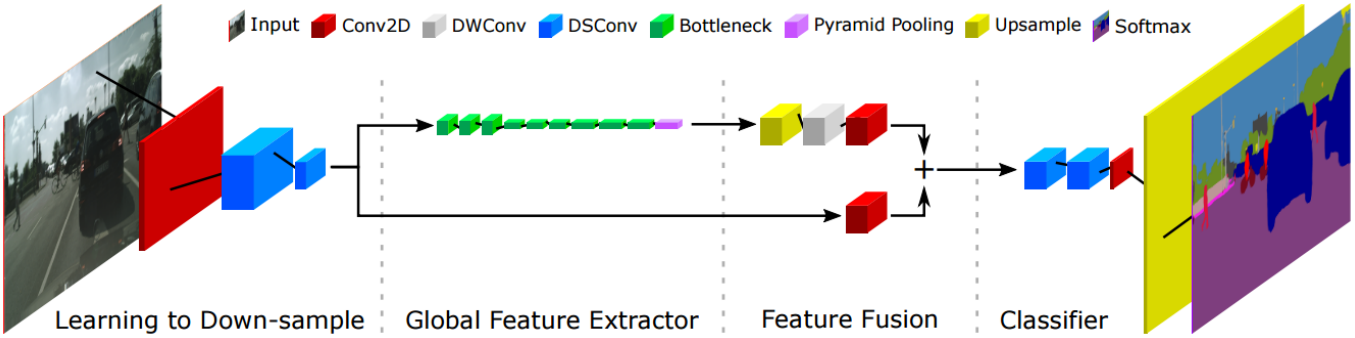

Fig. 4. Vista general de la arquitectura de Fast-SCNN, compuesta principalmente de 4 componentes: el módulo "aprendiendo a bajar la muestra" al inicio, seguido del extractor de características globales, el módulo de fusion de características, y al final el clasificador. Imagen extraida de [2]

de características. Además, se diseñó una estrategia de entrenamiento de refuerzo para mejorar el rendimiento de la segmentación sin ningún coste adicional de inferencia. Como resultado, este método logra 72.6% de mIoU para una entrada de 2048x1024 en el conjunto de pruebas de Cityscapes con una velocidad de 156 FPS en una NVIDIA GeForce GTX 1080 Ti. El detalle de esta arquitectura se puede ver en la Fig. 3.

Por otro lado, Fast-SCNN fusiona la configuración de dos ramas con el marco de codificador-decodificador clásico. Esta red primeramente usa un módulo llamado "aprendiendo a bajar la muestra" que adapta la conexión de salto, reduce la muestra y comparte la computación de las capas iniciales entre las ramas. Luego usa un extractor de características globales, para capturar el contexto global de la segmentación de imágenes. Seguidamente usa un módulo de fusión de características para la adición de las características de bajo y alto nivel. Y finalmente se tiene un clasificador estándar. La capacidad del modelo se mantiene especialmente baja para lograr una ejecución en dispositivos integrados y una mejor generalización. Esta red logro obtener una precisión del 68.0% de media IoU con una velocidad de 123.5 FPS en el conjunto de pruebas Cityscapes para imágenes de alta resolución (1024x2048), en una tarjeta Nvidia Titan XP. El detalle de esta arquitectura se puede ver en la Fig. 4.

A continuación, se detallará la propuesta de cada uno de los métodos.

B. BiSeNet V2: Bilateral Network with Guided Aggregation for Real-time Semantic Segmentation

1) Rama de detalle: Consta de 3 etapas de dos capas, donde cada capa es una capa de convolución seguida de normalización de lotes y una función de activación. La primera capa de cada etapa tiene un stride de 2, mientras que las otras capas en la misma etapa tienen el mismo número de filtros y tamaño de mapa de características de salida. Por lo tanto, esta rama extrae los mapas de características de salida que son 1/8 de la entrada original.

2) Rama semántica: Consta como primera etapa de un bloque vástago, el cual usa dos formas diferentes de reducir la resolución para reducir la representación de las características y que cuyas salidas se concatenan como salida final. Luego siguen 3 etapas de dos capas de reunión y expansión las cuales toman ventaja de los beneficios de las convoluciones en profundidad. Y finalmente, un bloque de inserción de contexto, el cual usa la combinación de promedio global y la conexión residual para incrustar la información contextual global de manera eficiente.

3) Agregación guiada bilateral: Esta capa emplea la información contextual de rama semántica para guiar la respuesta de las características de salida de la rama de detalle. Debido a la diferencia de las dimensiones espaciales de las salidas de cada rama, se tiene que hacer un reajuste de las dimensiones mediante operación de reducción de la resolución y aumento de la resolución, para así adoptar una estrategia de agregación bilateral.

4) Estrategia de entrenamiento de refuerzo: Mediante esta se puede mejorar la representación de características en la fase entrenamiento y puede descartarse en la fase de inferencia. Por lo tanto, aumenta poca complejidad de cálculo en la fase de inferencia. Este consta de un cabezal de segmentación auxiliar el cual se puede insertar en diferentes posiciones de la rama semántica.

C. Fast-SCNN: Fast Semantic Segmentation Network

1) Aprender a reducir la muestra: Se diseño esta técnica en donde se comparte las computaciones de las capas iniciales en el enfoque de dos ramas. En esta solo se emplea tres capas para garantizar que el uso compartido de características de bajo nivel sea válido y se implemente de manera eficiente. La primera capa es una capa de convolución estándar (Conv2D) y las dos capas restantes son capas convolucionales separables en profundidad (DSConv). Todas las capas usan un stride de 2, seguidas de una normalización de lotes y ReLU. El tamaño del kernel es de 3x3.

2) Extractor de características globales: Tiene como objetivo capturar el contexto global para la segmentación de imágenes. Esta toma la salida del módulo "aprender a reducir la muestra" la cual es 1/8 de la resolución de la entrada original. En esta se usa un bloque residual de cuello de botella. En particular, se usa conexiones residuales para los bloques residuales de cuello de botella cuando la entrada y la

salida son del mismo tamaño. El bloque de cuello de botella utiliza convoluciones separables en profundidad. Finalmente, un módulo de agrupación piramidal es adicionado al final para agregar la información de contexto basada en regiones diferentes.

3) Módulo de fusión de características: En este modelo se usa una simple adición de las características para asegurar la eficiencia. Para esto se usa el módulo de fusión de características, el cual consta, por la rama de baja resolución de un aumento de resolución seguida de una convolución en profundidad. Luego, en ambas ramas se tiene a la par una capa de convolución estándar, para finalmente unir ambas en una suma.

4) Clasificador estándar: En este se emplea dos convoluciones separables en profundidad y una capa de convolución estándar. Se sabe que agregando pocas capas después del módulo de fusión de características aumenta la precisión.

D. Conjunto de datos

Ambos modelos para poder realizar su entrenamiento y posterior validación utilizaron los siguientes conjuntos de datos de imágenes:

1) Cityscapes [9]: Es un gran conjunto de datos de escenas de calles urbanas desde la perspectiva del automóvil. Esta contiene 2975 imágenes con anotaciones finas para entrenamiento y otras 500 imágenes para validación. Para las pruebas, ofrece 1525 imágenes sin la verdad del terreno para una comparación justa. Todas las imágenes tienen una resolución de 2048x1024, en la que cada pixel esta anotado en 19 clases predefinidas.

2) CamVid [10]: Es otro conjunto de datos de escenas callejeras desde la perspectiva de un automóvil conduciendo. Contiene 701 imágenes en total, de las cuales 367 son para entrenamiento, 101 para validación y 233 para prueba. Las imágenes tienen una resolución de 960 × 720 y 11 categorías semánticas.

3) COCO-Stuff [11]: Aumenta las 164000 imágenes del popular conjunto de datos COCO, de las cuales tiene 118.000 imágenes para entrenamiento, 5.000 imágenes para validación, 20.000 imágenes para test-dev y 20.000 imágenes para test-challenge. Cubre 91 clases de material y 1 clase ”sin etiqueta”.

E. Métricas de evaluación de rendimiento

1) Intersección media sobre unión (mIoU): Es una métrica ampliamente usada para la segmentación semántica. Está definida como la intersección sobre unión (IoU) promedio de todas las clases. IoU se define como la intersección del mapa de segmentación pronosticado y la verdad del terreno, divida por el área de unión entre el mapa de segmentación pronosticado y la verdad del terreno [12].

2) Cuadros por segundo (FPS): Es una métrica estándar para evaluar el tiempo necesario que toma un modelo de aprendizaje profundo en procesar una serie de cuadros de imágenes de video. Es una métrica popular que sirve para comparar diferentes métodos y arquitecturas de segmentación [12].

3) Tiempo de inferencia: Es otra métrica que evalúa la velocidad, esta es la inversa de FPS, y mide el tiempo de ejecución por cada cuadro de imagen [12].

4) Uso de memoria: Esta puede ser medida de formas diferentes. Algunos investigadores usan el número de parámetros de la red, otros definen el tamaño de la memoria para representar la red y, por último, una métrica que se usa con frecuencia es medir el número de operaciones de punto flotante (FLOP) necesarias para la ejecución [12].

F. Comparación de modelos

Si bien ambos modelos usan el mismo enfoque de dos ramas para lograr el balance entre velocidad y precisión, estos muestran variaciones en diferentes partes de sus arquitecturas que los hacen funcionar de diferente manera. A continuación, veremos algunas de estas variaciones presentes en los modelos.

Con respecto a la velocidad obtenida por los modelos tenemos que BiSeNet V2 logra obtener una mayor velocidad de 156 FPS a comparación de Fast-SCNN que logra una velocidad de 123.5 FPS. Algunos puntos clave para explicar esto son los siguientes: (i) Fast-SCNN al realizar la extracción de las características espaciales de manera temprana para luego compartir estas características directamente con el módulo de fusión, busca simular las operaciones de “omitir conexiones” utilizadas para la segmentación semántica general que, si bien permiten mejorar la precisión, tienen un alto costo computacional. Estas en cambio en BiSeNet V2 son evitas para así lograr un procesamiento más rápido. (ii) La rama semántica del BiSeNet V2 busca ser ligera para reducir su complejidad de cómputo, en comparación de Fast-SCNN que no es una red ligera.

Hablando ahora de la precisión obtenida, BiSeNet V2 logra ser mejor con un 72.5% de mIoU, con comparación de Fast-SCNN con 68.0% de mIoU. Los motivos que explican esto son los siguientes: (i) La manera de fusionar las características a cargo de BiSeNet V2 logra una precisión similar a la obtenida por Fast-SCNN si solo se usara una simple adición para la fusión (68.6% de mIoU). Al hacer uso de la agregación guiada bilateral permite obtiener una mejor precisión de 69.67%. (ii) La estrategia de entrenamiento de refuerzo usada por BiSeNet V2 es la que le da una gran mejora a su precisión, obteniendo 72.5% de mIoU sobre Fast-SCNN, la cual resulta bastante útil ya que no sacrifica la velocidad de inferencia.

Un resumen de estos aspectos se encuentra en la Tabla I.

IV. Implementación de las técnicas

Los experimentos se llevaron a cabo mediante el uso de la librería Pytorch en Python y fueron ejecutados en Google Colab, el cual contiene una GPU Tesla K80 y una versión CUDA de 11.2. Para la evaluación de las técnicas propuestas se utilizó el conjunto de datos ya mencionado Cityscapes.

Para la carga de datos en el entrenamiento se utilizó un tamaño de lote de 12 para ambos modelos. Para el caso del modelo Fast-SCNN, se usa la función de perdida de entropía cruzada y el optimizador del descenso de gradiente estocástico (SGD), con una tasa de aprendizaje de 1e-2, un impulso de 0.9

Tabla I
COMPARACIÓN ENTRE LOS MODELOS BISENET V2 Y FAST-SCNN

BiSeNet V2	Fast-SCNN
Reducción de la resolución en una etapa temprana mediante el módulo "aprendiendo a bajar la muestra".	Reducción de la resolución en una etapa temprana dentro de la rama semántica.
Extracción de características espaciales mediante el módulo "aprendiendo a bajar la muestra" en una etapa temprana.	Extracción de características espaciales mediante la rama de detalle.
La fusión de características de ambas ramas la hace mediante una adición simple.	La fusión de características de ambas ramas la hace mediante una agregación guiada bilateral.
Obtiene una precisión del 68.0% mIoU y una velocidad de 123.5 FPS.	Obtiene una precisión del 72.5% mIoU y una velocidad de 156 FPS.

Tabla II
PARÁMETROS UTILIZADOS PARA LA FASE DE ENTRENAMIENTO

Parámetros	Fast-SCNN	BiSeNet V2
Función de perdida	Entropía cruzada	Entropía cruzada
Optimizador	SGD	SGD
Tasa de aprendizaje	1e-2	5e-2
Impulso	0.9	0.9
Decadencia de peso	1e-4	5e-4
Tamaño de lote	12	12

y una decadencia de peso de 1e-4. Para el caso del modelo de BiSeNet V2, también se usa la función de perdida de entropía cruzada y el optimizador SGD, con una tasa de aprendizaje de 5e-2, un impulso de 0.9 y una decadencia de peso de 1e-4. En ambos casos usando los parámetros de las implementaciones originales.

El consolidado de los parámetros utilizados en el entrenamiento tanto para el Fast-SCNN como para el BiSeNet V2 están detallados en la Tabla II.

Para la evaluación de los modelos, la carga de datos se hace con un tamaño de lote de 1, principalmente debido al límite de memoria de la GPU. En esta se obtiene la medida de precisión mIoU.

Se realizaron tres diferentes tipos de experimentos para imágenes de 1024x2048, 256x512 y 128x256. Esto con el motivo de realizar el mayor tiempo de entrenamiento posible, debido a las limitantes del uso de la GPU del Google Colab.

A. Precisiones de los modelos

En la Tabla III presentamos los resultados obtenidos de la predicción de los modelos para imágenes en diferentes resoluciones.

Para imágenes de 1024x2048, Fast-SCNN obtuvo una precisión del 32% de mIoU después de un entrenamiento de 13 épocas y BiSeNet V2 obtuvo una precisión del 13% de mIoU después de un entrenamiento de 11 épocas. Para imágenes de 256x512, Fast-SCNN obtuvo una precisión del 34% de mIoU después de un entrenamiento de 43 épocas y BiSeNet V2 obtuvo una precisión del 25% de mIoU después de un entrenamiento de 19 épocas. Y finalmente para imágenes de 128x256, Fast-SCNN obtuvo una precisión del 31% de mIoU después de un entrenamiento de 52 épocas y BiSeNet V2 obtuvo una precisión del 18% de mIoU después de un entrenamiento de 7 épocas. En la Fig. 5 y la Fig. 6 se muestran los resultados obtenidos por ambos modelos a diferentes resoluciones.

B. Velocidades de los modelos

Para la medición de los FPS se reportaron un promedio de 100 fotogramas. En este no fue necesario cargar las imágenes, simplemente basto con pasar una representación vectorial de cada imagen (torch).

Para el análisis de estas se utilizaron diferentes resoluciones de imágenes: 1024x2048, 256x512, 128x256. Los resultados obtenidos se muestran en la Tabla IV.

C. Discusión

En esta sección se hará una comparación de los resultados obtenidos en este trabajo con respecto a los obtenidos por los trabajos originales.

Como se observa en la Tabla III, los resultados obtenidos por este trabajo de mIoU, correspondientes a la precisión, se muestran inferiores a los obtenidos por el trabajo original. Esto es debido principalmente al poco entrenamiento que se hizo en ambos modelos, específicamente a la baja cantidad de épocas que se lograron entrenar por los límites de uso de GPU en el Google Colab.

En el caso de Fast-SCNN, hay que mencionar que para alcanzar los 68% de mIoU en imágenes de 1024x2048, en el trabajo original se tuvo un entrenamiento de 1000 épocas. Sin embargo, en los tres experimentos realizados, la cantidad de épocas no superan las 52 épocas, siendo esto menos del 5% de todo el entrenamiento original. Un detalle que hay que notar es que los resultados obtenidos en este trabajo con imágenes 1024x2048 y 256x512 muestran una precisión cercana a la mitad de la precisión deseada, lo que indica que este modelo podría funcionar bien sin la necesidad de mucho entrenamiento. Sin embargo, al bajarle la resolución a 128x256 y a pesar de hacer una mayor cantidad de épocas, la precisión baja. Esto nos indica que el Fast-SCNN no logra una buena predicción para bajas resoluciones con poco entrenamiento.

El caso de BiSeNet V2 es similar, para la obtención del 72.5% de mIoU en imágenes de 1024x2048, en el trabajo original se hizo un entrenamiento con 810 épocas. En cambio, las épocas alcanzadas en nuestros experimentos llegan máximo a las 19, lo cual representa menos del 3% del entrenamiento realizado originalmente. En este caso el entrenamiento para imágenes de una resolución de 128x256 logro menor cantidad de épocas (7 épocas) que las logradas para imágenes de 1024x2048 y 256x512. Sabiendo que debería suceder todo lo contrario (es decir, a menor resolución mayor entrenamiento), se puede ver una relación de este comportamiento con la medición de velocidad de este modelo a esta resolución, en donde también decae. Esto supone que el modelo ofrece muy bajo rendimiento para imágenes de baja resolución como 128x256. Por último, a diferencia del anterior modelo, este

Tabla III
COMPARACIÓN DE LOS MIOU(%) OBTENIDOS EN ESTE TRABAJO Y LOS TRABAJOS ORIGINALES.

Resolución	Fast-SCNN* (ep)	Fast-SCNN (ep)	BiSeNet V2* (ep)	BiSeNet V2 (ep)
1024x2048	32% (13)	68% (1000)	13% (11)	72.5% (810)
256x512	34% (43)	-	25% (19)	-
128x256	31% (52)	-	18% (7)	-

(*): Implementación en este trabajo, (ep): Número de épocas de entrenamiento, (-): Sin implementación

Tabla IV
COMPARACIÓN DE LOS FPS OBTENIDOS EN ESTE TRABAJO Y LOS TRABAJOS ORIGINALES.

Resolución	Fast-SCNN*	Fast-SCNN	BiSeNet V2*	BiSeNet V2
1024x2048	19	123.5	4.8	156
256x512	117	-	44	-
128x256	117	-	22	-

(*): Implementación en este trabajo, (-): Sin implementación

(a) (b) (c)

Fig. 5. Predicción realizada por Fast-SCNN para: (a) 1024x2048, (b) 256x512 y (c) 128x256.

(a) (b) (c)

Fig. 6. Predicción realizada por BiSeNet V2 para: (a) 1024x2048, (b) 256x512 y (c) 128x256.

se ve más pesado debido al poco entrenamiento realizado en todos los experimentos, con el uso de los recursos disponibles.

Finalmente, los resultados correspondientes a la velocidad (FPS) muestran ser muy inferiores a los esperados, como se muestra en la Tabla IV. El motivo de esto se sustenta en el poder computacional utilizado. En ambos modelos se utilizó la GPU Tesla K80 asignada por Google Colab, en comparación de las GPUs usadas por Fast-SCNN y BiSeNet V2 en sus trabajos originales, Nvidia Titan X/Xp y Nvidia GeForce GTX 1080Ti respectivamente. Esto es una muestra clara que ambos modelos no se ven realmente aplicables a sistemas de bajos recursos computacionales.

V. CONCLUSIONES

La segmentación semántica en tiempo real es una tarea importante para la conducción autónoma, pero plantea desafíos al momento de incrementar la velocidad de los modelos, sacrificando en muchos casos la precisión. De los enfoques planteados para lograr aumentar la velocidad en el estado del arte, el enfoque de dos ramas es uno que muestra muy buenos resultados, principalmente por que fue diseñado específicamente para realizar las tareas de segmentación logrando obtener un balance entre velocidad de inferencia y precisión de predicción.

Los dos métodos seleccionados para este trabajo muestran ser los que logran una mejor velocidad dentro del enfoque de dos ramas, sin embargo, abordan la solución de maneras diferentes. En la parte de extracción de características espaciales, BiSeNet V2 evita realizar operaciones de “omitir conexiones”, las cuales son usadas por Fast-SCNN y que tienen un alto costo computacional. Además, que la rama semántica del BiSeNet V2 busca ser ligera para reducir su complejidad de cómputo, en contraste con el Fast-SCNN. Esto explica los resultados obtenidos respecto al tiempo de 156 FPS para BiSeNet V2 y 123.5 FPS para el Fast-SCNN. En la fusión de características, Fast-SCNN realiza una adición simple a diferencia del BiSeNet V2, que usa una agregación guiada

bilateral que logra incrementar en un pequeño porcentaje la precisión a comparación del Fast-SCNN. Por último, debido a la estrategia de entrenamiento por parte BiSeNet V2 es que logra obtener una precisión de 72.6% mIoU, a comparación del Fast-SCNN que no utiliza la misma estrategia y que logra una precisión de 68.0% mIoU.

Con respecto a la implementacion de los modelos realizada por este trabajo, si bien no se obtuvieron resultados similares a los originales debido principalmente a los recursos computacionales disponibles, se pueden sacar algunas conclusiones. Primero que el modelo Fast-SCNN logro resultados prometedores de 31% y 34% de mIoU para imágenes de 1024x2048 y 256x512, con un poco porcentaje del entrenamiento usado por el trabajo original, mostrando que este modelo no requiere mucho entrenamiento. Sin embargo, este no logra una buena predicción con imágenes de baja resolución. En el caso de BiSeNet V2, muestra ser un modelo bastante pesado para sistemas de bajos recursos, ya que solo logro menos del 3% del entrenamiento total en los experimentos, además de tener un bajo rendimiento específicamente para imágenes de 128x256. Así mismo, la velocidad de ambos modelos es muy inferior a la esperada, 19 FPS para Fast-SCNN y 4.8 FPS para BiSeNet v2, lo cual denota que ambos modelos realmente no son aplicables a sistemas de bajos recursos. Pero de ambos modelos implementados, el que tuvo un mejor comportamiento en base a los recursos utilizados fue el Fast-SCNN tanto en el entrenamiento, y por ende en la precisión, así como en la velocidad.

Finalmente, el enfoque de dos ramas es uno de los más seguro para el estudio de la segmentación semántica en tiempo real. Sin embargo, nuevas técnicas prometedoras están apareciendo en este campo, como el aprendizaje auto supervisado, el aprendizaje débilmente supervisado o el aprendizaje transferible, los cuales se buscarán adaptarse a futuros esfuerzos de investigación para mejorar el rendimiento en la segmentación semántica en tiempo real.

Referencias

[1] S. Hao, Y. Zhou, and Y. Guo, "A brief survey on semantic segmentation with deep learning," *Neurocomputing*, vol. 406, pp. 302–321, 2020.

[2] R. P. K. Poudel, S. Liwicki, and R. Cipolla, "Fast-scnn: Fast semantic segmentation network," 2019.

[3] H. Li, P. Xiong, H. Fan, and J. Sun, "Dfanet: Deep feature aggregation for real-time semantic segmentation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[4] G. Li, I. Yun, J. Kim, and J. Kim, "Dabnet: Depth-wise asymmetric bottleneck for real-time semantic segmentation," 2019.

[5] H. Zhao, X. Qi, X. Shen, J. Shi, and J. Jia, "Icnet for real-time semantic segmentation on high-resolution images," in *Computer Vision – ECCV 2018*, V. Ferrari, M. Hebert, C. Sminchisescu, and Y. Weiss, Eds. Cham: Springer International Publishing, 2018, pp. 418–434.

[6] C. Yu, J. Wang, C. Peng, C. Gao, G. Yu, and N. Sang, "Bisenet: Bilateral segmentation network for real-time semantic segmentation," in *Computer Vision – ECCV 2018*, V. Ferrari, M. Hebert, C. Sminchisescu, and Y. Weiss, Eds. Cham: Springer International Publishing, 2018, pp. 334–349.

[7] W. Wang, Y. Fu, Z. Pan, X. Li, and Y. Zhuang, "Real-time driving scene semantic segmentation," *IEEE Access*, vol. 8, pp. 36 776–36 788, 2020.

[8] F. Chollet, "Xception: Deep learning with depthwise separable convolutions," *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1800–1807, 2017.

[9] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele, "The cityscapes dataset for semantic urban scene understanding," in *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[10] G. J. Brostow, J. Shotton, J. Fauqueur, and R. Cipolla, "Segmentation and recognition using structure from motion point clouds," in *Computer Vision – ECCV 2008*, D. Forsyth, P. Torr, and A. Zisserman, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 44–57.

[11] H. Caesar, J. R. R. Uijlings, and V. Ferrari, "Coco-stuff: Thing and stuff classes in context," *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 1209–1218, 2018.

[12] I. Papadeas, L. Tsochatzidis, A. Amanatiadis, and I. Pratikakis, "Real-time semantic image segmentation with deep learning for autonomous driving: A survey," *Applied Sciences*, vol. 11, no. 19, 2021.

[13] Z. Wu, C. Shen, and A. van den Hengel, "Real-time semantic image segmentation via spatial sparsity," 2017.

[14] R. P. K. Poudel, U. Bonde, S. Liwicki, and C. Zach, "Contextnet: Exploring context and detail for semantic segmentation in real-time," 2018.

[15] C. Yu, C. Gao, J. Wang, G. Yu, C. Shen, and N. Sang, "Bisenet v2: Bilateral network with guided aggregation for real-time semantic segmentation," *Int. J. Comput. Vis.*, vol. 129, pp. 3051–3068, 2021.

[16] W. Chen, X. Gong, X. Liu, Q. Zhang, Y. Li, and Z. Wang, "Fasterseg: Searching for faster real-time semantic segmentation," in *International Conference on Learning Representations*, 2020.

[17] Y. Wang, Q. Zhou, and X. Wu, "Esnet: An efficient symmetric network for real-time semantic segmentation," in *PRCV*, 2019.

[18] J. Zhuang, J. Yang, L. Gu, and N. Dvornek, "Shelfnet for fast semantic segmentation," in *2019 IEEE/CVF International Conference on Computer Vision Workshop (ICCVW)*. Los Alamitos, CA, USA: IEEE Computer Society, oct 2019, pp. 847–856.

[19] S. Mehta, M. Rastegari, L. S. Anat Caspi, and H. Hajishirzi, "Espnet: Efficient spatial pyramid of dilated convolutions for semantic segmentation," in *ECCV*, 2018.

[20] S. Mehta, M. Rastegari, L. Shapiro, and H. Hajishirzi, "Espnetv2: A light-weight, power efficient, and general purpose convolutional neural network," in *CVPR*, 2019.

[21] N. Vallurupalli, S. Annamaneni, G. Varma, C. V. Jawahar, M. Mathew, and S. Nagori, "Efficient semantic segmentation using gradual grouping," *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pp. 711–7118, 2018.

[22] M. Gamal, M. Siam, and M. Abdel-Razek, "Shuffleseg: Real-time semantic segmentation network," *ArXiv*, vol. abs/1803.03816, 2018.

[23] P. Chao, C.-Y. Kao, Y. Ruan, C.-H. Huang, and Y.-L. S. Lin, "Hardnet: A low memory traffic network," *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 3551–3560, 2019.

[24] E. Romera, J. M. Álvarez, L. M. Bergasa, and R. Arroyo, "Erfnet: Efficient residual factorized convnet for real-time semantic segmentation," *IEEE Transactions on Intelligent Transportation Systems*, vol. 19, no. 1, pp. 263–272, 2018.

[25] X. Zhang, X. Zhou, M. Lin, and J. Sun, "Shufflenet: An extremely efficient convolutional neural network for mobile devices," *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 6848–6856, 2018.

[26] Y. Wang, Q. Zhou, J. Liu, J. Xiong, G. Gao, X. Wu, and L. J. Latecki, "Lednet: A lightweight encoder-decoder network for real-time semantic segmentation," *2019 IEEE International Conference on Image Processing (ICIP)*, pp. 1860–1864, 2019.

[27] A. Paszke, A. Chaurasia, S. Kim, and E. Culurciello, "Enet: A deep neural network architecture for real-time semantic segmentation," *ArXiv*, vol. abs/1606.02147, 2016.

[28] S.-Y. Lo, H.-M. Hang, S.-W. Chan, and J.-J. Lin, "Efficient dense modules of asymmetric convolution for real-time semantic segmentation," in *Proceedings of the ACM Multimedia Asia*, ser. MMAsia '19. New York, NY, USA: Association for Computing Machinery, 2019.

[29] M. Fan, S. Lai, J. Huang, X. Wei, Z. Chai, J. Luo, and X. Wei, "Rethinking bisenet for real-time semantic segmentation," *ArXiv*, vol. abs/2104.13188, 2021.

[30] Y. Hong, H. Pan, W. Sun, and Y. Jia, "Deep dual-resolution networks for real-time and accurate semantic segmentation of road scenes," *arXiv preprint arXiv:2101.06085*, 2021.