

Centro Nacional de Investigación y Desarrollo Tecnológico

Subdirección Académica

Departamento de Ciencias Computacionales

TESIS DE MAESTRÍA EN CIENCIAS

**Sintonización de una Red Totalmente Conectada para Segmentación
de Dos Clases de Objetos en Imágenes**

presentada por

Lic. Diego Gabriel Suárez Santiago

como requisito para la obtención del grado de
Maestro en Ciencias de la Computación

Director de tesis
Dr. Dante Mújica Vargas

Codirector de tesis
Dr. Manuel Mejía Lavalle

Cuernavaca, Morelos, México. Enero de 2018.

Cuernavaca, Morelos a 15 de enero del 2018
OFICIO No. DCC/030/2018**Asunto:** Aceptación de documento de tesis**C. DR. GERARDO V. GUERRERO RAMÍREZ**
SUBDIRECTOR ACADÉMICO
PRESENTE

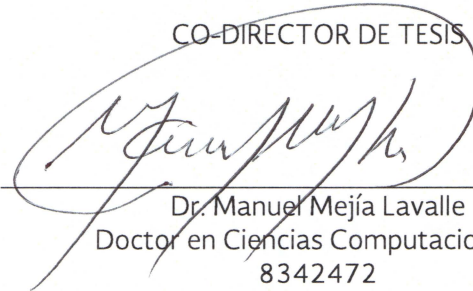
Por este conducto, los integrantes de Comité Tutorial del **Lic. Diego Gabriel Suárez Santiago**, con número de control M15CE094, de la Maestría en Ciencias de la Computación, le informamos que hemos revisado el trabajo de tesis profesional titulado **“Sintonización de una red totalmente conectada para segmentación de dos clases de objetos en imágenes”** y hemos encontrado que se han realizado todas las correcciones y observaciones que se le indicaron, por lo que hemos acordado aceptar el documento de tesis y le solicitamos la autorización de impresión definitiva.

DIRECTOR DE TESIS



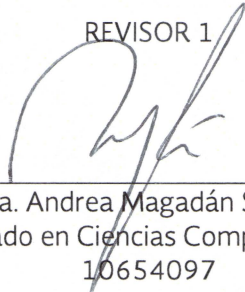
Dr. Dante Mújica Vargas
Doctor en Comunicaciones y Electrónica
09131756

CO-DIRECTOR DE TESIS



Dr. Manuel Mejía Lavalle
Doctor en Ciencias Computacionales
8342472

REVISOR 1



Dra. Andrea Magadán Salazar
Doctorado en Ciencias Computacionales
10654097

REVISOR 2



Dr. Raúl Pinto Elías
Doctor en Ciencias en la Especialidad de Ingeniería
Eléctrica
3890453

C.p. M.C. María Elena Gómez Torres - Jefa del Departamento de Servicios Escolares.
Estudiante
Expediente

NACS/lmz

SEP

SECRETARÍA DE
EDUCACIÓN PÚBLICA



TECNOLÓGICO NACIONAL DE MÉXICO
Centro Nacional de Investigación y Desarrollo Tecnológico

Cuernavaca, Mor., 17 de enero de 2018
OFICIO No. SAC/060/2018

Asunto: Autorización de impresión de tesis

**LIC. DIEGO GABRIEL SUÁREZ SANTIAGO
CANDIDATO AL GRADO DE MAESTRO EN CIENCIAS
DE LA COMPUTACIÓN
PRESENTE**

Por este conducto, tengo el agrado de comunicarle que el Comité Tutorial asignado a su trabajo de tesis titulado **“Sintonización de una red totalmente conectada para segmentación de dos clases de objetos en imágenes”**, ha informado a esta Subdirección Académica, que están de acuerdo con el trabajo presentado. Por lo anterior, se le autoriza a que proceda con la impresión definitiva de su trabajo de tesis.

Esperando que el logro del mismo sea acorde con sus aspiraciones profesionales, reciba un cordial saludo.

ATENTAMENTE

“CONOCIMIENTO Y TECNOLOGÍA AL SERVICIO DE MÉXICO”

**DR. GERARDO VICENTE GUERRERO RAMÍREZ
SUBDIRECTOR ACADÉMICO**



**SÉP TecNM
CENTRO NACIONAL
DE INVESTIGACIÓN
Y DESARROLLO
TECNOLÓGICO
SUBDIRECCIÓN
ACADÉMICA**

C.p. M.T.I. María Elena Gómez Torres.- Jefa del Departamento de Servicios Escolares.
Expediente

GVGR/mcr

Dedicatorias

Esta tesis se lo dedico a mi abuela †Petronila Hernández Sánchez y a mi hermano †José Antonio Zárate Santiago, que ya descansan y gozan de la vida eterna.

Agradecimientos

A Dios:

Mi primer y mayor agradecimiento es a Dios que me ha dado la vida, así como todo lo que tengo y lo que soy.

A las instituciones de mi país:

Específicamente al Conacyt por el apoyo económico prestado durante los dos años de la maestría. Así mismo, al Centro Nacional de Investigación y Desarrollo Tecnológico (CENIDET), del cual me enorgullezco de haber sido parte, y con la cual le estoy totalmente agradecido por la oportunidad que me brindó al aceptarme y formarme como alumno.

A mis profesores:

Mi director de tesis, el Dr. Dante Mújica Vargas por todos sus consejos, revisiones y dedicación para llevar a buen término este trabajo de tesis. Así como a mi comité, el Dr. Manuel Mejía Lavalle, la Dra. Andrea Magadán Salazar y el Dr. Raúl Pinto Elías, por todos sus aportes que me dieron para mejorar esta tesis. Así, como todos los demás profesores que me formaron durante mi estancia en el CENIDET.

A mi familia:

Mi abuela †Petronila Hernández Sánchez, quien fue toda una madre para mí, y que en gran parte gracias a ella estoy hasta aquí.

Mi tía, Juana Santiago Hernández, por ser otra madre para mí, y que ha dedicado toda su vida a cuidarme.

Mi madre, Ninfa Justina Santiago Hernández por otorgarme el apoyo moral y económico. Mis hermanos, Viridiana Hernández Santiago y †José Antonio Zárate Santiago.

Y a todos mis demás familiares por darme su cariño incondicional en todo momento.

A mi novia:

Dalia Clemente Hernández, por todo su apoyo, paciencia y amor durante el tiempo transcurrido al realizar esta maestría, ya que a pesar de la distancia perseveramos como pareja.

A mis amigos:

A mis compañeros de generación por darme sus valiosas sugerencias respecto a esta tesis.

A mis hermanos en Cristo de la Escuela de Pastoral y demás grupos de la Iglesia Católica, por todo este tiempo de sana convivencia.

Resumen

En esta tesis se realizó la sintonización de una Red Neuronal Convolutiva (*Convolutional Neural Network*, CNN) para poder segmentar dos clases de objetos del repositorio BSDS500 del Grupo de Visión por Computadora (*Computer Vision Group*, CVG) de la Universidad de Berkeley. Las imágenes que se utilizaron para evaluar este trabajo fueron obtenidas del repositorio mencionado, ya que éstas presentan condiciones que hacen difícil realizar este proceso, por ejemplo: la presencia de sombras, texturas y colores similares, iluminación no uniforme, oclusión, entre otras; además, de proporcionar métricas para evaluar la calidad de la segmentación, así como segmentaciones manuales sugeridas por diferentes usuarios.

En cuanto a las Redes Neuronales Convolutivas, se utilizaron los modelos FCN-Alexnet y FCN-8s para realizar los experimentos, dando mejores resultados éste último, tanto cualitativa como cuantitativamente, con las métricas: Índice Probabilístico Rand (PRI), Variación de la Información (VI) y Error de Consistencia Global (GCE). Las Redes Neuronales Convolutivas se consideran técnicas de Aprendizaje Profundo (*Deep Learning*, DL) debido a que cuentan con más de una capa oculta. Estas redes permiten la extracción de características significativas en sus capas de convolución, las cuales son útiles para realizar diversas tareas, y que han demostrado su efectividad en el campo de Visión Artificial (Lee *et al.*, 2009).

Estos modelos fueron implementados en una Unidad de Procesamiento Gráfico (*Graphics Processing Unit*, GPU) NVIDIA, para agilizar el procesamiento de las imágenes, optimizar los recursos disponibles y que fuera factible la realización de múltiples entrenamientos y pruebas con dicho modelo.

Palabras Clave: Segmentación de imágenes a color, Redes Neuronales Convolutivas, Aprendizaje Profundo, GPU.

Abstract

In this thesis we tuned a Convolutional Neural Network (*Convolutional Neural Network*, CNN) to be able to segment two classes of objects from BSDS500 dataset of the Computer Vision Group (CVG) from the University of Berkeley. The images that were used to evaluate this work were obtained from the mentioned dataset, since they present conditions that make this process difficult, for example: the presence of shadows, textures and similar colors, non-uniform illumination, occlusion, among others; in addition, to provide metrics to evaluate the quality of the segmentation, as well as manual segmentations suggested by different users.

Concerning the Convolutional Neural Networks, the FCN-Alexnet and FCN-8s models were used to carry out the experiments, giving better results the last model, both qualitatively and quantitatively, with the metrics: Rand Probabilistic Index (PRI), Variation of Information (VI) and Global Consistency Error (GCE). Convolutional Neural Networks are considered Deep Learning techniques (DL) because they have more than one hidden layer. These networks allow the extraction of significant features in their convolution layers, which are useful to perform various tasks, and have demonstrated their effectiveness in the field of Artificial Vision (Lee *et al.*, 2009).

This model will be implemented in NVIDIA's Graphics Processing Unit (GPU), as it is expected to streamline image processing, optimize available resources and make multiple tests possible with NVIDIA.

Keywords: Color image segmentation, Convolutional Neural Networks, Deep Learning, GPU.

Nomenclatura

α	Escalar
β	Exponente
b	Sesgo
C_i	i -ésima capa de convolución
crl	Tamaño del campo receptivo local
D	Profundidad de la imagen o matriz de entrada
D'	Cantidad de kernels o filtros
D''	Profundidad del mapa de características generado
k	Filtro de convolución
F_i	i -ésima capa totalmente conectada
GCE	Error de Consistencia Global
H	Número de filas de la imagen o matriz de entrada
H'	Número de filas del filtro de convolución
H''	Número de filas del mapa de características resultante
k	Filtro o máscara
L	Cantidad de capas ocultas de la red
LRN	Normalización de Respuesta Local
mem	Cantidad de memoria computacional utilizada
N	Tamaño de lote de entrada
N''	Tamaño de lote de salida
num_p	Número de pesos aprendidos
p	Probabilidad
pad	Relleno
P_h^+	Relleno de abajo
P_h^-	Relleno de arriba
P_w^+	Relleno de la derecha
P_w^-	Relleno de la izquierda
PRI	Índice Probabilístico de <i>Rand</i>
r_j^i	Conjunto de distribuciones según la función de Bernoulli
S_i	i -ésima capa de submuestreo
S	Paso
S_h	Paso en la altura
$shift$	Desplazamiento
S_w	Paso en la anchura
VI	Variación de Información

W	Número de columnas de la imagen o matriz de entrada
W'	Número de columnas del filtro de convolución
W''	Número de columnas del mapa de características resultante
x	Imagen o matriz de entrada
y	Mapa de característica

Índice general

Dedicatorias	i
Agradecimientos	ii
Resumen	iii
Abstract	iv
Nomenclatura	v
Índice de figuras	ix
Índice de tablas	xi
Presentación	xii
1. Marco Referencial	1
1.1. Descripción del problema	1
1.2. Delimitación del problema	1
1.3. Complejidad del problema	1
1.4. Antecedentes	2
1.5. Estado del Arte	6
1.6. Objetivos	15
1.6.1. Objetivo General	15
1.6.2. Objetivos Específicos	15
1.7. Alcances y Limitaciones	15
1.7.1. Alcances	15
1.7.2. Limitaciones	15
1.8. Justificación	16
1.9. Organización de la tesis	16
2. Marco Conceptual	17
2.1. Segmentación de imágenes	17
2.2. Aprendizaje Profundo	18
2.3. Redes Neuronales Convolucionales	19
2.4. Computación acelerada	30

2.5. Discusión	30
3. Experimentación y resultados	31
3.1. Entorno de desarrollo	31
3.1.1. Arquitectura de hardware	31
3.1.2. Arquitectura de software	31
3.2. Repositorios	32
3.3. Métricas	35
3.4. Plan de pruebas	36
3.5. Modelos implementados	39
3.5.1. Modelo para segmentación de imágenes binarias	39
3.5.2. Modelo para segmentación semántica	41
3.6. Pre-procesamiento del repositorio	46
3.7. Entrenamiento y pruebas del modelo empleando el repositorio BSDS500 . . .	54
3.8. Discusión	67
4. Conclusiones	69
4.1. Objetivos y alcances logrados	69
4.2. Productos	70
4.3. Aportaciones y mejoras realizadas	76
4.4. Conclusiones generales	78
4.5. Trabajos futuros	79
Referencias bibliográficas	80
Anexo A. Hiperparámetros de la red	87
Anexo B. Imágenes del BSDS500	89

Índice de figuras

1.1. Metodología general del algoritmo de superpíxeles y umbralización adaptativa.	7
1.2. Resultados de segmentación. (a) imagen original, (b) imagen segmentada basado en la umbralización y (c) imagen segmentada basado en la Colonia de Hormigas.	8
1.3. Segmentación basada en el crecimiento de regiones. (a), (c), (e) y (g) imágenes originales, (b), (d), (f) y (h) imágenes segmentadas.	9
2.1. Esquema general de una Red Neuronal Convolutiva con capas de deconvolución para segmentar imágenes (Noh <i>et al.</i> , 2015).	20
2.2. Capa de convolución (Eckroth, 2014).	20
2.3. Capa de submuestreo (<i>pooling</i>) (Deeplearning4j, 2016).	22
2.4. Capa de activación ReLU (Fergus, 2013).	23
2.5. Ejemplos de algunos métodos de preprocesamiento aplicados a una imagen del BSDS500.	24
2.6. Resultados obtenidos al escalar, desplazar y/o elevar los valores de entrada.	25
2.7. Resultado tras realizar la normalización a un mapa de características (Huang, 2015).	25
2.8. Resultado de aplicar capas de deconvolución y <i>unpooling</i> a un mapa de características generado por la Red Neuronal Convolutiva (Noh <i>et al.</i> , 2015).	26
2.9. Técnica de deserción (<i>Dropout</i>) (Heaton, 2015).	27
2.10. Recorte de mapas de características de acuerdo al tamaño de la imagen original.	28
2.11. Función MathWorks (2016) y regresión <i>softmax</i> que conforma la capa de <i>softmax</i> con pérdida (Jia y Shelhamer, 2014a).	29
2.12. Capa totalmente conectada (Behnke, 2016).	29
3.1. Estructura del repositorio PASCAL VOC 2012.	33
3.2. Estructura del repositorio BSDS500.	34
3.3. Modelo de Red Neuronal Convolutiva para segmentación de imágenes binarias.	40
3.4. Segmentación en una imagen binaria. (a) imagen de entrada y (b) imagen segmentada (visualización de la inferencia).	41
3.5. Modelo de Red Neuronal Convolutiva para segmentación semántica.	42
3.6. Ejemplo de segmentación semántica. (a) imagen de entrada, (b) visualización de la inferencia (imagen segmentada) y (c) etiquetas de las regiones segmentadas.	42
3.7. Dos imágenes del BSDS500 segmentadas con el modelo FCN-Alexnet.	43
3.8. Arquitectura del modelo FCN-8s diseñado en (García-Peraza <i>et al.</i> , 2016).	45
3.9. Transformación de datos a través de la red FCN-8s (Shan, 2016).	46

3.10. Ejemplos de segmentaciones reales o manuales de una imagen del repositorio BSDS500.	47
3.11. Ejemplos de algunas clases de objetos que aparecen tan sólo una vez en todo el repositorio de BSDS500.	47
3.12. Rendimiento de los métodos de Aprendizaje Profundo y clásicos en relación a la cantidad de información con las que son entrenadas (Ng, 2016), (Amaratunga, 2017) y (Amin, 2016).	48
3.13. Estructura de una imagen indexada: (a) imagen original, (b) etiqueta o segmentación verdadera, (c) ampliación de una sección de la etiqueta, (d) índices pertenecientes a esa sección ampliada y (e) mapa de colores al cual se relacionan los índices de la imagen etiquetada, cada índice también representa a una clase de objeto.	49
3.14. Ejemplo de una imagen a aplicarle las transformaciones geométricas para aumentar la información.	51
3.15. Muestra de imágenes del repositorio BSDS500 para entrenamiento.	53
3.16. Muestra de imágenes del repositorio BSDS500 para validación.	53
3.17. Muestra de imágenes del repositorio BSDS500 para prueba.	54
3.18. Proceso general seguido en la experimentación.	55
3.19. Gráfica comparativa entre los resultados obtenidos con la métrica PRI.	64
3.20. Gráfica comparativa entre los resultados obtenidos con la métrica VI.	65
3.21. Gráfica comparativa entre los resultados obtenidos con la métrica GCE.	65
3.22. Precisión por clases.	68
4.1. Primera página del artículo realizado para el CITID 2017.	72
4.2. Constancia de participación en el CITID 2017.	73
4.3. Póster presentado en la Escuela de Inteligencia Artificial y Robótica.	74
4.4. Reconocimiento por participar en la Escuela de Inteligencia Artificial y Robótica.	75

Índice de tablas

1.1. Información de artículos.	13
3.1. Lista de clases de objetos encontradas en el BSDS500	50
3.2. Sintonización de hiperparámetros de la arquitectura FCN-Alexnet	56
3.3. Sintonización de hiperparámetros originalmente para PASCAL VOC	56
3.4. Sintonización de hiperparámetros originalmente para SIFT FLOW	57
3.5. Comparación entre la sintonización de FCN-Alexnet y FCN-8s para PASCAL VOC y SIFT FLOW (con todas las imágenes de prueba)	58
3.6. Comparación entre la sintonización de FCN-Alexnet y FCN-8s para PASCAL VOC y SIFT FLOW (con una imagen de prueba)	59
3.7. Comparación entre diferentes sintonizaciones (con todas las imágenes de prueba)	60
3.8. Comparación entre diferentes sintonizaciones (con una imagen)	62
3.9. Comparación de resultados con sintonización para SIFT FLOW (con todas las imágenes de prueba)	63
4.1. Objetivos alcanzados.	69
4.2. Alcances logrados.	70
4.3. Mejoras con respecto a los dos modelos originales utilizados	77

Presentación

La segmentación de imágenes es una de las tareas más complejas del procesamiento digital de imágenes, debido a que existe una serie de factores que la dificultan. Por ejemplo: las variaciones de intensidad o color en las regiones, iluminación no uniforme, texturas parecidas, sombras y reflexión, objetos con bordes suavizados o difuminados con el fondo, traslapamiento de objetos y oclusión. Por consiguiente, la segmentación es una de las partes más importantes en la cadena de procesamiento de imágenes (Jähne, 2002), ya que determina el eventual éxito o fracaso de todo el proceso de análisis de la imagen (Martín, 2004). Además, los algoritmos desarrollados para segmentar imágenes son a menudo muy costosos computacionalmente.

Dado lo anterior, se propone la implementación de una Red Neuronal Convolutiva basada en técnicas de Aprendizaje Profundo para segmentar imágenes a color. Este tipo de redes tiene la ventaja de ser robusto a ciertas invarianzas, según lo menciona (LeCun *et al.*, 1998), (Zeiler y Fergus, 2013a) y (Girshick *et al.*, 2014). Además, existen diversas herramientas que facilitan su implementación y optimización al utilizarlas sobre una GPU.

Capítulo 1

Marco Referencial

1.1. Descripción del problema

La segmentación de una imagen a color es el proceso de agrupar sus píxeles por regiones, compartiendo cierta característica que los diferencien de las demás regiones. Este proceso se dificulta por la presencia de sombras, texturas y colores similares, iluminación no uniforme y oclusión. Ante esto, los métodos tradicionales de segmentación resultan poco prácticos, al tener que especificarles manualmente las características o al extraer características insuficientes o no muy útiles para realizar una correcta segmentación; también en muchas ocasiones, se requiere cambiar el espacio de color de las imágenes a segmentar. Para solucionar las problemáticas antes mencionadas, se propuso utilizar las Redes Neuronales Convolucionales, ya que éstas generan automáticamente características que se adecuan al tipo de imágenes a segmentar.

1.2. Delimitación del problema

El campo de Visión Artificial es muy amplio, por lo que se delimitó a la tarea de segmentar imágenes a color sin ruido, pero con las condiciones inherentes de las imágenes mencionadas anteriormente. Dentro de la segmentación de imágenes existen diversas formas de hacerlo, como son por regiones, fronteras y píxeles. Este trabajo de tesis se enfocó en la segmentación por píxeles, ya que de esta manera trabajan las redes utilizadas.

1.3. Complejidad del problema

La complejidad del problema radica en las condiciones inherentes de las imágenes con las que trabaja la Red Neuronal Convolutiva, como son las sombras, oclusión, texturas y colores similares, iluminación no uniforme, entre otras (Wang, 2015).

1.4. Antecedentes

Se revisaron los proyectos de tesis que fueron desarrollados en el CENIDET y que tienen alguna relación con las áreas de segmentación de imágenes digitales y Redes Neuronales Artificiales. Se muestra a continuación una breve descripción de las tesis revisadas.

1. Análisis y detección de anomalías en mamografías utilizando Redes Neuronales Convolucionales (Matuz, 2016)

En este trabajo se propuso implementar una Red Neuronal Convolutiva para extraer y analizar las características más significativas de anomalías en mamografías que puedan producir cáncer de mama. Se logró generar un sistema que abarque las etapas de procesamiento de imágenes mamográficas (preprocesamiento, segmentación, extracción de características y etapas de clasificación), que no sólo caracterizaba las anomalías existentes, sino que también, tenía la capacidad de clasificarlas con una buena precisión por medio del Aprendizaje Profundo que proporciona las Redes Convolucionales. Estas redes están en capacidad de poder ser implementadas en una GPU para optimizar recursos, y hacer factible trabajar con ellas, aunque no es obligatorio hacer uso de la GPU para implementarlas, pero sí se recomienda hacerlo.

2. Implementación y evaluación de Redes Neuronales Artificiales tipo “*Pulse-Coupled Neural Network*” (PCNN) aplicadas a Visión Artificial (Cardenas, 2015)

En este trabajo se implementaron los paradigmas pulsantes (PCNN, ICM y variantes) para segmentar imágenes, cuyos resultados fueron comparados con la segmentación realizada por un humano y con otros detectores de bordes (Canny y Sobel), utilizando las imágenes de Lena y Cameraman.

La detección de bordes se evaluó con la métrica PCM (*Pixel Correspondence Metric*) y PixToPix (pixel a pixel), mientras que la experimentación con regiones se evaluó con la entropía y entropía cruzada. Los resultados finales de esta tesis mostraron que los detectores de bordes tradicionales Canny y Sobel aún presentan mejores resultados que los algoritmos pulsantes al evaluar su similitud frente a los bordes trazados por el ser humano.

3. Localización de regiones para el reconocimiento de objetos en imágenes (Cervantes, 2014)

En esta investigación se abordó el problema de la localización de objetos de las categorías “silla” y “mesa” en imágenes de escenas no controladas. El reconocimiento de estos dos objetos incluye invarianza a escala, forma, textura, color, perspectiva, iluminación, falta de contraste con el fondo, las oclusiones parciales o la variabilidad de características entre objetos de una misma categoría. Se considera las regiones de una imagen obtenidas a partir

de múltiples segmentaciones jerárquicas, aplicando dos heurísticas, y considerando las áreas de las regiones adyacentes. El método propuesto en esta investigación, reportó una precisión de 96.02 % en la categoría de silla y un 97.20 % en la categoría de mesa. Esta tesis implementa un método llamado JeReEs, el cual mejoró en el porcentaje de localización a su estado del arte, para la base de prueba PASCAL VOC 2007.

4. Modelado del comportamiento de conducción de vehículos de transporte y/o carga aplicando redes neuronales y Visión Artificial (Sánchez, 2009)

En este proyecto se propuso una aproximación al modelado del comportamiento de conductores de camiones de transporte y/o carga, la cual consta de la utilización de un clasificador de comportamiento que, basado en Redes Neuronales Artificiales (RNAs), es alimentado con datos provenientes de ejercicios de conducción realizados en la cabina de simulación del proyecto CABINTEC, y es apoyado con un sistema de Visión Artificial (VA) que le proporciona información de la posición de las manos del conductor durante las sesiones simuladas.

El resultado principal que se obtuvo en este trabajo de investigación, fue el desarrollo de un sistema de adquisición y representación del conocimiento que permite, por medio de las RNA's entrenadas con información adquirida de expertos en seguridad vial, la clasificación automática del comportamiento de conductores mediante la reproducción visual de ejercicios monitoriados en la cabina de simulación.

5. Metodología para la estructuración y uso de conocimiento en segmentación de imágenes digitales (Cervantes, 2006)

Este trabajo consistió en establecer una metodología para la estructuración y uso del conocimiento en problemas de Visión Artificial, especialmente en la etapa de segmentación de imágenes, con lo cual se permite la reutilización, modularidad e independencia entre la base de conocimiento y los algoritmos utilizados. Este modelo trabaja con el conocimiento explícito que proporcionan los usuarios, y no con el conocimiento implícito de los algoritmos.

Una aportación que hizo este trabajo fue el desarrollo de un lenguaje de segmentación de imágenes basado en código, el cual posee una estructura dinámica, permitiendo de esta forma, la adición de nuevos elementos sin necesidad de modificar el intérprete de códigos. Así mismo, se creó una biblioteca de operadores de segmentación y manipulación de imágenes (14 operadores). También se desarrolló un prototipo de lenguaje para representar el conocimiento, basándose en los formalismos de reglas de producción y marcos; se crearon las bases de conocimiento para segmentar puntos, líneas, círculos y elipses.

6. Implementación de una Red Neuronal Holográfica para el control de un brazo robot articulado (Hernández, 2003)

Este tesis tuvo por objetivo implementar una Red Neuronal Holográfica y observar su desempeño en la identificación y el control de un brazo robótico. Para llevar a cabo este objetivo, se implementó un simulador dinámico con interfaz gráfica, de un manipulador planar de dos grados de libertad, con características tales como: inercia, gravedad, velocidad de procesamiento y fricción viscosa. Posteriormente, se realizó una comparación entre la Red *Backpropagation* y la Red Holográfica en la identificación de sistemas dinámicos. Por último, se evaluó el desempeño de la Red Holográfica en el control de un simulador planar con respecto a una Red “*Backpropagation*”, demostrando que los niveles de error de la Red Holográfica son menores a los de la Red *Backpropagation*.

Se concluyó que las Redes Neuronales Artificiales son una buena alternativa para el control de robots debido a que tienen la capacidad de: aprendizaje adaptativo, auto-organización, generalización, tolerancia a fallas, además que con sólo conocer las entradas al sistema y las salidas que estas generan, además, se puede obtener un modelo inverso de la planta basado en una Red Neuronal Artificial. En el caso de las Redes Neuronales Holográficas su principal ventaja está en el número de épocas utilizadas para converger, ya que casi usan dos o máximo cuatro, mientras que la Red *Backpropagation* utiliza miles de épocas en convergencia. Mientras, que su mayor desventaja se debe a que es más laboriosa que la implementación de los modelos más populares de redes neuronales, debido principalmente a que puede tener varias arquitecturas.

7. Comprensión de imágenes usando Redes Neuronales Artificiales recurrentes (Pérez, 2000)

En este trabajo se utilizó una Red Neuronal Artificial (RNA) con múltiples entradas, múltiples salidas y con autorecurrencia en las neuronas de la capa intermedia, todo esto para aumentar el radio de compresión de imágenes con respecto a un esquema muy semejante. La finalidad de este trabajo de investigación es evaluar el desempeño de la estructura de la RNA en compresión de imágenes. Para esto, se tomaron fotografías en formato BMP, se convirtieron en escala de grises de 8 bits y se las comprimieron con un sistema en *Matlab* basado en redes neuronales que se desarrolló como parte de este trabajo.

Al final, se buscó minimizar lo más posible las diferencias entre los patrones que entregaba la RNA y los patrones deseados. No se pudo conseguir y por ello los resultados que se obtuvieron no fueron los esperados. Se puede afirmar categóricamente que el esquema de compresión propuesto no es bueno, ya que sólo se usó un método de estimación paramétrica basado en el gradiente descendiente, en el que no se hace una búsqueda exhaustiva de parámetros.

8. Reconocimiento de patrones de fallas en generadores eléctricos empleando Redes Neuronales Artificiales (Rocha, 1999)

La presente tesis tuvo como primer objetivo desarrollar un método de preprocesamiento que reduzca información de los datos originales y mantenga la información esencial de los datos que entran al proceso de reducción. Este preprocesamiento trabaja con datos obtenidos del ICM++ como entrada y como resultado se obtiene vectores de elementos. Otro de los objetivos de este trabajo fue evaluar los paradigmas de RNAs *Backpropagation*, PHAF II (paradigma basado en la Red *Hamming*) y la red conocida como *Radial Basis Function* (RBF, paradigma basado en Funciones de Base Radial) con el objeto de detectar cuál de estos paradigmas se desempeñaba mejor en el problema de diagnóstico de fallas en generadores eléctricos. Y una vez cumplido el objetivo anterior, se desarrolló el sistema prototipo llamado “Diagnóstico de fallas en generadores eléctricos con redes neuronales (*DIFGERN*)”, el cual es capaz de reconocer patrones con fallas de generadores eléctricos. Para el *DIFGERN* se empleó la red neuronal *PHAF II*, paradigma de red neuronal que presentó el mejor desempeño. La red *PHAF II* presentó el menor error en el reconocimiento de patrones, es por eso que se empleó en este sistema.

9. Sistema de Visión Artificial para la verificación del llenado de recipientes no opacos utilizando Redes Neuronales Artificiales (Velarde, 1998)

En este proyecto se desarrolló un sistema de inspección visual automatizado, haciendo uso de las técnicas de tratamiento digital de imágenes y de reconocimiento de patrones, para la verificación del llenado de recipientes no opacos (botellas de refresco), detectando su forma, el nivel del líquido que contiene y la calidad del tapado de los mismo, para realizar una clasificación identificando aquellos con defectos y sin defectos, así como el desarrollo de un ejemplo que mostró su viabilidad. Este trabajo va dirigido a la industria dedicada al envasado de refrescos. El sistema de inspección visual automatizado creado tiene la capacidad de procesar imágenes de 105 columnas y 216 renglones de tamaño, con 4 bits por pixel; el proceso de obtener imágenes de este tamaño se realiza una vez capturada la imagen, mediante una cámara de video CCD. Este sistema se ejecuta bajo una plataforma del sistema operativo MS-DOS.

Cabe mencionar que, la finalidad de esta tesis no fue llevar a cabo una valoración de todas las técnicas existentes para segmentar y clasificar, sino poner a prueba la funcionalidad de las Redes Neuronales Artificiales para la identificación de objetos en escena, cuando se sabe que éstos pueden presentar ruido. El sistema respondió con un buen reconocimiento a las muestras presentadas, y en cuanto al tiempo de ejecución. El sistema pudo verificar hasta dos recipientes por segundo, lo que se puede mejorar con procesadores más rápidos.

10. Desarrollo de un sistema diagnosticador general de señales gráficas basado en la tecnología de redes neuronales (Ontiveros, 1995)

La aportación más importante del presente proyecto, la constituye su característica de generalidad lo cual significa que el sistema desarrollado se podría aplicar en cualquier proceso en donde se tenga un esquema similar. El mismo uso de las Redes Neuronales Artificiales constituyó otra de las aportaciones en este proyecto, debido a que esto implica una opción para el desarrollo de un sistema diagnosticador eficiente tanto en uso de la tecnología, como en tiempo de respuesta. Finalmente, con este trabajo se aportó la creación del modelo RNMap (Red Neuronal basada en el Mapeo de datos), cuyo funcionamiento, de acuerdo a una serie de pruebas realizadas, se consideró rápido y confiable.

Al término de este trabajo, se elaboró un sistema en C++, bajo el paradigma orientado a objetos. Este sistema es capaz de aprender y reconocer señales gráficas. A partir del análisis de las señales gráficas, el sistema pudo emitir distintos diagnósticos.

11. Discusión

Los trabajos analizados en esta sección se enfocan más a la clasificación de imágenes, como la tesis de (Matuz, 2016), hay otro trabajo como la tesis de (Cervantes, 2014) que se enfocó en la localización de regiones, así como para la comprensión de imágenes, como el trabajo de (Pérez, 2000). Mientras los que se encargan de la segmentación, como la tesis de (Cardenas, 2015) o el de (Cervantes, 2006), utilizan otro tipo de redes y métodos, por lo que este trabajo de tesis es el primero en utilizar las Redes Neuronales Convolucionales para segmentación de imágenes a color tomadas en ambientes no controlados, las cuales han dado resultados competitivos en esta área.

1.5. Estado del Arte

En esta sección se describen brevemente algunos de los artículos más actuales que tratan de la segmentación de imágenes a color con distintas técnicas, en otros más tratan de las Redes Neuronales Convolucionales, así como de algunos otros métodos de Aprendizaje Profundo para la tarea de segmentación de imágenes.

1. Segmentación de imagen rápida y efectiva por superpíxeles y umbralización adaptativa (Jiang y Ma, 2014)

En este documento se propuso una segmentación de imágenes rápida y efectiva por medio de la agrupación de un pequeño número de píxeles llamado superpíxeles, luego se fusionaron estos superpíxeles cuyas distancias son inferiores a un umbral adaptativo para conseguir las áreas segmentadas finales. La utilización de superpíxeles reduce el costo de cálculo, mientras que el umbral adaptativo pretende seleccionar una segmentación razonable de un conjunto de segmentaciones posibles con escalas jerárquicas.

Para los experimentos se utilizó el repositorio para segmentación de Berkeley BSDS500 y para medir el rendimiento de la segmentación realizada, se utilizaron tres métricas: Segmentación Cubierta (*Segmentation Covering*, SC), Índice probabilístico de Rand (*Probability Rand Index*, PRI) y Variación de Información (*Variation of Information*, VI). Se obtuvieron los siguientes resultados en promedio: 1.8 para la métrica SC, 1.0 para la métrica PRI y 5.1 para la métrica VI.

En la Figura 1.1 se puede apreciar un modelo general de cómo funciona el algoritmo de segmentación por superpíxeles y umbralización adaptativa, aplicada a una imagen del repositorio BSDS500.

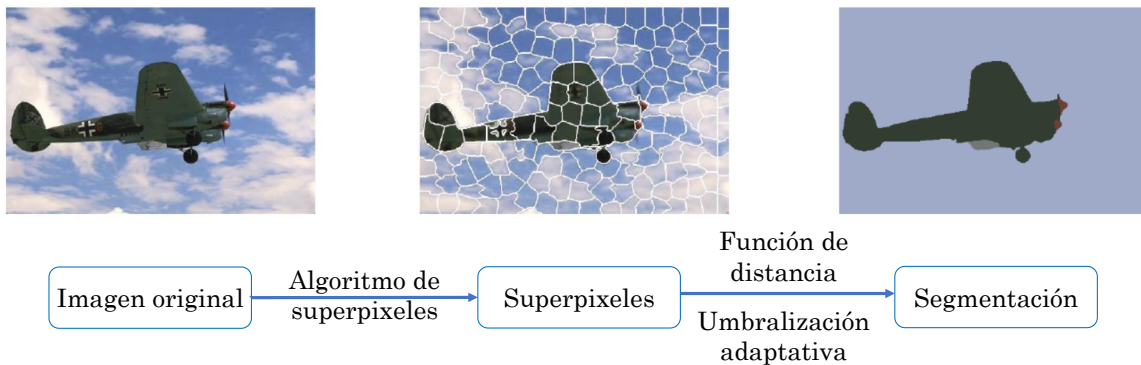


Figura 1.1. Metodología general del algoritmo de superpíxeles y umbralización adaptativa.

2. Segmentación de imagen usando características aleatorias (Bull *et al.*, 2015)

Este trabajo presentó un algoritmo para seleccionar características aleatorias a través de la detección comprimida para mejorar el rendimiento de los cortes normalizados en la segmentación de imágenes, reduciendo el tiempo computacional y uso de memoria, así como aumentando la precisión en las segmentaciones.

Este trabajo aportó tres mejoras con respecto a: las segmentaciones convencionales basadas en bordes y regiones (Canny y Otsu), la segmentación de imagen interactiva y la segmentación basada en su estado del arte mencionado. Los experimentos se realizaron usando el repositorio BSDS500, cuyas imágenes fueron segmentadas en tres espacios de colores distintos: en escala de grises, en el espacio de color RGB y en el espacio de color CIELab (*Commission Internationale d'Eclairage, Luminosity*, a =coordenadas rojo/verde y b =coordenadas amarillo/azul). El algoritmo propuesto fue programado tanto en Matlab como en C++ y se obtuvieron mejores segmentaciones que utilizando los cortes normalizados.

3. Segmentación de imágenes a color basado en el algoritmo de Colonia de Hormigas (Lü *et al.*, 2015)

En este artículo se implementó un sistema basado en la aplicación del algoritmo de Colonia de Hormigas para la segmentación de imágenes a color en tres etapas, el cual es fundamentado en los aspectos discretos de la imagen digital y la habilidad del agrupamiento difuso del algoritmo de Colonia de Hormigas. La primera parte consiste en extraer las características de una imagen considerando los valores RGB, el gradiente y el vecindario. La segunda parte es establecer el centro de la agrupación por medio de la combinación de estadística y selección artificial. Y la tercera parte es aplicar el algoritmo de Colonia de Hormigas para segmentar una imagen a color.

La segmentación para la Figura 1.2 se realizó en 28.59 segundos con el algoritmo de Colonia de Hormigas, mientras que el algoritmo de umbralización se llevó 0.14 segundos para segmentar la misma imagen. Estos resultados prueban que es mucho más rápido la segmentación de imágenes a color por medio del algoritmo de umbralización que por el algoritmo de colonia de hormigas, aunque se obtienen mejores resultados con este último método.

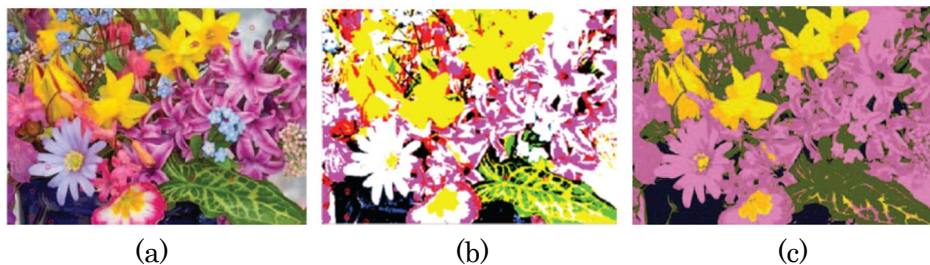


Figura 1.2. Resultados de segmentación. (a) imagen original, (b) imagen segmentada basado en la umbralización y (c) imagen segmentada basado en la Colonia de Hormigas.

4. Algoritmo de segmentación de imágenes a color basado en los canales RGB (Gothwal *et al.*, 2014)

Este artículo presentó una nueva técnica de segmentación de imágenes a color basada en el crecimiento de regiones. Las fases de selección del pixel semilla y el crecimiento de región fueron basadas en la medida de similitud en la conectividad de 8 vecinos. El crecimiento de regiones fue aplicado a cada uno de los tres canales de colores (rojo, verde y azul) separadamente, luego la matriz etiquetada combinada fue creada para obtener una imagen segmentada. El umbral fue utilizado para parar el proceso de crecimiento de regiones. También se utilizó un método de eliminación aleatoria de pixeles para fusionar pequeñas regiones con grandes regiones vecinas en la imagen segmentada. Se utilizó el repositorio proporcionado por Berkeley para segmentar imágenes a color. Las imágenes de la Figura 1.3 (a, c, e y g) se segmentaron en 8, 13, 25 y 15 regiones respectivamente (b, d, f y h), mientras que el tiempo de ejecución fue de 101, 423, 332.08 y 7.69 segundos

respectivamente; estos datos se pueden corroborar en la Tabla 1 del artículo de (Gothwal *et al.*, 2014). Por lo que se concluye, que el tiempo de ejecución aumenta con el incremento del número de regiones y con la complejidad de la imagen de entrada.

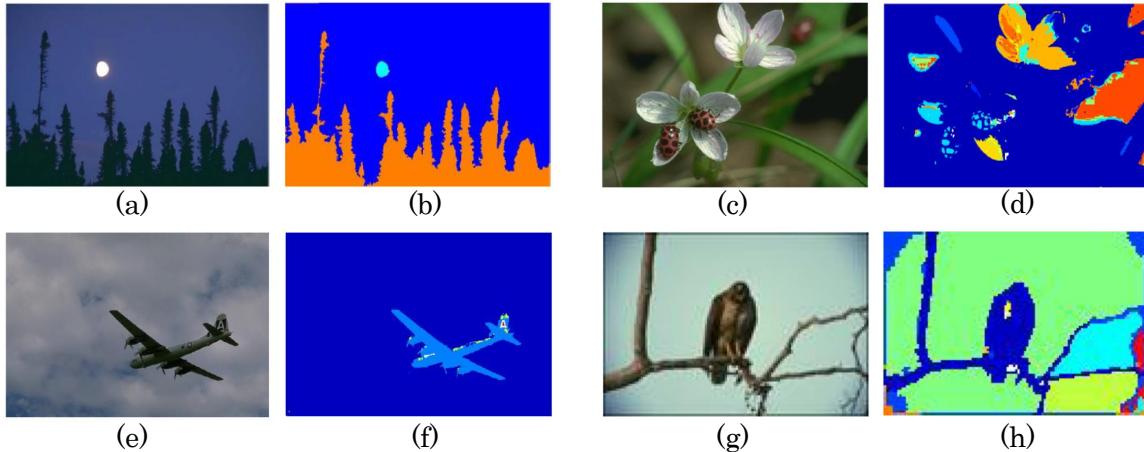


Figura 1.3. Segmentación basada en el crecimiento de regiones. (a), (c), (e) y (g) imágenes originales, (b), (d), (f) y (h) imágenes segmentadas.

5. Corte de Árbol para segmentación probabilística de imagen (Hu *et al.*, 2015)

En este trabajo se presentó el modelo generativo probabilístico llamado Corte de Árbol, para la segmentación de imágenes. Este modelo se basó en una representación de nivel medio de la imagen, llamado árbol de regiones, en donde las regiones se dividen en subregiones de forma recursiva hasta que se alcanzan superpíxeles (es una metodología encargada de reducir una imagen, teniendo en cuenta los contornos presentes en ésta, agrupando por zonas de color, y a estas agrupaciones se les llaman superpíxeles (Primo, 2011)). Dado el árbol de regiones, la segmentación de imagen se formaliza como cortes de muestreo en el árbol del modelo. El modelo de Corte del Árbol se puede ajustar para probar segmentaciones en una determinada escala de interés de entre muchas posibles segmentaciones a multiescala de imagen. Esto generaliza la noción común que debe haber sólo una segmentación correcta por imagen. Además, permite ir más allá de la evaluación estándar de una sola escala, donde el resultado de la segmentación para una imagen se promedia con los obtenidos en los 3 grupos de imágenes establecidos: grupo de imágenes con 1 a 8 segmentos llamado segmentación gruesa, grupo de 9 a 31 segmentos llamado segmentación media y grupo de 32 segmentos o más llamado segmentación fina; todo esto para llevar a cabo una evaluación a escala específica. Los resultados cuantitativos son comparables con los del método que conduce a GPB-owt-UCM, con la ventaja de producir una distribución a lo largo de todas las posibles segmentaciones consistentes del árbol de la imagen.

6. Generación de marcado automático para la segmentación por *Watershed* de imágenes naturales (Sigut *et al.*, 2014)

La transformada *Watershed* es un método de segmentación por cuencas o líneas divisorias. El método propuesto consiste principalmente de tres etapas: división del histograma, líneas divisorias marcadas y una etapa final de fusión. Para esta tarea se utilizó el espacio de color LAB. Este método evita la sobsegmentación.

El método propuesto fue probado con las imágenes naturales del repositorio BSDS500 mediante el código hecho en MATLAB. La calidad de las segmentaciones fue evaluada usando la Variación de Información (*Variation of Information*, VI), el Índice Probabilístico de Rand (*Probabilistic Rand Index*, PRI) y la Segmentación Cubierta (*Segmentation Covering*, SC). Se reportaron dos puntajes (*ODS* y *OIS*) para las métricas VI y PRI y tres puntajes (*ODS*, *OIS* y *Best*) para SC.

7. Agrupación Combinacional Multiescala (Arbeláez *et al.*, 2014)

Se propuso un enfoque unificado de la segmentación de imagen jerárquica de abajo hacia arriba y la generación de objetos candidatos para reconocimiento, llamado Agrupación Combinatoria Multiescala (*Multiscale Combinatorial Grouping*, MCG). Para esto, primero se desarrolló un algoritmo de cortes normalizados rápidos. Luego, se formuló un segmentador jerárquico de alto rendimiento que hace uso efectivo de imágenes en múltiples escalas. Finalmente, se presentó una estrategia de agrupación que combina las regiones multiescalas dentro de los objetos candidatos propuestos.

Para los experimentos se utilizaron los repositorios BSDS500 y PASCAL 2012. En los experimentos con el repositorio BSDS500 se hicieron segmentaciones en una escala y en multiescalas, tanto en las segmentaciones por bordes como por regiones; se lograron mejoras en la precisión de la segmentación conforme a las métricas SC (*Segmentation Covering*), PRI (*Probability Rand Index*), VI (*Variation of Information*).

8. Detección y segmentación simultánea (Hariharan *et al.*, 2014)

El objetivo de este trabajo fue detectar todas las instancias de una categoría en una imagen, y por cada instancia marcar los píxeles que pertenecen a ella. Al método propuesto que realiza esta tarea se le llama detección y segmentación simultánea (DSS). Este método salió del resultado de refinar las máscaras de las regiones obtenidas a partir del algoritmo C (éste método resulta al entrenar una única gran red neuronal con dos vías) y utilizar también una máquina de vector soporte, así como la Red Neuronal Convolutiva basada en región (*Region-based Convolutional Neural Network*, R-CNN).

Se entrenó todo el sistema con el repositorio PASCAL VOC 2012 sobre el entorno de trabajo de *Caffe*. Se evaluó el método propuesto, llamado C+ref, conforme a las métricas AP^r y AP_{vol}^r , donde se obtuvieron 49.7% y 41.4% respectivamente de efectividad al momento de segmentar 20 clases de objetos distintos. Posteriormente, se evaluó la precisión de la segmentación hecha por la red con la métrica de intersección sobre la unión (*intersection over union*, IU), dando como resultado 51.6% en promedio para el repositorio PASCAL VOC 2012, y con ésta métrica también se probó el repositorio PASCAL VOC 2011, resultando 52.6% en promedio.

9. Hipercolumnas para la segmentación de objetos y localización de puntos finos (Hariharan *et al.*, 2015)

En el presente artículo se propuso la implementación de hipercolumnas, las cuales conjunta la información de la última capa de las Redes Neuronales Convolucionales con las capas ocultas, para obtener tanto una representación de características, como una localización precisa. La hipercolumna se define como el vector de activaciones de todas las unidades de la red por encima de un pixel. Usando las hipercolumnas como descriptores de pixeles, se mostró resultados en tareas de localización de puntos finos para detección y segmentación simultánea.

Se utilizaron los repositorios VOC2009 y VOC2012, con los cuales se mejoraron los resultados mostrados en el estado del arte de este artículo, pasando de un promedio de 49.7% a 60.0% con la métrica mAP^r en 0.5 y 40.4 en promedio con la métrica mAP^r en 0.7. También se logró aumentar de 3.3 a 6.6 puntos con la métrica APK sobre los enfoques del estado del arte, en donde trabajaron sólo con las características de la última capa de la red.

10. Aprendizaje de una Red Deconvolucional para la segmentación semántica (Noh *et al.*, 2015)

En éste artículo se propuso un algoritmo de segmentación semántica mediante la conjunción de una Red Neuronal Convolutiva y otra Deconvolucional. La Red Neuronal Deconvolucional está compuesta por capas de *unpooling* y deconvolución, éstas hacen más grande el mapa de características y lo densifica, para llegar al tamaño y forma original pero ya segmentada por regiones.

Se aplicó un modelo de Red Neuronal Convolutiva de 16 capas con una más de clasificación, los Campos Aleatorios Condicionales (*Conditional Random Fields*, CRF) totalmente conectados, al final, ésto por la parte convolutiva; mientras que para la parte deconvolutiva se ensambló la Red Neuronal Deconvolucional con una Red Totalmente Convolutiva (*Fully Convolutional Network*, FCN). Se utilizó en el repositorio PASCAL VOC 2012 mejorando la precisión en aproximadamente 1% en promedio, en comparación con otros modelos existentes (hipercolumnas, MSRA-CFM, FCN8s, TTI-Zoomount-16 y DeepLab-CRF).

11. Discusión

Algunos de los artículos anteriores fueron de utilidad para comparar sus resultados con los que se obtuvieron en la presente tesis, debido a que muestran resultados cualitativos y cuantitativos de segmentaciones realizadas con el mismo repositorio a trabajar, el BSDS500 de la Universidad de Berkeley, a excepción de (Lü *et al.*, 2015), (Hariharan *et al.*, 2014) y (Hariharan *et al.*, 2015), los cuales trabajaron con otros repositorios que también contienen imágenes a color. Los dos últimos trabajos sirvieron de guías para saber de dónde partir, ya que implementaron una Red Neuronal Convolutiva para segmentar imágenes a color. La mayoría de los trabajos citados realizaron la prueba de rendimiento basándose en las métricas de Variación de Información (*Variation of Information*, VI), el Índice Aleatorio Probabilístico (*Probabilistic Rand Index*, PRI) y la Envoltura de Segmentación (*Segmentation Covering*, SC), las mismas métricas con las que se evaluaron cuantitativamente los resultados obtenidos en este trabajo.

Revisando estos artículos desde otros aspectos, se encontraron que algunos de ellos dan buenos resultados, pero son muy lentos en el procesamiento, como el algoritmo de (Gothwal *et al.*, 2014) con el crecimiento de regiones, y también aquellos basados en estadística, como el de (Lü *et al.*, 2015).

La mayoría de los artículos recientes acerca de las Redes Neuronales Convolucionales, segmentan imágenes médicas, microscópicas, satelitales o de interiores con información de profundidad, sin embargo, no lo hacen con imágenes a color en condiciones no controladas. Otros artículos, como los de (Hariharan *et al.*, 2014) y (Hariharan *et al.*, 2015), hacen la segmentación con el repositorio de PASCAL VOC 2012, pero no con el repositorio BSDS500, con el cual se trabajó en los experimentos de la presente tesis. Hasta el momento, no se han encontrado artículos recientes que segmenten imágenes del repositorio BSDS500 utilizando las Redes Neuronales Convolucionales.

En la Tabla 1.1 se presenta información relevante de los artículos antes citados.

Tabla 1.1: Información de artículos.

Artículo	Objetivo	Técnicas	Características	Resultados
Segmentación de imágenes rápida y efectiva por superpíxeles y umbralización adaptativa (Jiang y Ma, 2014)	Segmentar imágenes naturales.	Superpíxeles y umbralización adaptativa.	Se utilizó el repositorio BSDS500.	1.8 para la métrica SC, 1.0 para la métrica PRI y 5.1 para la métrica VI.
Segmentación de imagen usando características aleatorias (Bull <i>et al.</i> , 2015)	Segmentar imágenes a color.	Características aleatorias.	Se utilizó el repositorio BSDS500.	Se mejoró en las segmentaciones convencionales, de imagen interactiva y en las segmentaciones reportadas en el estado del arte.
Segmentación de imágenes a color basado en el algoritmo de colonia de hormigas (Lü <i>et al.</i> , 2015)	Segmentar imágenes a color por medio del algoritmo de colonia de hormigas.	Colonia de hormigas y estadística.	Utiliza la estadística para calcular los centros de las regiones.	Se llevó 26.30 y 28.59 segundos para segmentar las Figuras 2.3 (a) y (b) respectivamente.
Algoritmo de segmentación de imágenes a color usando la técnica de crecimiento de regiones. (Gothwal <i>et al.</i> , 2014)	Segmentación de imágenes a color usando la técnica de crecimiento de regiones.	Crecimiento de regiones, umbralización y un método de eliminación aleatoria de píxeles.	Se utilizó el repositorio llamado BSDS de Berkeley.	El tiempo de ejecución aumenta con el incremento del número de regiones y con la complejidad de la imagen.
Corte de árbol para segmentación probabilística de imagen (Hu <i>et al.</i> , 2015).	Segmentación de imágenes a color.	Árbol de regiones y superpíxeles.	Utilizan los repositorios BSDS300 y BSDS500.	El factor de escala óptimo para BSDS300 es de 0.0004, y para BSDS500 es de 0.0005.
Generación de marcado automático para la segmentación por <i>watershed</i> de imágenes naturales (Sigut <i>et al.</i> , 2014)	Segmentación de imágenes naturales.	Histogramas y segmentación por cuencas o líneas divisorias (<i>watershed</i>).	Utilizan el repositorio BSDS500 y el código está en MATLAB.	La calidad fue evaluada con las métricas VI (<i>Variation of Information</i>), PRI (<i>Probabilistic Rand Index</i>) y SC (<i>Segmentation Covariance</i>).

Tabla 1.1. Información de artículos (continuación).

Artículo	Objetivo	Técnicas	Características	Resultados
Agrupación combi- nacional multiescala (Arbeláez <i>et al.</i> , 2014)	Segmentar y clasificar imágenes.	Cortes normalizados rápidos, segmentador jerárquico y agrupador de regiones multiesca- las dentro de objetos candidatos propuestos.	Se utilizaron los reposi- torios BSDS500 y PAS- CAL 2012.	Se lograron mejoras en la precisión de la segmenta- ción conforme a las métri- cas SC (<i>Segmentation Co- vering</i>), PRI (<i>Probabilistic Rand Index</i>) y VI (<i>Variation of Information</i>).
Detección y segmenta- ción simultánea (Hariharan <i>et al.</i> , 2014)	Segmentar, extraer ca- racterísticas, clasificar regiones y refinar di- chas regiones para lo- calizar objetos.	Se empleó el modelo de red neuronal R-CNN.	Se utilizó el repositorio PASCAL VOC 2012 sobre el entorno de tra- bajo de <i>Caffe</i> .	Se obtuvieron 49.7% y 41.4% en promedio para las métricas AP^r y AP^r_{vol} respectivamente.
Hipercolumnas para la segmentación de objetos y localización de puntos finos (Hariharan <i>et al.</i> , 2015)	Segmentar y localizar puntos finos de imáge- nes mediante hiperco- lumnas.	Redes neuronales con- volucionales.	Se evaluó este traba- jo con los repositorios PASCAL VOC 2009 y 2012.	Se consiguió un promedio de 60.0% con la métrica AP^r .
Aprendizaje de una Red Deconvolucional para la segmentación semántica (Noh <i>et al.</i> , 2015)	Segmentar semántica- mente imágenes a co- lor.	Redes Neuronales Con- volucionales y Decon- volucionalmente, Redes To- talmente Convolucio- nales y Campos Alea- torios Condicionales.	Se utilizó el repositorio PASCAL VOC 2012 sobre el entorno de tra- bajo de <i>Caffe</i> .	Se mejoró en 1% en prome- dio sobre los modelos de hi- percolumna, MSRA-CFM, FCN8s, TTI-Zoomout-16 y DeepLab-CRF.

1.6. Objetivos

1.6.1. Objetivo General

Sintonizar una Red Neuronal Convolutiva para segmentar dos clases de objetos (persona y ave) del repositorio BSDS500 implementada en una Unidad de Procesamiento Gráfico.

1.6.2. Objetivos Específicos

- Comprender los conceptos básicos de segmentación de imágenes.
- Comprender los conceptos básicos de las Redes Neuronales Convolucionales.
- Comprender cómo procesa información una GPU.
- Implementar un modelo basado en una Red Neuronal Convolutiva que realice la segmentación de imágenes a color.
- Cuantificar la calidad de la segmentación y comparar los resultados con los obtenidos en otros trabajos mencionados en el estado del arte.

1.7. Alcances y Limitaciones

1.7.1. Alcances

- Se trabajó con imágenes a color del repositorio BSDS500 (Berkeley, 2011).
- El procesamiento de imágenes se realizó con una GPU NVIDIA con 640 núcleos.
- Se evaluó conforme a las métricas analizadas, proporcionadas por el Grupo de Visión por Computadora de la Universidad de Berkeley (Berkeley, 2013).
- Se comparó los resultados obtenidos con los resultados reportados en otros trabajos del estado del arte, sólo aquellos que mostraron la precisión de la segmentación en términos de las métricas analizadas en esta tesis.

1.7.2. Limitaciones

- Se trabajó únicamente con imágenes en el espacio de color RGB.
- No se utilizaron imágenes médicas, como lo son: MRI, PET, TAC, etc.
- Las imágenes que se segmentaron no tienen ruido.
- El trabajo se centró sólo en la segmentación de imágenes y no en su clasificación ni en el reconocimiento de toda la imagen.

1.8. Justificación

Se utilizaron las Redes Neuronales Convolucionales porque tienen propiedades de ser invariantes a traslación, escala y distorsión (son robustas) (LeCun, 1998, 2013), (Zeiler y Fergus, 2013a) y (Girshick *et al.*, 2014), debido a las características que se extraen en las capas de convolución (Johnson, 2016) (LeCun, 2013). Las Redes Neuronales Artificiales tienen implícitas la capacidad de ser paralelizables, por lo que esta propiedad hace factible implementarlas en una GPU para procesar las miles de características y demás valores que generan esta red al entrenarla, validarla y probarla con cientos de imágenes. Además, la implementación de este modelo podría utilizarse para, por ejemplo, automatizar robots o drones para localizar personas o fotografiar aves en su hábitat natural, entre otras tareas más.

1.9. Organización de la tesis

A continuación, se presenta la estructura de este documento:

En el Capítulo 2, se detallan los conceptos más importantes para comprender este trabajo de tesis, considerando brevemente los temas de segmentación de imágenes, Aprendizaje Profundo y computación acelerada, mientras que se explora más en el tema de Redes Neuronales Convolucionales, ya que este método abarca la mayor parte del presente trabajo.

En el Capítulo 3, se describe todo el proceso de experimentación, desde mencionar lo que se utilizó para dichas pruebas, pasando por el plan de pruebas y terminando por el pre-procesamiento del repositorio BSDS500 y la implementación de los modelos de CNNs, así como la muestra y análisis de los resultados obtenidos.

En el Capítulo 4, se muestra todo lo logrado en este trabajo, así como las conclusiones generales de toda la tesis y los trabajos futuros que podrían realizarse a partir de este trabajo.

Después, se presenta la bibliografía de toda la tesis en estilo APA (*American Psychological Association*, Asociación Americana de Psicología).

Por último, se presentan dos anexos. En el Anexo A se definen todos los parámetros que se afinaron de los modelos de Redes Neuronales Convolucionales que se emplearon en esta tesis. Y en el Anexo B vienen todas las imágenes que se emplearon para el entrenamiento, validación y prueba de los modelos mencionados.

Capítulo 2

Marco Conceptual

A continuación, se describen los conceptos que tienen mayor importancia en la elaboración de esta tesis y con los cuales se podrá entender mejor el presente trabajo.

2.1. Segmentación de imágenes

Para este contexto, se define la segmentación de imágenes como la partición de una imagen digital en un conjunto de regiones homogéneas con respecto a una o más características (como por ejemplo el brillo, el color o la textura) que corresponden al todo o partes de un objeto dentro de la imagen, lo cual ayuda a un posterior análisis o reconocimiento automático (BVLC, 2016) (Vélez *et al.*, 2003). En otras palabras, el proceso de segmentación se puede considerar como asignarle una etiqueta distintiva a los píxeles que conforman una imagen, de acuerdo al objeto al que pertenece. El etiquetado se hace de tal forma que los píxeles que pertenecen a una región homogénea con respecto a una o más características que compartan la misma etiqueta, y regiones de píxeles con características significativamente distintas estén identificados con etiquetas distintas (Duarte, 2006). El nivel al que se realiza la segmentación depende de la aplicación en particular (Martín, 2004); por ejemplo, sobre una fotografía grupal se puede segmentar el contorno de las personas (si sólo se desea saber cuántas personas se encuentran en la foto), o el rostro de las personas (si se desea identificar algunas características de esas personas, como género, edad, estado de humor, etc.), o algún logotipo en su ropa (si se desea saber a qué organización pertenecen).

Existen diversas técnicas de segmentación, las que se pueden agrupar en tres clases (Serna y Román, 2009):

Técnicas basadas en píxeles: Estos métodos pueden ser clasificados en locales (basadas en las propiedades de los píxeles y su entorno) o globales (basadas en la información global obtenida, por ejemplo, con el histograma de la imagen) (Múñoz, 2016).

Técnicas basadas en bordes: Consiste en dividir una imagen basándose en los cambios bruscos (discontinuidad) del nivel de intensidad de los píxeles. Los temas más importantes en la discontinuidad son: detección de puntos aislados, detección de líneas y detección de bordes o contornos de una imagen (Serna y Román, 2009).

Técnicas basadas en regiones: Se utilizan propiedades espaciales de una imagen para segmentarla por regiones, es decir, la imagen es dividida en regiones conexas, en donde cada región tiene propiedades distintas que las diferencian unas de otras. Los temas más importantes en continuidad son: crecimiento de regiones, división y fusión (Serna y Román, 2009).

2.2. Aprendizaje Profundo

El Aprendizaje Profundo es un campo de las ciencias computacionales que implica el uso de Redes Neuronales Profundas (con más de una capa oculta) para el aprendizaje de características a partir de un conjunto de datos. Esto implica que la profundidad de la arquitectura de una red se refiera al número de niveles que tenga (Morales, 2016). El número de capas ocultas requeridas para resolver un problema ha sido siempre un tema de investigación en sí mismo. Porque los problemas de clasificación más simple de baja complejidad requieren sólo una capa oculta (Kishore *et al.*, 2015).

Las ventajas del Aprendizaje Profundo son (Nvidia, 2015):

Robusto: No se necesita diseñar las características antes de tiempo, sino que las características son aprendidas automáticamente para ser las óptimas para la tarea en cuestión.

Generalizable: El mismo modelo de red neuronal se puede usar en diversas tareas.

Escalable: El rendimiento mejora con más datos, el método es enormemente paralelizable.

Existen tres formas de categorizar las técnicas de Aprendizaje Profundo por su arquitectura, las cuales son (Morales, 2016):

Redes profundas para aprendizaje no supervisado. Son aquellas en las que no se especifica la salida deseada (Heaton, 2015). En esta categoría están los modelos basados en energía, que incluyen los *auto-encoders* y las máquinas de Boltzmann profundas (*Deep Boltzmann Machines*, DBM) (Morales, 2016).

Redes profundas para aprendizaje supervisado. Son aquellas en las que se especifica la salida deseada. El entrenamiento supervisado enseña a la red neuronal a producir la salida deseada (Heaton, 2015). Algunos ejemplos de esta categoría son los Campos Aleatorios Condicionales (*Conditional Random Fields*, CRF) profundos y las Redes Neuronales Convolucionales (*Convolutional Neural Network*, CNN) (Morales, 2016).

Redes profundas híbridas. Usan los dos esquemas anteriores en la misma red pero en partes distintas (Morales, 2016).

2.3. Redes Neuronales Convolucionales

Este tipo de redes son una extensión del tradicional Perceptrón Multi-Capa (*Multi-Layer Perceptron*, MLP), en las cuales se incorporan conceptos como los campos receptivos locales, pesos y sesgos compartidos, así como el submuestreo espacial; con el objeto de dotarle a la red de invarianza a traslación, escala y distorsión (LeCun *et al.*, 1998). Éstos conceptos son descritos a continuación:

Campos receptivos locales. Es una región de la imagen de entrada del mismo tamaño que el kernel a aplicar, y con la cual se hace el proceso de convolución (Nielsen, 2015).

Pesos y sesgos compartidos. Todos los pesos y el sesgo serán los mismo para la primera capa oculta, lo que significa que todas las neuronas de esta capa detectarán la misma característica, si sólo es una capa oculta (Nielsen, 2015).

Submuestreo espacial o temporal. Simplifica la información que proviene de la capa de convolución (Nielsen, 2015).

En la Figura 2.1 se muestra un ejemplo de un esquema general de las Redes Neuronales Convolucionales utilizada para segmentar semánticamente imágenes a color, en donde se puede apreciar cómo están estructuradas en función a las capas que las conforman. En dicha figura se aprecia que entra a la red una imagen completa a color de 224x224 pixeles (en términos de la red se les llaman neuronas al valor de los pixeles) pasando primero por dos capas convolucionales con relleno de ceros a los lados. Luego de la primera capa de submuestreo (con el recuadro en rojo) disminuye las dimensiones a 112x112 neuronas seguido de otras dos capas de convolución con relleno; después vuelve a disminuir sus dimensiones sucesivamente hasta llegar a un tamaño de 7x7 neuronas, para después pasar a una capa totalmente conectada de 1x1 neurona pero con una dimensión de varias neuronas. Posteriormente, se vuelven aplicar las mismas operaciones pero de forma inversa, siendo las capas de deconvolución la inversa de las de convolución y las capas de *unpooling* las inversas de las de submuestreo, esto con el fin de regresar a su tamaño original la imagen procesada pero esta vez ya segmentada. También, en esta figura se puede apreciar qué tipo de características se obtienen en cada tramo de la red. De manera gráfica, se describe a continuación el funcionamiento de la misma.

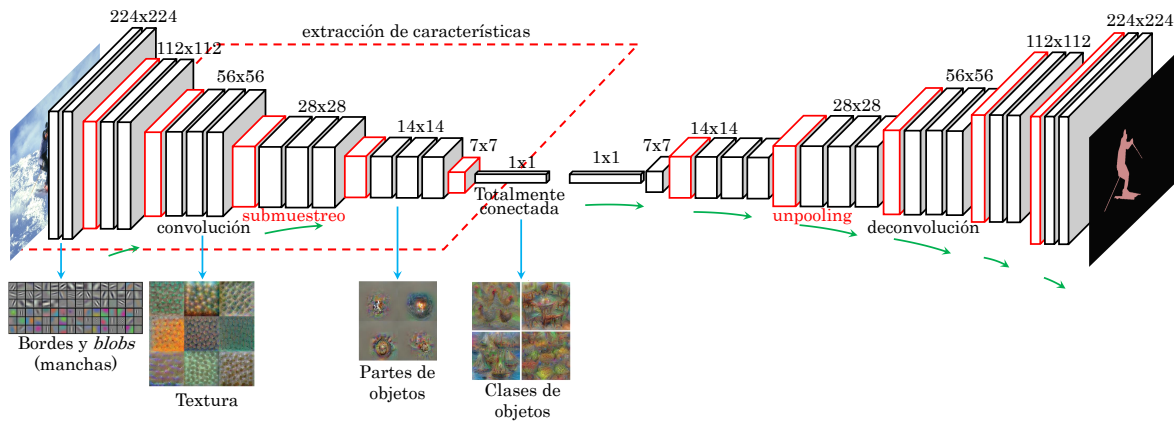


Figura 2.1. Esquema general de una Red Neuronal Convolutiva con capas de deconvolución para segmentar imágenes (Noh *et al.*, 2015).

Capa de convolución

El proceso de convolución en sí, consiste en multiplicar cada elemento de una máscara por el equivalente en el entorno o parte de la imagen en cuestión (campo receptivo local), luego sumar todos los productos, y asociarlos a un pixel (x,y) en específico, tal como se aprecia en la Figura 2.2. Una Red Neuronal Convolutiva usa las convoluciones para extraer características de regiones locales de una imagen de entrada.

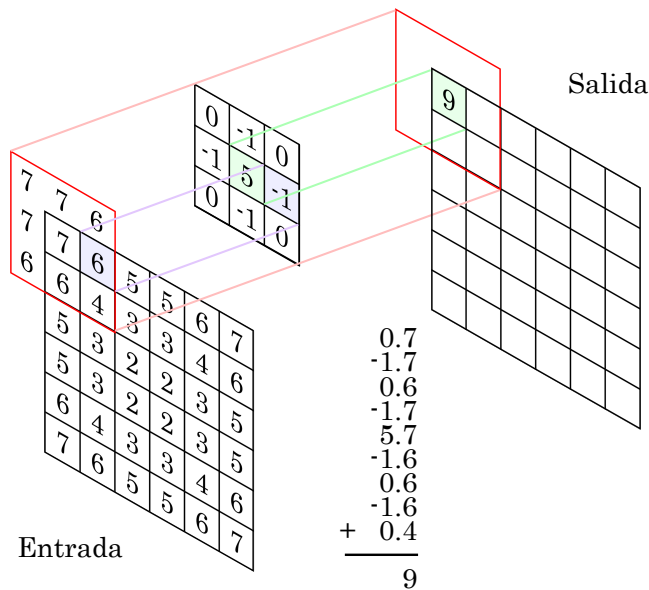


Figura 2.2. Capa de convolución (Eckroth, 2014).

La representación formal o matemática del proceso de convolución, es como se aprecia en la ecuación 2.1 (Vedaldi y Lenc, 2015):

$$y_{i''j''d''} = b_{d''} + \sum_{i'=1}^{H'} \sum_{j'=1}^{W'} \sum_{d'=1}^D k_{i'j'd'} \times x_{S_h(i''-1)+i'-P_h^-, S_w(j''-1)+j'-P_w^-, d', d''} \quad (2.1)$$

donde $y_{i''j''d''}$ es el resultado (mapa de características) de la operación de convolución, $b_{d''}$ es el valor del sesgo que se suma al resultado de la convolución entre el filtro $k_{i'j'd'}$ y las neuronas de entrada x (Vedaldi y Lenc, 2015) y (Ginzburg, 2014a). Por otro lado, P_h^- es el relleno (*padding*) de arriba, P_h^+ es el relleno de abajo, P_w^- es el relleno de la izquierda, P_w^+ es el relleno de la derecha, S_h es el paso o zancada en la altura y S_w es el paso en lo ancho. En el Algoritmo 1 se aprecia como se representa la convolución en pseudocódigo de forma más amplia, sin tomar en consideración el paso ni el relleno.

Algoritmo 1: Convolución.

```

1 imagen o mapa de características de entrada: x
2 núcleo o kernel de convolución: k
3 for  $d' \leftarrow 0; d' < D'; d'++$  do
4   for  $d \leftarrow 0; d < D; d++$  do
5     for  $h \leftarrow 0; h < H; h++$  do
6       for  $w \leftarrow 0; w < W; w++$  do
7         for  $h' \leftarrow 0; h' < H'; h'++$  do
8           for  $w' \leftarrow 0; w' < W'; w'++$  do
9              $x_L(h, w, d') \leftarrow x_L(h, w, d') + x_{L-1}(h+h', w+w', d) * k(h', w', d, d')$ 
10            end
11          end
12           $x_L(h, w, d') \leftarrow x_L(h, w, d') + b_{d'}$ 
13        end
14      end
15    end
16 end

```

Para calcular el tamaño de la salida de la convolución (mapa de activación) se necesita la ecuación 2.2 (Vedaldi y Lenc, 2015) y (Araujo, 2016a):

$$H'' = 1 + \frac{H + (2 \cdot pad) - H'}{S} \quad (2.2)$$

$$W'' = 1 + \frac{W + (2 \cdot pad) - W'}{S}$$

donde H y W es la altura y anchura de la entrada de la convolución respectivamente, H'' y W'' es la altura y anchura de la salida de la convolución, el *pad* es el relleno, S es el paso y H' y W' es la altura y anchura del filtro utilizado. Por otro lado, también se puede calcular el número de parámetros (pesos) utilizados y aprendidos para la operación de convolución con

la ecuación 2.3 (Araujo, 2016a):

$$num_p = ((H' \cdot W' \cdot D) + 1) \cdot D' \quad (2.3)$$

donde H' y W' es la altura y anchura del filtro respectivamente, D es la profundidad de la entrada, el 1 representa al valor del sesgo utilizado y D' es el número de filtros a utilizar. También se puede calcular la cantidad de memoria necesaria para la convolución, la cual se ve a continuación en la ecuación 2.4 (Araujo, 2016a):

$$mem = N \cdot D' \cdot H'' \cdot W'' \quad (2.4)$$

donde N es el tamaño de lote de entrada, es decir, el número de imágenes que procesará la capa al mismo tiempo, D' es el volumen de salida o en el caso de la convolución sería el número de filtros utilizados, H'' y W'' es la altura y anchura del mapa de activación de salida respectivamente.

Capa de submuestreo

La capa de submuestreo (*pooling*) se utiliza después de la capa de convolución para simplificar su información (Nielsen, 2015). En detalle, una capa de *pooling* toma el mapa de características que se produjo en la capa de convolución y lleva a cabo un condensado del mapa de características, al tomar pequeñas regiones de ésta (el tamaño de estas regiones se le llama extensión espacial) y realizar una operación sobre ella, por lo general se procede obteniendo el máximo valor de cada una de estas regiones (*Max-Pooling*). Un ejemplo de esta capa se puede ver en la Figura 2.3, en la cual se tiene cuatro cuadrantes en el mapa de características, y se reduce sacando el máximo valor de cada región (Gibiansky, 2014).

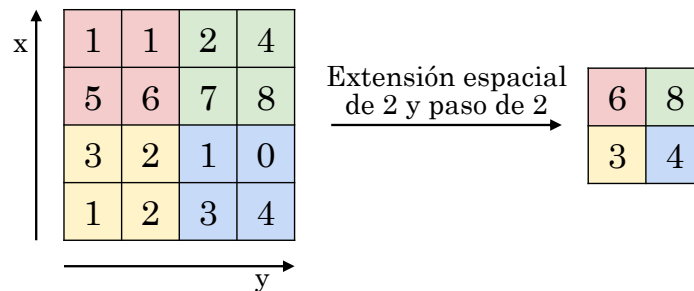


Figura 2.3. Capa de submuestreo (*pooling*) (Deeplearning4j, 2016).

La función matemática del *max-pooling* se puede expresar matemáticamente mediante la ecuación 2.5:

$$y_{i'',j'',d} = \max_{1 \leq i' \leq H', 1 \leq j' \leq W'} x_{i+i'-1, j+j'-1, d} \quad (2.5)$$

donde se toma el valor máximo del rango de 1 hasta el tamaño del alto (H') y 1 hasta el tamaño del ancho (W') del filtro (extensión espacial). Para calcular la salida de esta capa se pueden utilizar las siguientes ecuaciones para obtener el ancho, alto y profundidad resultantes,

respectivamente (Araujo, 2016b), como se muestra en la ecuación 2.6:

$$\begin{aligned} H'' &= \frac{H - H'}{S} + 1 \\ W'' &= \frac{W - W'}{S} + 1 \\ D'' &= D \end{aligned} \quad (2.6)$$

siendo W y H el ancho y la altura de la entrada, H' y W' es el tamaño de la altura y anchura del núcleo (*kernel*) de submuestreo (*pooling*), S el tamaño del paso y D y D'' es la profundidad de la entrada y salida respectivamente.

Capa de activación

Existe también una capa de activación en la Red Neuronal Convolutiva, la cual contiene una función de activación para las neuronas ocultas llamada unidad lineal rectificadora (*Rectified Linear Units*, ReLU), siendo la función que más se utiliza y que ha demostrado superioridad sobre otras. Se muestra su comportamiento en la Figura 2.4 (Heaton, 2015), en donde x es el valor de la neurona a evaluar, y $f(x)$ es el resultado de aplicar la función ReLU.

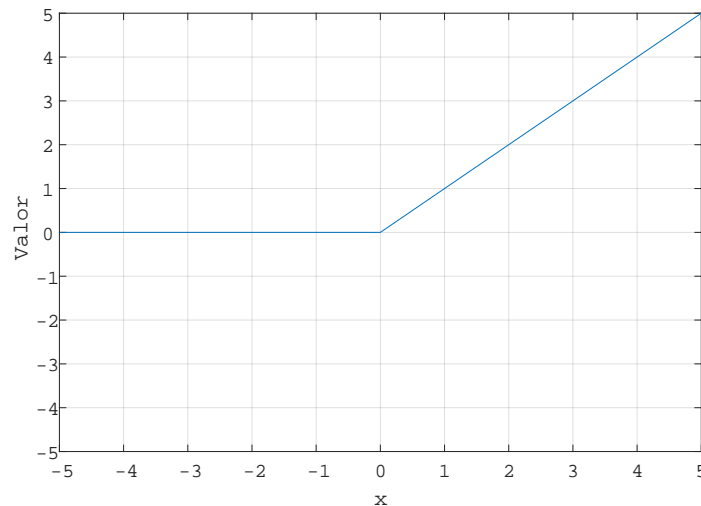


Figura 2.4. Capa de activación ReLU (Fergus, 2013).

Su representación formal es como se describe a continuación (Vedaldi y Lenc, 2015):

$$y_{ijd} = \max\{0, x_{ijd}\} \quad (2.7)$$

donde se elige el valor máximo entre el 0 y el valor de cada neurona de entrada.

Capa de datos

La capa de datos se encarga de introducir los datos, en este caso las imágenes, a la red neuronal, así como de analizar y optimizar las propiedades estadísticas de los datos entrantes para facilitar su paso a través de la Red Neuronal Convolutiva (Leónard, 2013). En esta capa se puede especificar si se desea aplicar algún método de preprocesamiento, tal como es la sustracción media, escalado, recorte aleatorio y reflejo, también se puede definir la ruta y tipo de la BD a cargar, así como el tamaño de lote (número de imágenes a procesar al mismo tiempo) (Jia y Shelhamer, 2014a). En la Figura 2.5 se puede apreciar una imagen antes y después de aplicar el método de la sustracción media.

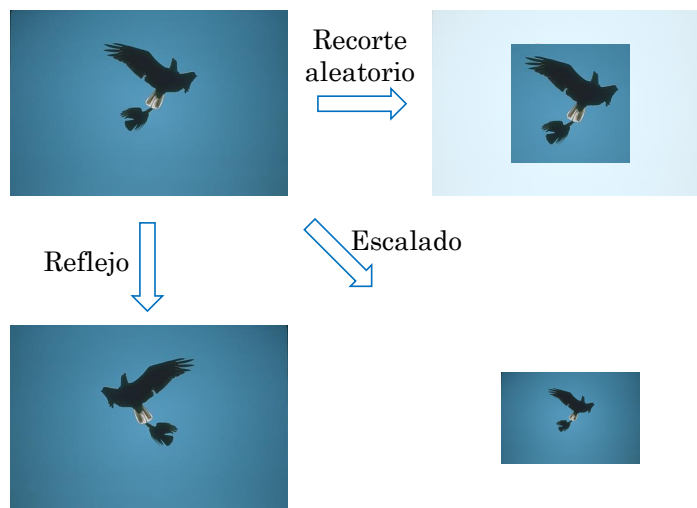


Figura 2.5. Ejemplos de algunos métodos de preprocesamiento aplicados a una imagen del BSDS500.

Capa de potencia

La capa de potencia sirve para normalizar los datos de entrada, en específico quitar números negativos y/o mover los valores de entrada a un rango determinado (Jia y Shelhamer, 2014a). En la Figura 2.6 se ejemplifica los resultados al mover los valores de entrada o expandirlos, por ejemplo, si se tiene una imagen con mucho brillo se puede desplazar sus valores de intensidad hacia bajo para oscurecer la imagen y en caso que se quiera aumentar su contraste solo sería cuestión de expandir los valores por el rango permitido de valores. Existen tres variables en esta capa que se pueden modificar para hacer lo antes dicho: la variable de potencia, escala y desplazamiento; por omisión, los valores de la potencia y de la escala están en 1 y el de desplazamiento está en 0.

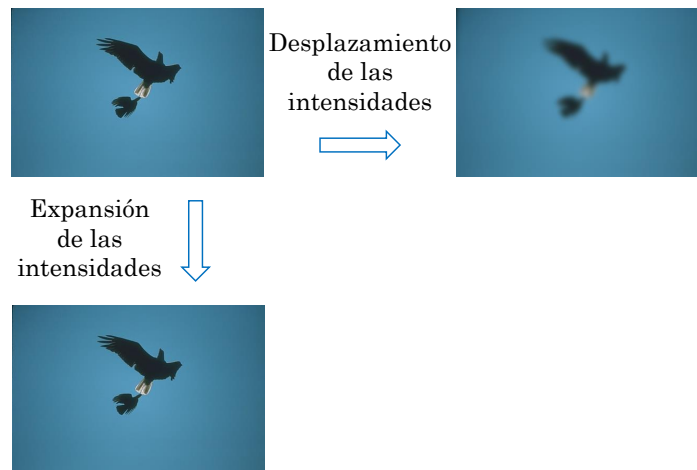


Figura 2.6. Resultados obtenidos al escalar, desplazar y/o elevar los valores de entrada.

De forma matemática se puede describir con la ecuación 2.8:

$$y_{ijd} = (shift + \alpha * x_{ijd})^{\beta} \quad (2.8)$$

donde *shift* es el valor de desplazamiento, α es un escalar y β es una potencia.

Capa de normalización

La capa de normalización de respuesta local (*Local Response Normalization*, LRN) (Jia y Shelhamer, 2014a), o también llamada normalización de contraste local (Jia y Shelhamer, 2014a), se utiliza para detectar características de alta frecuencia al tiempo que amortigua respuestas uniformemente grandes, semejándose a como el cerebro realiza la inhibición lateral (Jia y Shelhamer, 2014a). En la Figura 2.7 se puede observar el resultado que se obtiene al pasar un mapa de características por esta capa.

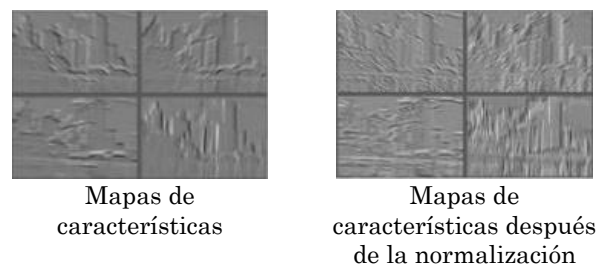


Figura 2.7. Resultado tras realizar la normalización a un mapa de características (Huang, 2015).

La expresión matemática de la función de normalización se muestra a continuación en la ecuación 2.9:

$$y_{hwd} = \frac{x_{hwd}}{\left(d'' + \left(\frac{\alpha}{crl} \right) \sum_{t \in G(d'')} x_{hwd}^2 \right)^{\beta}} \quad (2.9)$$

donde d'' corresponde al canal de salida, $G(d'') \subset \{1, 2, \dots, D\}$ es un conjunto correspondiente de los canales de entrada, crl es el tamaño de cada región local, h, w, d son los subíndices que pertenecen a fila, columna y canal respectivamente, mientras que α y β son hiperparámetros que son utilizados como escalares. Algo importante que hay que considerar, es que tanto la entrada x como la salida y son de las mismas dimensiones.

Capa de deconvolución

Para el proceso de segmentación se utiliza la capa llamada deconvolución. La cual, a partir del mapa de características generado, realiza la operación inversa de la convolución para regresarlo a la imagen original, pero ésta vez con las regiones segmentadas, tal como se muestra en la Figura 2.8, en donde se introduce una imagen, como la imagen (a), a la Red Neuronal Convolutiva, al final de ésta se genera un mapa de característica, la cual se le aplica una serie de operaciones de deconvolución (b), (d), (f), (h) y (j) y de *unpooling* (c), (e), (g) y (i) intercaladas. La operación de *unpooling* regresa los valores filtrados por el *pooling* a su posición original, mientras que la operación de deconvolución densifica los espacios dejados por la operación de *unpooling*.

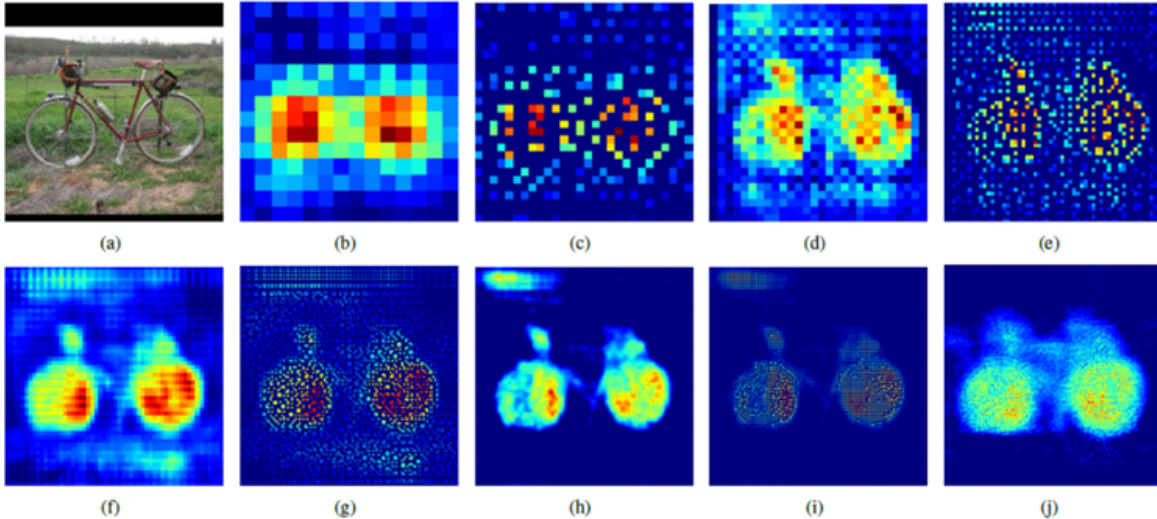


Figura 2.8. Resultado de aplicar capas de deconvolución y *unpooling* a un mapa de características generado por la Red Neuronal Convolutiva (Noh *et al.*, 2015).

Capa de deserción

La deserción (*dropout*) es una técnica de regularización para las redes neuronales que previenen o reducen el sobreajuste (Britz, 2015) y (Heaton, 2015). Para esto en algunos esquemas, se cambia aleatoriamente una fracción de neuronas a cero en cada iteración del entrenamiento (Britz, 2015), y en otros más se cambia a cero la salida de cada neurona oculta con probabilidad de 0.5 (Krizhevsky *et al.*, 2012) y (Ginzburg, 2014b). Las neuronas con sus conexiones desertadas (*dropped out*), tal como se muestra en la Figura 2.9, no contribuyen en el cálculo hacia adelante de la red, y tampoco participa en la propagación hacia atrás. Se debe de asignar una capa exclusivamente para realizar la deserción, sin mezclar nunca en una sola capa la convolución y la deserción (Heaton, 2015).

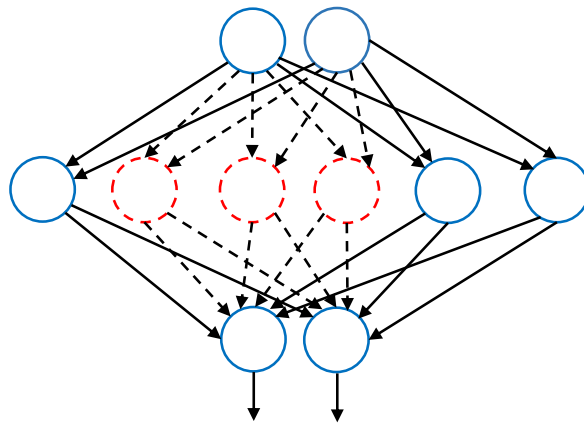


Figura 2.9. Técnica de deserción (*Dropout*) (Heaton, 2015).

La función de *dropout* se puede expresar formalmente de la siguiente manera:

$$\begin{aligned}
 r_j^i &\sim \text{Bernoulli}(p) \\
 x^l &= r^l * x^l \\
 z_i^{l+1} &= k_i^{(l+1)} x^l + b_i^{(l+1)} \\
 y_i^{(l+1)} &= f(z_i^{(l+1)})
 \end{aligned}
 \tag{2.10}$$

donde r_j^i es un conjunto de distribuciones según la función de Bernoulli con probabilidad de p , el índice l c $1, \dots, L$ es el índice de las L capas ocultas de la red y j es el índice de las neuronas de la capa l , luego se multiplica los valores de r^l por las neuronas de entrada $y(l)$, el resultado $x(l)$ se multiplica (se convoluciona) por el peso $k_i^{(l+1)}$ más el valor del sesgo $b_i^{(l+1)}$, por último se utiliza una función de activación f al cual se le pasa el resultado anterior z_i^{l+1} dando como resultado el valor de salida $y_i^{(l+1)}$.

Capa de recorte

La capa de recorte es la encargada de recortar la imagen de forma que mantenga la relación de aspecto (proporción entre el ancho y altura de una imagen) de la imagen original (Jia y Shelhamer, 2014a), como se puede ver en la Figura 2.10.

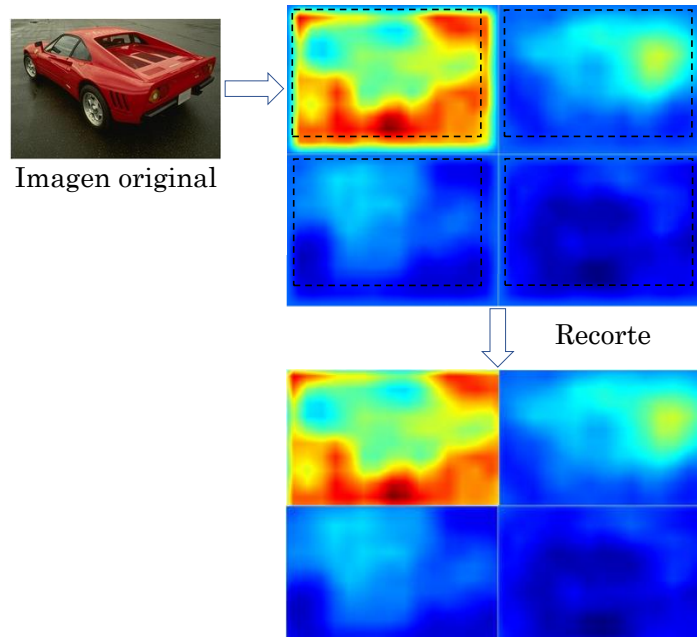


Figura 2.10. Recorte de mapas de características de acuerdo al tamaño de la imagen original.

Capa de softmax

La capa de *softmax* con pérdida compara los valores de salida con una etiqueta (objetivo al que se quiere llegar) asignando un costo para minizarlo, impulsando al aprendizaje mediante una función de pérdida (también conocida como error, costo o función objetivo) (Jia y Shelhamer, 2014b). A esto se reduce el aprendizaje de la red, a encontrar un ajuste de los pesos que minimice la función de pérdida. Esta última capa calcula la pérdida o regresión logística multinomial al realizar la función de *softmax*. Es conceptualmente idéntica a una capa de *softmax* seguida de una capa de pérdida logística multinomial, pero proporciona un gradiente numéricamente más estable (Jia y Shelhamer, 2014a). En la Figura 2.11, se puede observar que al pasar una serie de números por la función *softmax*, ésta normaliza los valores de entrada dando como resultado valores positivos flotantes en un rango de 0 a 1, también se aprecia que la capa de *softmax* está casi al final de la red.

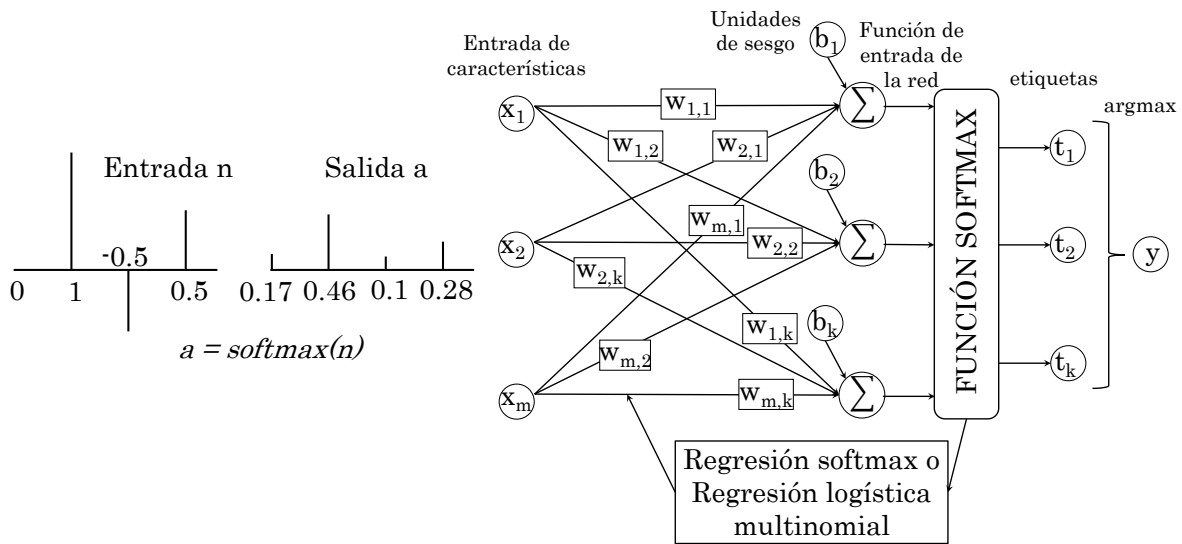


Figura 2.11. Función MathWorks (2016) y regresión *softmax* que conforma la capa de *softmax* con pérdida (Jia y Shelhamer, 2014a).

Capa totalmente conectada

En la capa totalmente conectada se introduce una Red Neuronal Totalmente Conectada (Xiaoyang, 2016), de la forma estándar, donde cada neurona de esta capa está conectada a todas las neuronas de la capa anterior, tal como se puede apreciar en la Figura 2.12, conformada por la capa de entrada (que contiene la imagen de entrada), seguida de 4 capas de convolución y submuestreo llamadas C1, S1, C2 y S2, terminando con 2 capas totalmente conectadas F3 y F4. Las activaciones totales de esta capa se pueden calcular con una multiplicación de matrices (Johnson, 2016).

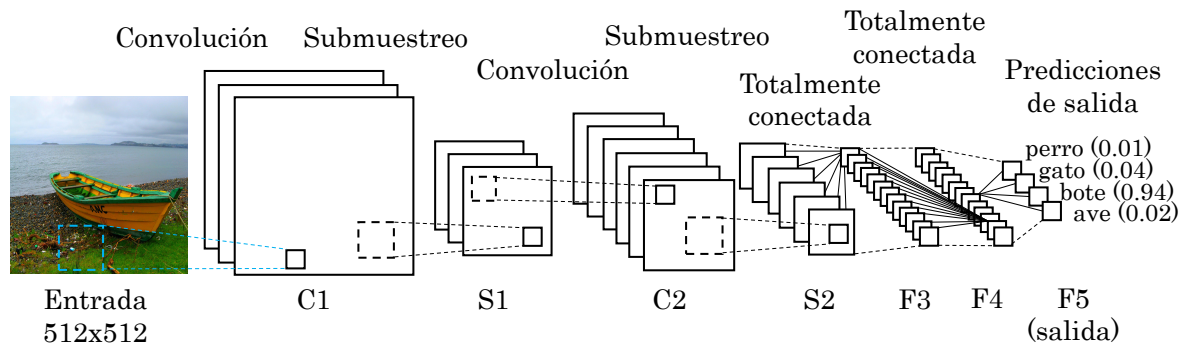


Figura 2.12. Capa totalmente conectada (Behnke, 2016).

Las Redes Neuronales Convolucionales deben contener al menos una combinación de capas de convolución, *pooling* (*Max-Pooling*) y totalmente conectadas (Britz, 2015). El orden y la repetición de las capas varían según el modelo de Red Neuronal Convolutivo que se esté implementando.

2.4. Computación acelerada

El cálculo acelerado en la Unidad de Procesamiento Gráfico (GPU) puede definirse como el uso de una GPU en combinación con una Unidad Central de Procesamiento (CPU), según NVIDIA, para acelerar aplicaciones de empresas, consumo, ingeniería, análisis y cálculo científico. NVIDIA lo introdujo en 2007 y, desde entonces, las GPU aceleradoras han pasado a instalarse en centros de datos muy importantes y de gran tamaño. Las GPUs aceleran las aplicaciones de plataformas diversas, desde automóviles hasta teléfonos móviles y tablets, drones y robots (NVIDIA, 2016a).

Para realizar el cálculo acelerado en la GPU, se trasladan las partes de la aplicación con mayor carga computacional a ésta y se deja el resto del código ejecutándose en la CPU.

Para enviarle código a la GPU se utiliza el *kit* de herramientas de CUDA. Esta es una arquitectura de cálculo paralelo de NVIDIA para C/C++/Fortran/Python, que aprovecha la gran potencia de la GPU para proporcionar un incremento extraordinario del rendimiento del sistema (NVIDIA, 2016).

2.5. Discusión

La tarea de segmentación de imágenes naturales a color es muy compleja, pero según el estado del arte, las Redes Neuronales Convolucionales son aptas para esa tarea, debido a que la extracción de características lo realiza de forma automática, sin tener que elegir manualmente entre muchas de las que existen. Además, como se vio a lo largo del estado del arte, estas redes se basan en el funcionamiento de la corteza visual primaria de los organismos vivos e imitan el funcionamiento de las redes neuronales biológicas, por lo que las hacen propicias para alcanzar o superar el rendimiento de las redes neuronales biológicas. Por último, se aclara que estas Redes Neuronales Artificiales no son una idea nueva, sin embargo, nunca tuvieron un gran éxito, porque se necesita una cantidad importante de recursos computacionales para su entrenamiento y ejecución, lo cual no había en el siglo XX, pero en la actualidad ya existen y se pueden utilizar, como son las Unidades de Procesamiento Gráfico (GPU), lo que hace factible su implementación en términos de tiempo, costo y capacidad de procesamiento.

Capítulo 3

Experimentación y resultados

En este capítulo se describe desde el entorno de desarrollo que se utilizó para las pruebas hasta el proceso completo que se llevó a cabo para poder experimentar con las arquitecturas FCN-8s y FCN-Alexnet, junto con el repositorio BSDS500. También se incluyen en este capítulo los resultados cualitativos y cuantitativos obtenidos a partir de dicha experimentación.

3.1. Entorno de desarrollo

En esta sección se presenta la arquitectura tecnológica que se utilizó para hacer el entrenamiento, validación y prueba con las Redes Neuronales Convolucionales.

3.1.1. Arquitectura de hardware

- Laptop DELL inspiron 7559.
- Arquitectura de 64 bits.
- Procesador Intel Core i5-6300HQ CPU @ 2.30GHz, 2301 Mhz, 4 procesadores principales y 4 procesadores lógicos.
- Memoria RAM de 8GB.
- Unidad de Procesamiento Gráfico (GPU) GeForce 960M.
- Memoria dedicada de video 4 GB.
- Numero de núcleos en GPU de 640.

3.1.2. Arquitectura de software

- Sistema operativo Linux en su distribución de XUbuntu (basado en Ubuntu), versión 14.04.5 LTS.
- Driver para la GPU versión 361.77.

- Toolkit de CUDA versión 8.0.27
- cuDNN versión 5.1
- Caffe versión 0.15.9
- Digits versión 4.1

3.2. Repositorios

El repositorio PASCAL VOC 2012 se utilizó mientras se trataba de implementar el modelo de red elegido y el repositorio BSDS500 se utilizó para conseguir los resultados finales con los cuales se pudiera evaluar y comparar con otros resultados del estado del arte. Debido a que este segundo repositorio se definió desde el principio de la tesis titulada “Sintonización de una Red Totalmente Conectada para segmentación de dos clases de objetos en imágenes” por las complejas propiedades que presentan sus imágenes para segmentarlas.

A continuación se describen brevemente los dos repositorios:

Clases de Objetos Visuales para el Análisis de Patrones, Modelado Estadístico y Aprendizaje Computacional (*Pattern Analysis, Statistical Modelling and Computational Learning Visual Object Classes*, PASCAL VOC 2012)

Para la tarea de segmentación por clases, este repositorio contiene 1464 imágenes para entrenamiento y 1449 imágenes para validación, con su respectivo tanto de etiquetas (segmentaciones verdaderas), como se alcanza a ver en la Figura 3.1. Las etiquetas son las segmentaciones realizadas manualmente, también llamadas segmentaciones reales o verdaderas; para este repositorio se cuenta con una sola etiqueta por imagen. Las imágenes de este repositorio tienen una profundidad de 24 bits ya que éstas son a color, de dimensiones variadas, en formato JPG, mientras que sus etiquetas son imágenes indexadas con una profundidad de 8 bits y están en formato PNG. Este repositorio es uno de los más utilizados para segmentación de imágenes mediante técnicas de Aprendizaje Profundo, ya que contiene todos los elementos necesarios para ser utilizados con estos tipos de técnicas; además, de tener una complejidad media en sus imágenes.

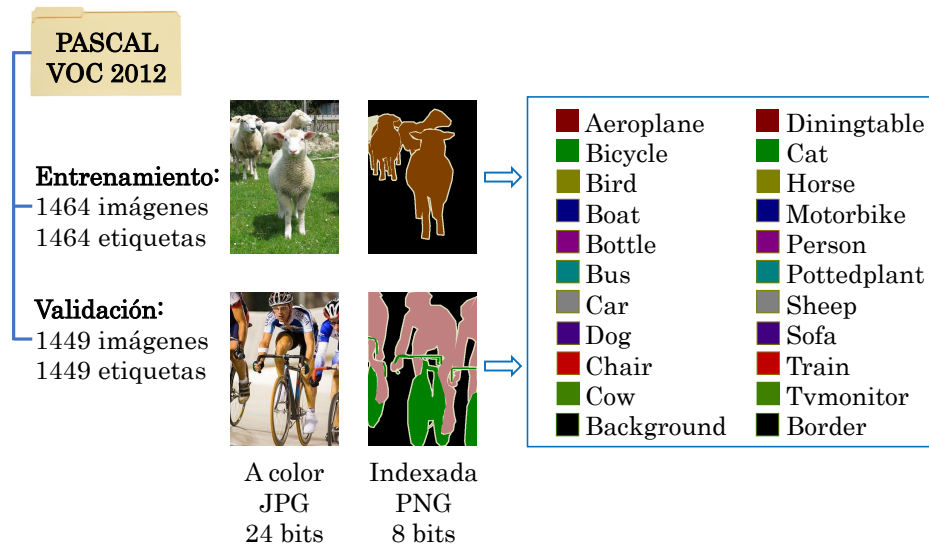


Figura 3.1. Estructura del repositorio PASCAL VOC 2012.

Este repositorio cuenta con 20 clases de objetos pre-establecidas más uno contando el fondo de la imagen, como también se observa en la Figura 3.1. Cabe mencionar que, el mapa de color de sus etiquetas es la que aceptan la mayoría de los modelos de Red Neuronal Convolutiva para entrenar y validar sus modelos. En la figura anterior se puede apreciar los colores pertenecientes a las 21 clases (20 objetos más el fondo) así como el color que se utilizó para los bordes de las regiones pertenecientes a las clases.

Base de Datos para Segmentación de Berkeley 500 (*Berkeley Segmentation Data Set 500, BSDS500*)

Este repositorio fue hecho principalmente para la tarea de segmentación y detección de contornos, la cual contiene 200 imágenes para entrenamiento, 100 para validación y 200 para pruebas. Las imágenes son de tipo naturales, ya que fueron tomadas en ambientes no controlados y no tienen pre-procesamiento alguno. Están a color con dimensiones de 481×321 píxeles, algunas imágenes tienen una orientación vertical y otras horizontal, con una profundidad de 24 bits, en formato JPG, mientras que sus etiquetas están en formato MAT (*MAT-file*, es un archivo de datos de MATLAB), contando con 4 a 9 etiquetas por imagen, tal como se ve en Figura 3.2, ya que cada etiqueta fue hecha por una persona que segmentó la imagen a su criterio. Las imágenes de este repositorio son particularmente difíciles de segmentar debido a la presencia de sombras, texturas y colores similares, iluminación no uniforme y oclusión. Además, este repositorio no fue originalmente diseñado para ser utilizado con técnicas de Aprendizaje Profundo, por lo que su adaptación será de interés y marcará la pauta para que otros repositorios que estén en la misma situación puedan ser igualmente utilizados con estos métodos y así aprovechar sus capacidades en cualquier otra área de interés científica.

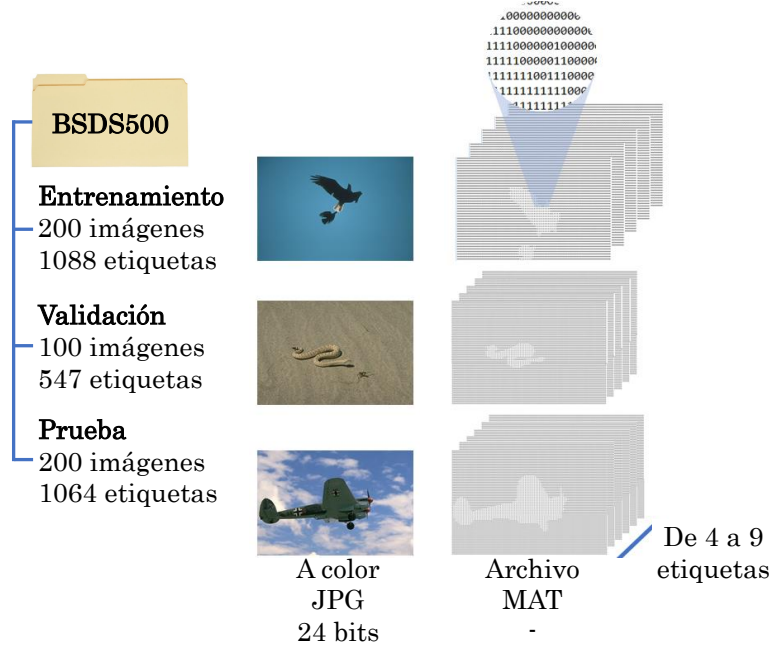


Figura 3.2. Estructura del repositorio BSDS500.

Este repositorio no cuentan con clases pre-establecidas, por lo que el usuario se encarga de definir sus propias clases de objetos, según le convenga. Se menciona también, que este repositorio no cuenta con un mapa de color propio, debido a que sus segmentaciones verdaderas se encuentran en archivos MAT y no en PNG como las del PASCAL VOC.

De los dos repositorios descritos, se utilizó el BSDS500 para los experimentos detallados más adelante, debido principalmente a tres causas:

- El BSDS500 fue el que se comprometió a utilizar desde la propuesta de tesis.
- Este repositorio tiene las etiquetas necesarias para poder utilizar las métricas PRI, VI y GCE.
- Las múltiples etiquetas por imagen que contiene el BSDS500 representa mejor la subjetividad del criterio humano al momento de definir la región correspondiente a una clase de objeto.

Mientras que, el repositorio PASCAL VOC 2012 se utilizó sólo para entrenar y validar los modelos originales FCN-Alexnet y FCN-8s y verificar que estuvieran aprendiendo correctamente, así como que se obtuvieran los resultados mostrados en otros proyectos de investigación.

3.3. Métricas

Las métricas que se emplearon para medir la precisión de la segmentación de los modelos empleados, fueron las proporcionadas por el mismo repositorio BSDS500, las cuales son: variación de información (*Variation of Information*), índice probabilístico de Rand (*Probabilistic Rand Index*) y error de consistencia global (*Global Consistency Error*, GCE).

Variación de información (*Variation of Information*, VI)

Fue propuesta originalmente en el trabajo de (Meilă, 2005) y utilizada en la tesis de (Stutz *et al.*, 2014) y en varios trabajos más, fue implementada inicialmente para comparación de agrupamientos. Esta métrica mide la distancia entre dos segmentos (en este caso entre dos segmentos, uno inferido y el otro verdadero o también llamado *ground-truth*) como la entropía condicional promedio de una segmentación dada la otra y, por tanto, mide aproximadamente la cantidad de aleatoriedad en una segmentación que no puede ser explicada por la otra. Esta métrica está compuesta por las ecuaciones 3.1 y 3.2:

$$H(S) = - \sum_{\substack{k=1 \\ S_k \in S}}^K \frac{|S_k|}{N} \log \left(\frac{|S_k|}{N} \right) \quad (3.1)$$

$$I(S, G) = \sum_{S_k \in S}^K \sum_{G_{k'} \in G}^{K'} \frac{|S_k \cap G_{k'}|}{N} \log \left(\frac{|S_k \cap G_{k'}|}{N} \frac{N}{|S_k|} \frac{N}{|G_{k'}|} \right) \quad (3.2)$$

donde H e I representan respectivamente las entropías de los grupos de las segmentaciones S (segmentación inferida) y G (segmentación verdadera), k y k' son los índices de los K y K' grupos existentes en las dos segmentaciones respectivamente y N es el número total de píxeles en la imagen. Ya juntando estas dos fórmulas, la métrica de la variación de información es definida como se muestra en la ecuación 3.3, que se observa a continuación:

$$VI(S, G) = H(S) + H(G) - 2I(S, G) \quad (3.3)$$

donde se obtienen las entropías para las segmentaciones S y G por separado y luego juntas. El resultado mientras más cercano sea a 0 es mejor, ya que indicaría que no hay incertidumbre.

Índice Probabilístico Rand (*Probabilistic Rand Index*, PRI)

Fue tomado del método creado por (Rand, 1971) y modificado por (Unnikrishnan y Hebert, 2005). Según los artículos de (Unnikrishnan y Hebert, 2005) y de (Stutz *et al.*, 2014), la métrica PRI cuenta la fracción de pares de píxeles cuyas etiquetas son consistentes entre la segmentación calculada y todas las segmentaciones verdaderas (segmentaciones manuales) disponibles en el repositorio. Para entender mejor la ecuación general de esta métrica, se presentan dos ecuaciones previas (ecuación 3.4 y ecuación 3.5), la primera calcula la probabilidad de la relación de un par de píxeles etiquetados \hat{l}_i y \hat{l}_j de las K segmentaciones verdaderas de cada imagen, mientras que la segunda calcula la probabilidad de indiferencia de ese mismo par de píxeles etiquetados \hat{l}_i y \hat{l}_j para las mismas K segmentaciones verdaderas de cada imagen:

$$\hat{P}(\hat{l}_i = \hat{l}_j) = \frac{1}{K} \sum_{k=1}^K I(l_i^{(k)} = l_j^{(k)}) \quad (3.4)$$

$$\hat{P}(\hat{l}_i \neq \hat{l}_j) = \frac{1}{K} \sum_{k=1}^K I(l_i^{(k)} \neq l_j^{(k)}) \quad (3.5)$$

Ya juntando las dos ecuaciones previas $\hat{P}(\hat{l}_i = \hat{l}_j)$ y $\hat{P}(\hat{l}_i \neq \hat{l}_j)$, la ecuación general de la métrica PRI queda como sigue en la ecuación 3.6:

$$PRI(S, G_{\{1\dots K\}}) = \frac{1}{\binom{N}{2}} \sum_{\substack{i,j \\ i \neq j}} [I(l_i = l_j)P(\hat{l}_i = \hat{l}_j) + I(l_i \neq l_j)P(\hat{l}_i \neq \hat{l}_j)] \quad (3.6)$$

donde S es la segmentación inferida por el modelo de Red Neuronal Convolutiva, $G_{\{1\dots K\}}$ son el conjunto de segmentaciones verdaderas hechas manualmente, N es el número de píxeles y sus salidas toman valores entre 0 y 1, 0 cuando no hay similitud entre la región segmentada y las regiones verdaderas y 1 cuando la región segmentada es idéntica a todas las regiones verdaderas.

Error de consistencia global (*Global Consistency Error, GCE*)

Propuesta por (Martin *et al.*, 2001), esta métrica mide hasta qué punto una segmentación puede ser vista como un refinamiento de la otra. Las segmentaciones que se relacionan de esta manera se les consideran consistentes, ya que podrían representar la misma imagen natural segmentada a diferentes escalas (Yang *et al.*, 2007). Esta métrica está definida de la siguiente forma:

$$E(S_1, S_2, p_i) = \frac{|R(S_1, p_i) \setminus R(S_2, p_i)|}{|R(S_1, p_i)|} \quad (3.7)$$

$$GCE(S_1, S_2) = \frac{1}{n} \min \left\{ \sum_i E(S_1, S_2, p_i), \sum_i E(S_2, S_1, p_i) \right\} \quad (3.8)$$

donde toma dos segmentaciones de entrada, S_1 que es la segmentación inferida por el modelo, S_2 es la segmentación verdadera y p_i es el pixel a evaluar, por lo tanto, $R(S, p_i)$ es el conjunto de píxeles correspondiente a la región en la segmentación S que contiene el pixel p_i . La salida que produce esta métrica es un valor real entre 0 y 1, donde el 0 significa que no hay error.

3.4. Plan de pruebas

Las pruebas realizadas en la sección de experimentación tienen la finalidad de medir el desempeño de las métricas PRI, VI y GCE al variar algunos hiperparámetros, como la tasa de aprendizaje base entre otros, así como probar la efectividad de algunos métodos coadyuvantes, como lo son la transferencia de conocimientos, la media del repositorio en el entrenamiento y

el aumento de la información. A continuación se detallan como se conformó las nueve pruebas contempladas para la experimentación:

Experimento 1: Su objetivo fue implementar y evaluar el rendimiento del modelo FCN-Alexnet con las imágenes del BSDS500.

- Se utilizaron 86 imágenes de entrenamiento (71 de personas y 15 de aves), 33 de validación (28 de personas y 5 de aves) y 84 de prueba (70 de personas y 14 de aves) del repositorio BSDS500.
- Se utilizó el modelo FCN-Alexnet.
- Este modelo fue creado inicialmente para el repositorio PASCAL VOC.
- Este modelo implementa el gradiente descendente estocástico.
- Se sintonizó el grado de aprendizaje base en $1e - 4$.
- La sintonización completa está definida en la Tabla 3.2 (sección 3.7, pág. 56).
- Se implementó la transferencia de conocimientos.

Experimento 2: Su objetivo fue implementar y evaluar el rendimiento del modelo FCN-8s con las imágenes del BSDS500. Se realizó la misma configuración que el experimento 1, a excepción de los siguientes puntos:

- Se utilizó el modelo FCN-8s.
- Se sintonizó el grado de aprendizaje base en $1e - 14$.
- La sintonización completa está definida en la Tabla 3.3 (sección 3.7, pág. 56).

Experimento 3: Su objetivo fue implementar y evaluar el rendimiento del modelo FCN-8s con las imágenes del BSDS500. Se realizó la misma configuración que el experimento 2, a excepción de los siguientes puntos:

- Este modelo fue creado inicialmente para el repositorio SIFT FLOW.
- Se sintonizó el grado de aprendizaje base en $1e - 12$.
- La sintonización completa está definida en la Tabla 3.4 (sección 3.7, pag. 57).

Experimento 4: Su objetivo fue evaluar el rendimiento del modelo FCN-8s con un grado de aprendizaje base en $1e - 16$. Se realizó la misma configuración que el experimento 3, a excepción de los siguientes puntos:

- Se sintonizó el grado de aprendizaje base en $1e - 16$.
- La sintonización completa es la misma que aparece en la Tabla 3.4 (sección 3.7, pag. 57) a excepción del grado de aprendizaje base.

Experimento 5: Su objetivo fue evaluar el rendimiento del modelo FCN-8s con un grado de aprendizaje base en $1e - 10$. Se realizó la misma configuración que el experimento 3, a excepción de los siguientes puntos:

- Se sintonizó el grado de aprendizaje base en $1e - 10$.
- La sintonización completa es la misma que aparece en la Tabla 3.4 (sección 3.7, pag. 57) a excepción del grado de aprendizaje base.

Experimento 6: Su objetivo fue evaluar el rendimiento del modelo FCN-8s con un grado de aprendizaje base gradual empezando desde $1e - 12$ hasta $1e - 16$. Se realizó la misma configuración que el experimento 3, a excepción de los siguientes puntos:

- Se sintonizó el grado de aprendizaje base desde $1e - 12$ hasta $1e - 16$.
- La sintonización completa es la misma que aparece en la Tabla 3.4 (sección 3.7, pag. 57) a excepción del grado de aprendizaje base.

Experimento 7: Su objetivo fue evaluar el rendimiento del modelo FCN-8s sin realizar la transferencia de conocimientos. Se realizó la misma configuración que el experimento 4, a excepción de los siguientes puntos:

- La sintonización completa es la misma que aparece en la Tabla 3.4 (sección 3.7, pag. 57) a excepción del grado de aprendizaje base.
- No se implementó la transferencia de conocimientos.

Experimento 8: Su objetivo fue evaluar el rendimiento del modelo FCN-8s al aumentar la información del BSDS500. Se realizó la misma configuración que el experimento 4, a excepción de los siguientes puntos:

- Se utilizaron 1,032 imágenes de entrenamiento (852 de personas y 180 de aves), 396 de validación (336 de personas y 60) y 84 de prueba (70 de personas y 14 de aves) del repositorio BSDS500. Este aumento del número de imágenes se debió a que por cada imagen se crearon otras 12 imágenes distintas de la misma.
- La sintonización completa es la misma que aparece en la Tabla 3.4 (sección 3.7, pag. 57) a excepción del grado de aprendizaje base.
- Se realizó el aumento de la información al aplicar 12 transformaciones geométricas a las imágenes de entrenamiento y validación del BSDS500. En la sección 3.6 se mencionan todas las transformaciones realizadas.

Experimento 9: Su objetivo fue evaluar el rendimiento del modelo FCN-8s al trabajar con la media del BSDS500. Se realizó la misma configuración que el experimento 4, a excepción de los siguientes puntos:

- La sintonización completa es la misma que aparece en la Tabla 3.4 (sección 3.7, pag. 57) a excepción del grado de aprendizaje base.
- Se utilizó la media de las imágenes de entrenamiento del BSDS500.

En los experimentos realizados con los modelos originales, el modelo FCN-8s alcanzó una mejor precisión en las tres métricas tomadas en cuenta (PRI, VI y GCE) que el modelo FCN-Alexnet, por lo que el resto de los experimentos se realizó con el primer modelo mencionado, modificando algunos hiperparámetros para lograr una mejor precisión.

3.5. Modelos implementados

Con el fin de determinar cuál modelo era el indicado para lograr el objetivo general, se utilizaron diferentes arquitecturas de Red Neuronal Convolutiva, las cuales se describen a continuación:

3.5.1. Modelo para segmentación de imágenes binarias

Este modelo realiza una segmentación en imágenes binarias (en blanco y negro) con diferentes tipos de triángulos (equilátero, isósceles y escaleno) en diferentes medidas, posiciones y ángulos. Fue entrenado con 9000 imágenes de triángulos sintéticos creados aleatoriamente desde Python, validado con otras 1000 imágenes y probado con una imagen que contiene varios triángulos. Como se puede apreciar en la Figura 3.3, la red está compuesta por 2 capas de convolución, 2 de activación con la función ReLU, 2 de datos, 1 de deconvolución y 1 de pérdida.

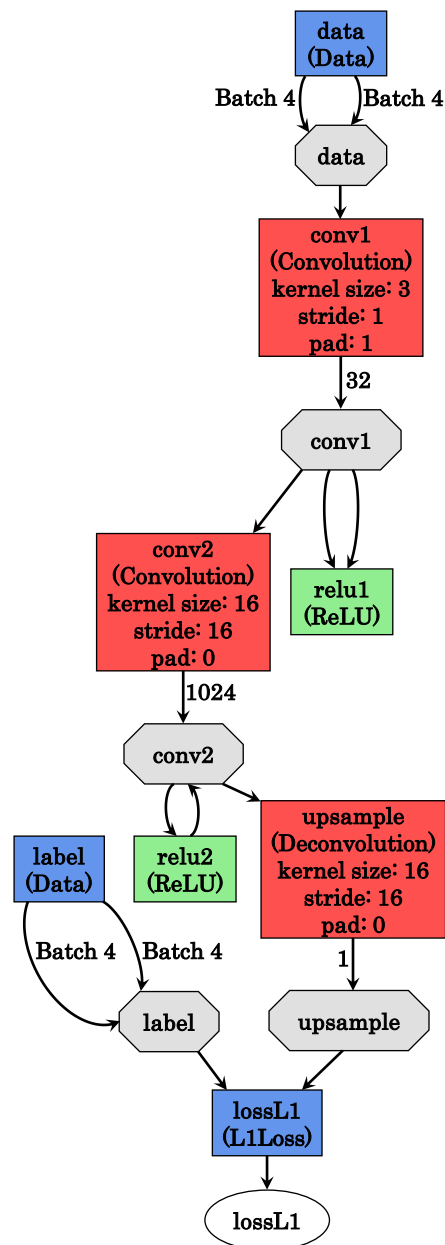


Figura 3.3. Modelo de Red Neuronal Convolutiva para segmentación de imágenes binarias.

En la Figura 3.4.(a) se puede ver una imagen con varios triángulos con la que fue probada el modelo y la cual fue segmentada, dando como resultado la Figura 3.4.(b), en donde se puede apreciar que fueron segmentados correctamente los triángulos, aunque con pequeñas regiones aparentemente grises entre algunos triángulos, debido al porcentaje de error existente durante el entrenamiento, que hasta cierto punto es conveniente para evitar el sobreajuste o sobreentrenamiento.

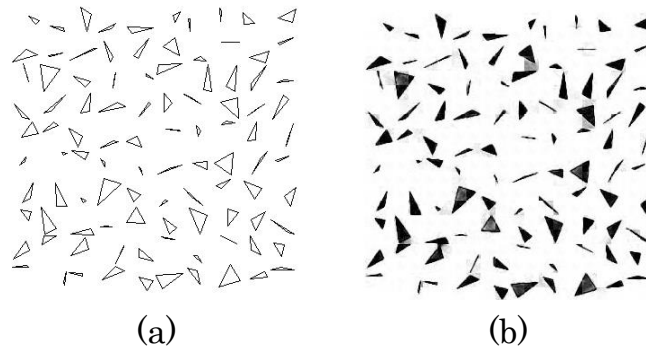


Figura 3.4. Segmentación en una imagen binaria. (a) imagen de entrada y (b) imagen segmentada (visualización de la inferencia).

Después de implementar este modelo y analizar sus resultados, se trató de ajustarlo para poder ser entrenado y probado con las imágenes del BSDS500, pero al final sólo mostró como inferencias imágenes en negro, resultando incapaz de segmentar las imágenes del repositorio BSDS500 debido a que intrínsecamente éstas tienen tres canales (RGB) y al convertirlas a imágenes binarias pierden muchas de sus características, que son esenciales, para poder realizar una segmentación correcta, por lo que se decidió ya no utilizarlo más adelante.

3.5.2. Modelo para segmentación semántica

Dado que había sólo modelos para segmentación semántica en el proyecto de (Shelhamer *et al.*, 2017), cuyo proyecto ha sido muy importante debido a la precisión que han alcanzado sus modelos, se probó dos de los modelos disponibles, el FCN-Alexnet y el FCN-8s. Cabe mencionar que por segmentación semántica se entiende la tarea de dividir una imagen en regiones que delinean objetos significativos y se etiquetan esas regiones con una clase de objeto que representa (Ikeuchi, 2014). Este tipo de segmentación se utiliza cuando se quiere conocer a qué objeto pertenece cada región segmentada.

FCN-Alexnet

El primer modelo de segmentación semántica que se implementó fue el propuesto por (NVIDIA, 2016b), el cual realiza una segmentación semántica de imágenes a color. Se tomó este modelo como venía entrenado y validado por (NVIDIA, 2016b), para luego ser probado con el repositorio BSDS500, esto con el fin de corroborar que se obtuvieran los resultados reportados. Como se puede apreciar en la Figura 3.5, la red está compuesta por 8 capas de convolución, 3 capas de max-pooling, 7 de activación con la función ReLU, 2 de deserción, 1 de potencia, 2 de normalización, 2 de datos, 1 de recorte, 1 de softmax con pérdida y 1 de deconvolución.

En la Figura 3.6 se muestra un ejemplo de segmentación semántica realizada con este modelo, en donde a la región de azul se le asigna la etiqueta de bote (embarcación) y a la región negra se le asigna la etiqueta de fondo.

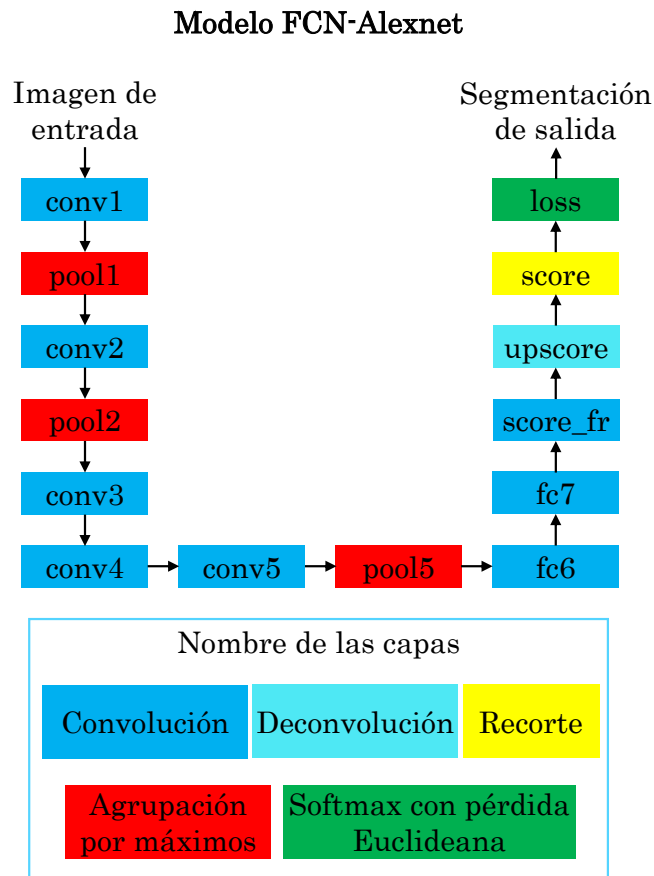


Figura 3.5. Modelo de Red Neuronal Convolutiva para segmentación semántica.

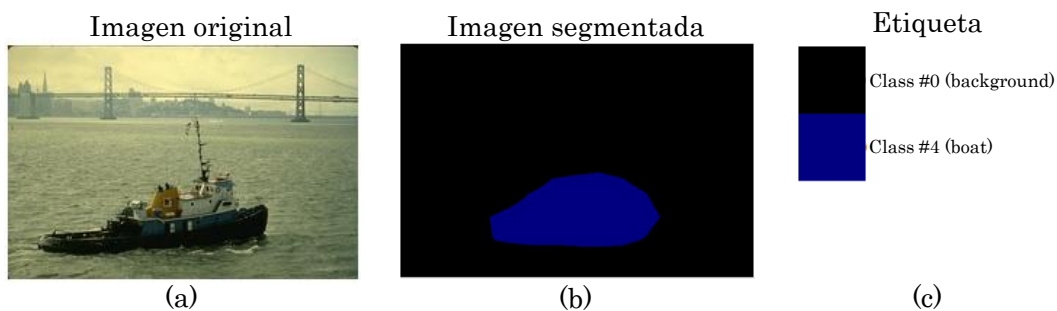


Figura 3.6. Ejemplo de segmentación semántica. (a) imagen de entrada, (b) visualización de la inferencia (imagen segmentada) y (c) etiquetas de las regiones segmentadas.

Como se aprecia en la figura anterior, la segmentación realizada es bastante burda, ya que se pierden los detalles finos de la embarcación que aparece en la imagen. Después, se prosiguió a entrenar y validar otro modelo con la misma arquitectura (FCN-Alexnet), pero ahora con las imágenes del repositorio BSDS500, para después probarlos con este mismo repositorio. En la Figura 3.7 se muestran dos imágenes segmentadas con este modelo.

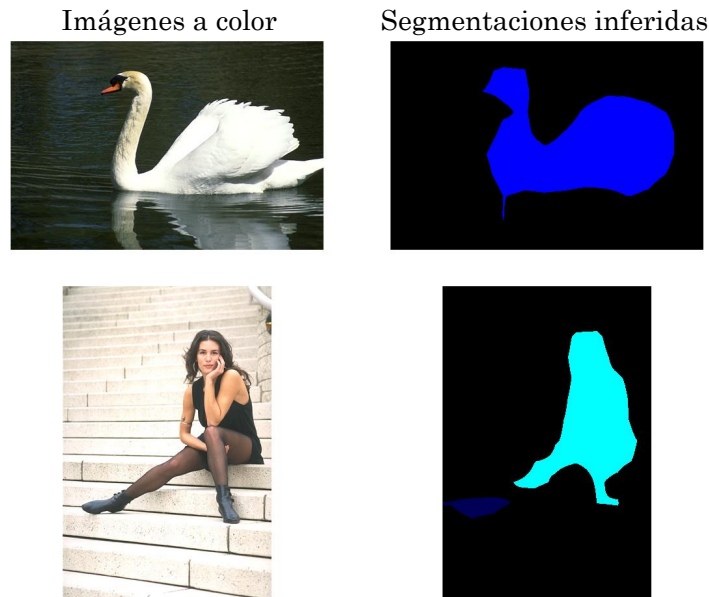


Figura 3.7. Dos imágenes del BSDS500 segmentadas con el modelo FCN-Alexnet.

Para entrenar este modelo se utilizó lo que es la transferencia de conocimiento, el cual consistió en utilizar los pesos con información significativa de la red VGG ya entrenada con el repositorio ILSVRC (Long *et al.*, 2015) para sembrar esos mismos pesos en otra red sin entrenar (Gutstein *et al.*, 2008). Esto con el fin que sea más rápido el entrenamiento, transfiriendo los mapas de características de bajo y medio nivel y ajustando solo los de alto nivel (Gutstein *et al.*, 2008). También se ocupa la transferencia de conocimiento cuando el repositorio de entrenamiento es menor a 10,000 imágenes, como en el artículo de (Oquab *et al.*, 2014) y (Gutstein *et al.*, 2008), para que se obtenga un buen entrenamiento.

FCN-8s

El modelo elegido para implementarlo fue la Red Totalmente Convolutiva de paso 8 (*Fully Convolutional Network*, FCN-8s), propuesto por (Long *et al.*, 2015), debido a su alta precisión en la segmentación de imágenes alcanzada en otros trabajos del estado del arte. La segmentación que realiza este modelo es más precisa que los otros dos modelos de Red Totalmente Convolutiva, el FCN-32s y FCN-16s, debido a que realiza un número de pasos más pequeño, esto significa que las características obtenidas con ella pertenecen a regiones de la imagen más pequeñas, por consiguiente son más precisas. Aunque tampoco se recomienda que sea un paso muy pequeño, ya que dejaría de aprender las características de los píxeles vecinos y esto ocasionaría la obtención de características inadecuadas para la segmentación de imágenes.

Este tipo de red acepta de entrada imágenes de tamaño arbitrario, debido a que cuenta en casi la totalidad de sus capas con funciones de convolución, las cuales realizan ese proceso por pequeñas secciones de la imagen (llamados campos receptivos locales) hasta abarcar toda

la imagen (Long *et al.*, 2015). En esto se diferencia de las Redes Totalmente Conectadas, las que sirven para clasificación, que ellas al contar con capas de Perceptrón Multi-Capa, están obligadas a recibir imágenes de tamaño fijo que concuerden con las capas de Perceptrón Multi-Capa (llamadas también totalmente conectadas), lo que les hace perder adaptabilidad.

El modelo FCN-8s se deriva de los modelos AlexNet, VGG-16 y GoogLeNet, fue pre-entrenado originalmente por (Long *et al.*, 2015) con el repositorio ILSVRC, ajustando los pesos restantes con el repositorio PASCAL VOC.

A continuación, en la Figura 3.8, se puede apreciar el modelo de la red FCN-8s, en donde se puede ver que la imagen entra por la capa de convolución (CONV_1), y que al pasar por las siete capas existentes de convolución y submuestreo se va perdiendo información al irse disminuyendo los mapas de características de la imagen de entrada; posteriormente se implementa la capa de deconvolución (UPSAMPLE_32) para regresar la imagen a su tamaño original pero ya con sus píxeles etiquetados, aunque aquí surge el inconveniente de que los mapas de características obtenidos de las capas anteriores perdieron detalles finos al ser sub-muestreados varias veces. Es por ello que después se realizan otras dos deconvoluciones (UPSAMPLE_16 y UPSAMPLE_8) pero esta vez tomando en consideración las características obtenidas de capas anteriores (Pool_4 y Pool_3 respectivamente) en donde todavía no se perdían tanto los detalles pequeños. Al final de la red, se obtiene una imagen del mismo tamaño que la imagen de entrada, con los objetos contenidos en ella casi de la misma forma que en la imagen original, pero esta vez ya etiquetados cada uno de sus píxeles con un color que representa a una clase de objeto en específico.

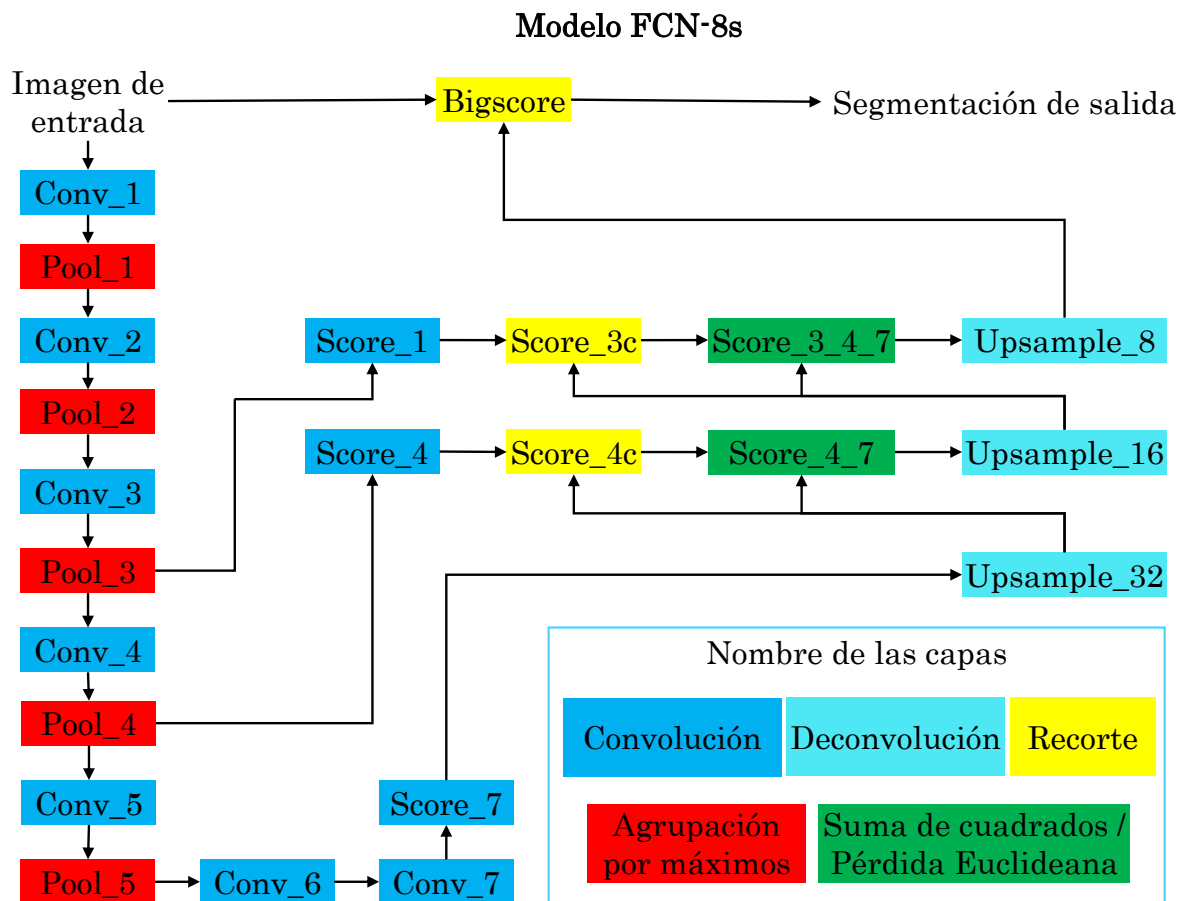


Figura 3.8. Arquitectura del modelo FCN-8s diseñado en (Garcia-Peraza *et al.*, 2016).

En la Figura 3.9 se puede observar un ejemplo de cómo van cambiando las dimensiones de una imagen a través de la red FCN-8s. En el ejemplo entra a la red una imagen de 3 canales (RGB) y de 500×500 píxeles, pasa primero por dos capas de convolución (recuadro en azul), luego por la capa de agrupamiento (recuadro en verde), así lo hace por 5 veces disminuyendo los mapas de características a un tamaño de 22×22 . Posteriormente, le siguen otras 3 capas más de convolución lo que disminuye un poco más las dimensiones a 16×16 y sobretodo disminuye la profundidad a 21, ya que 21 son los posibles objetos encontrados. Después se realiza la deconvolución aumentando la información en anchura y altura a 75%, luego vuelve a realizar otra deconvolución pero ahora sumando (recuadro en amarillo) los resultados de la primera deconvolución más los datos obtenidos de la cuarta capa, luego vuelve a realizar una tercera deconvolución con paso de 8. A partir de los datos de la segunda deconvolución más los datos obtenidos de la tercera capa, dando como resultado una matriz de $568 \times 568 \times 21$ y como hay datos sobrantes lo recorta (recuadro en gris) para obtener una matriz de dimensiones iguales que la imagen de entrada, la cual sería la imagen ya segmentada.

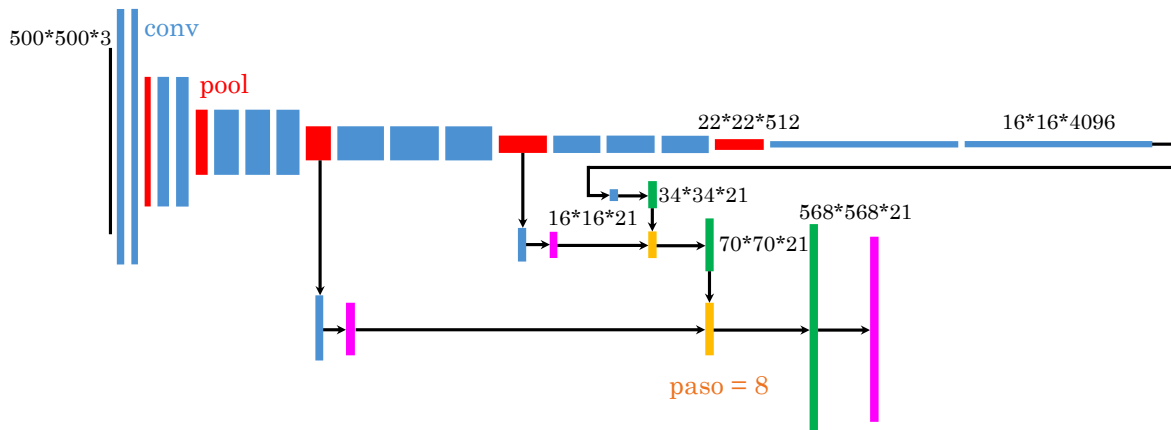


Figura 3.9. Transformación de datos a través de la red FCN-8s (Shan, 2016).

Este modelo fue implementado sobre una tarjeta gráfica NVIDIA GeForce GTX 960M, con 640 núcleos y 4 GB de memoria de video dedicada.

3.6. Pre-procesamiento del repositorio

Primero, se tuvo que preparar el repositorio BSDS500 para poder ser utilizada por el modelo FCN-8s, ya que este repositorio presenta una serie de problemáticas que se enuncian a continuación:

- Sus etiquetas no pueden ser utilizadas como se encuentran en el repositorio original para el entrenamiento ni validación, debido a que el entorno de trabajo llamado Caffe, donde se implementó estas redes, no soporta este tipo de archivos para entrenar ni validar. Además, cada imagen tiene asociado un archivo MAT, que cuenta con cuatro a nueve segmentaciones (*ground truth*) distintas por imagen, tal como se puede ver en la Figura 3.10, causando así un conflicto al no saber cuál segmentación utilizar.
- Las etiquetas al ser modificadas en PAINT u otro editor de imágenes, adquirieron un mapa de colores distinto al que contiene las etiquetas provistas por el repositorio de PASCAL VOC y con el cual no se pudo entrenar directamente en CAFFE, ya que marcaba error.
- Las clases de objetos que se muestran en sus imágenes son muy variadas, además que la frecuencia de aparición de éstas también varía mucho, pudiendo aparecer tan solo una vez en todo el repositorio, tal como las imágenes que aparecen en la Figura 3.11.
- Cuenta con pocas imágenes, con lo cual hace que el modelo no alcance una buena precisión en la segmentación, al verificar sus resultados a simple vista o comparando sus resultados con los de otros trabajos similares. Esta afirmación concuerda con el artículo de (Ng, 2016), el cual menciona que mientras más se nutra de información a los métodos de Aprendizaje Profundo, el rendimiento de éstos aumentará significativamente, como se

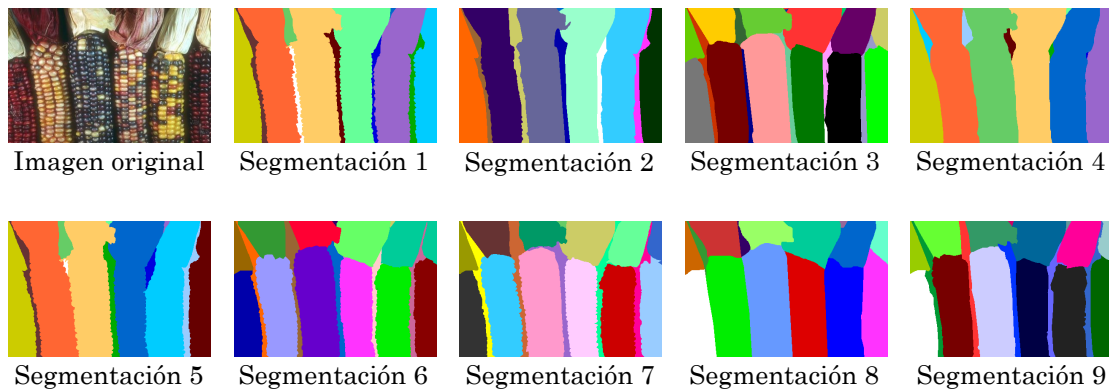


Figura 3.10. Ejemplos de segmentaciones reales o manuales de una imagen del repositorio BSDS500.



a) Clase “estrella de mar”



b) Clase “bisonte”



c) Clase “granos”



d) Clase “chile”

Figura 3.11. Ejemplos de algunas clases de objetos que aparecen tan sólo una vez en todo el repositorio de BSDS500.

aprecia en la Figura 3.12. También, en esta figura se observa que no hay estancamiento en el rendimiento de los métodos de Aprendizaje Profundo a diferencia de los algoritmos tradicionales de aprendizaje automático, en donde se llega a estancar el rendimiento, ya que por más imágenes que se les introduzca ya no aumenta significativamente su precisión.

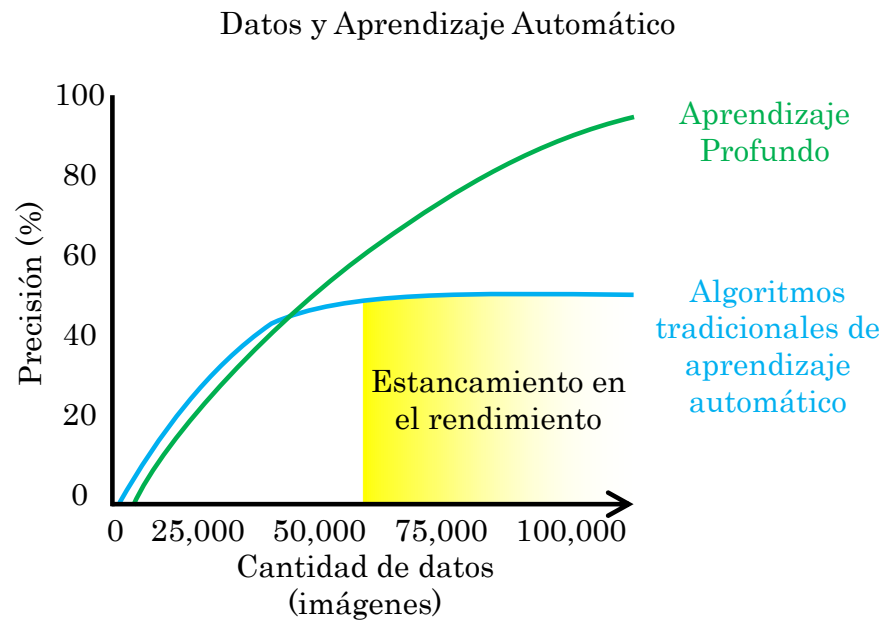


Figura 3.12. Rendimiento de los métodos de Aprendizaje Profundo y clásicos en relación a la cantidad de información con las que son entrenadas (Ng, 2016), (Amaratunga, 2017) y (Amin, 2016).

Para solucionar las problemáticas antes mencionadas, se siguieron los siguientes pasos:

- Se pasaron los datos de todas las segmentaciones de archivos en formato MAT a imágenes de 24 bits mediante una paleta de colores llamada *colorcube*, la cual visualmente da colores más diversos que el resto de paletas disponibles en MATLAB (MATLAB, 2017a), dando como resultado diversas etiquetas para una misma imagen, como se mostró en la Figura 3.10. Al final, se obtuvieron 1635 etiquetas de entrenamiento y validación, pero se tuvo que seleccionar manualmente sólo una etiqueta por imagen. Después de seleccionirlas, se prosiguió a colorearlas en PAINT y a transformarlas en imágenes indexadas de 8 bits, indexando casi a nivel de pixel cada región, donde cada índice representa a un conjunto de tres valores flotantes, que son los valores en RGB pero normalizados a 1, y a una clase de objeto. Esto último se muestra en la Figura 3.13, en donde se observa cómo está compuesta una etiqueta.

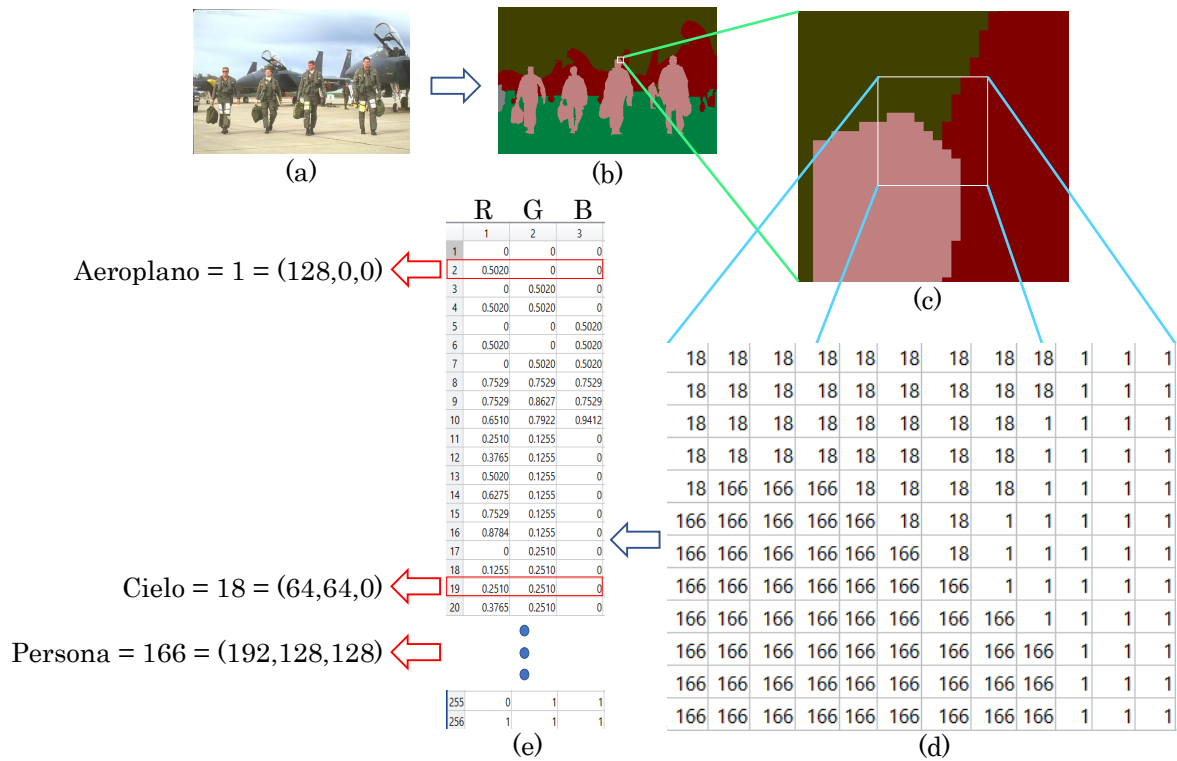


Figura 3.13. Estructura de una imagen indexada: (a) imagen original, (b) etiqueta o segmentación verdadera, (c) ampliación de una sección de la etiqueta, (d) índices pertenecientes a esa sección ampliada y (e) mapa de colores al cual se relacionan los índices de la imagen etiquetada, cada índice también representa a una clase de objeto.

- Después, se tuvo que cambiar el mapa de color de todas las etiquetas del BSDS500 al mapa de color del PASCAL VOC, haciendo un archivo de equivalencias de índices con el cual algunas clases de objetos preservaron su color inicial y otras fueron cambiadas para ajustarse al nuevo mapa de color.
- Posteriormente, se analizó imagen por imagen para crear una lista con todas las clases de objetos encontradas (la lista completa se puede ver en la Tabla 3.1), esto fue a criterio propio, relacionando una etiqueta a cada clase de objeto para después contabilizar la frecuencia de aparición de ellas en las imágenes del repositorio BSDS500.

Tabla 3.1. Lista de clases de objetos encontradas en el BSDS500

Índice	Clase de objeto	Índice	Clase de objeto	Índice	Clase de objeto
0	Fondo	40	Papel	80	Instrumento musical
1	Aeroplano	41	Libélula	81	Zorro
2	Anémona	42	Hongo	82	Árbol
3	Ave	43	Koala	83	Tabla de surf
4	Bote	44	Sábana, manta, tela	84	Cuerda, mecate
5	Botella	45	Canguro	85	Sombrero, gorra, casco
6	Pingüino	46	Leopardo	86	Rinoceronte
7	Carro	47	Cactus	87	Puente
8	Ocelote	48	Trineo	88	Recipiente
9	Pastizal	49	Remo	89	Bola, pelota
10	Vaca	50	Gato salvaje	90	Planta sin maceta
11	Mesa, comedor	51	Cerdo	91	Ovis vignei
12	Perro	52	Reno	92	Arena
13	Caballo	53	Pirámide	93	Luna
14	Chile	54	Tigre	94	Ballena
15	Persona	55	Alce africano	95	Paja
16	Planta con maceta	56	Agua	96	Iguana
17	Palmera	57	Cerca	97	Montaña
18	Serpiente	58	Jirafa	98	Arma
19	Tren	59	Edificación, construcción	99	Elefante
20	Flores	60	Pasto	100	Camello, dromedario
21	Oveja	61	Elote	101	Pedazo de carne
22	Estrella de mar	62	Oso	102	Sombrilla
23	Pez	63	Cebra	103	Caja
24	Cielo	64	Mariposa	104	Búfalo
25	Tierra	65	Hoja	105	Esquie
26	Saltamontes	66	Palo	106	León
27	Bisonte	67	Toro	107	Araña
28	Granos, maíz, frijol	68	Mariquita	108	Gorila
29	Herramienta	69	Caña de pescar	109	Lobo
30	Zapatos	70	Arbusto	110	Espada
31	Nutria	71	Silla, sillón	111	Carreta
32	Pintura, cuadro	72	Sol	112	Coral
33	Llama	73	Nube, humo, fumarola	113	Ardilla
34	Suelo, piso	74	Venado	114	Paracaidas
35	Cocodrilo	75	Copa	115	Canasta
36	Marmota	76	Puma	116	Delfín
37	Cámara fotográfica	77	Cabra, chivo		
38	Nieve	78	Lagarto		
39	Roca, piedra	79	Jaguar		

Aumento de la información

- Después, se aplicó el método llamado aumento de información, el cual consiste en tomar todas las imágenes con sus respectivas etiquetas y aplicarles ciertas transformaciones geométricas, tales como escalamiento, rotación, recortes y voltearla. En este caso, se escalaron las imágenes y etiquetas a 0.4, 0.8 y 1.2 veces su tamaño, además de ser rotadas 90, 180 y 270 grados, así como recortar las imágenes en cuatro partes iguales.

En la Figura 3.14 se muestra la imagen original y todas las posibles transformaciones geométricas establecidas para el aumento de la información.

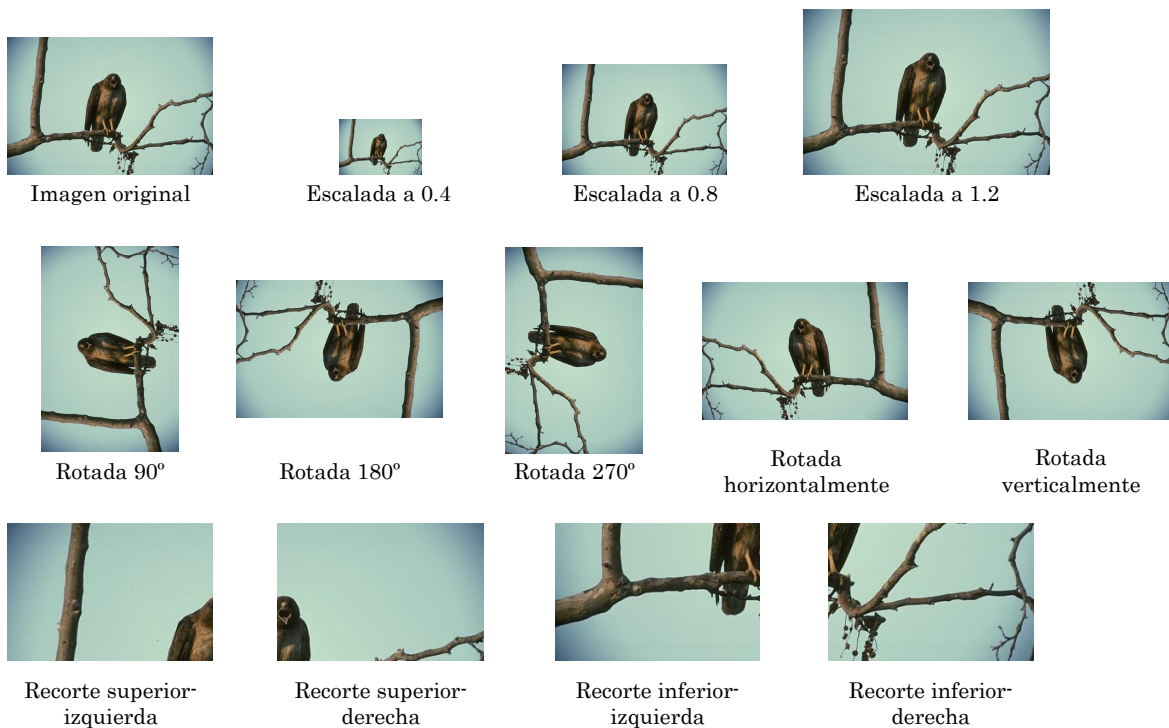


Figura 3.14. Ejemplo de una imagen a aplicarle las transformaciones geométricas para aumentar la información.

Transferencia de conocimiento

De igual manera, para aumentar la precisión en repositorios de pocas imágenes como este caso, se implementó el método de transferencia de conocimiento o también llamado transferencia de aprendizaje, como se realizaron en los trabajos de (Liu *et al.*, 2016) y (Chen *et al.*, 2015). Este método consiste en transferir características aprendidas de una red pre-entrenada a otra red similar con diferente repositorio, transfiriendo principalmente características profundas de primer y segundo orden (dadas en las primeras capas), dejando a las últimas capas afinar los pesos para que aprenda la red las características específicas del nuevo repositorio, tal como lo mencionan los trabajos de (MATLAB, 2017b) y (Oquab *et al.*, 2014). Para este caso, se transfirió el conocimiento de un modelo llamado FCN-8s, el cual fue entrenado con el repositorio ILSVRC, que cuenta con un millón trescientas mil imágenes a color (Russakovsky *et al.*, 2015).

Media del repositorio

Así mismo, se obtuvo la media aritmética por los tres canales (R, G y B) de las imágenes para entrenamiento del repositorio BSDS500 para asignárselo a los archivos de definición de la red (train.prototxt y val.prototxt) donde la ocuparon las capas de datos y/o de potencia, para realizar procedimientos como la sustracción media (Simonyan y Zisserman, 2014), o también como configurar el brillo, contraste, etc. Esta técnica lo utilizaron trabajos como el de (Liang y Hu, 2015), (Krizhevsky *et al.*, 2012) y (Zeiler y Fergus, 2013b).

Para terminar esta sección, se puede ver en la Figura 3.15 algunas de las imágenes del repositorio BSDS500 que se utilizaron para el entrenamiento de los diferentes modelos implementados.

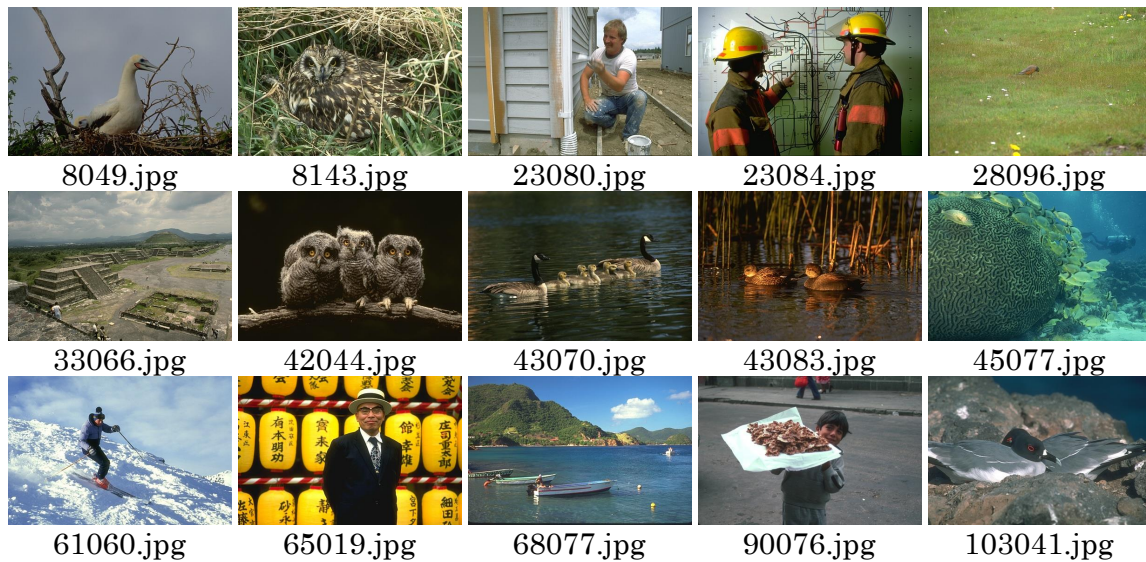


Figura 3.15. Muestra de imágenes del repositorio BSDS500 para entrenamiento.

Así mismo, en la Figura 3.16 se muestran algunas otras imágenes del BSDS500 empleadas para la validación de las redes.

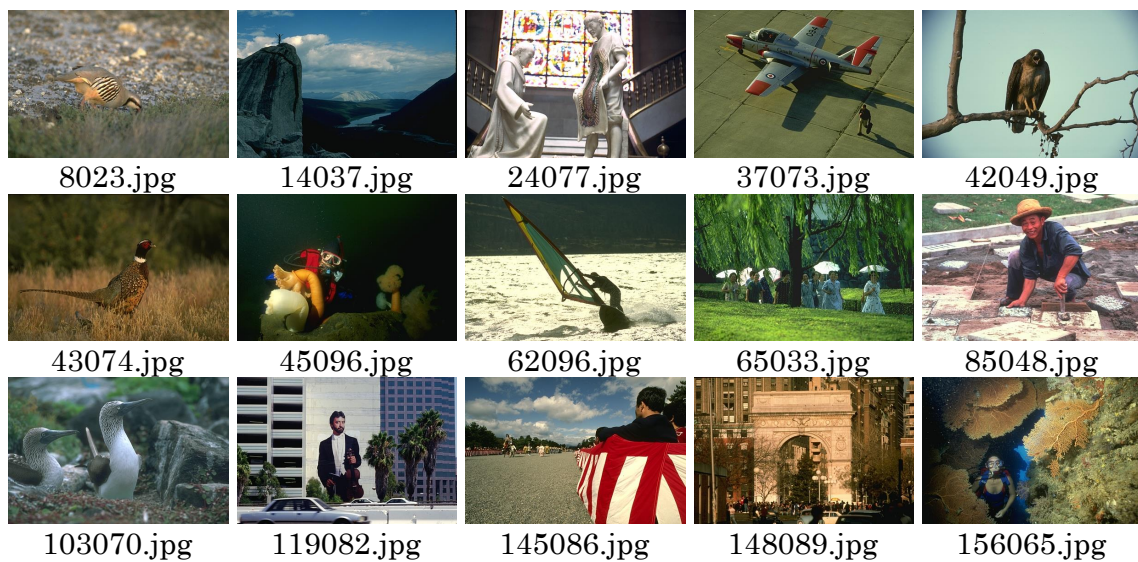


Figura 3.16. Muestra de imágenes del repositorio BSDS500 para validación.

Por último, en la Figura 3.17 se muestran algunas imágenes que se ocuparon para las pruebas de los modelos entrenados.

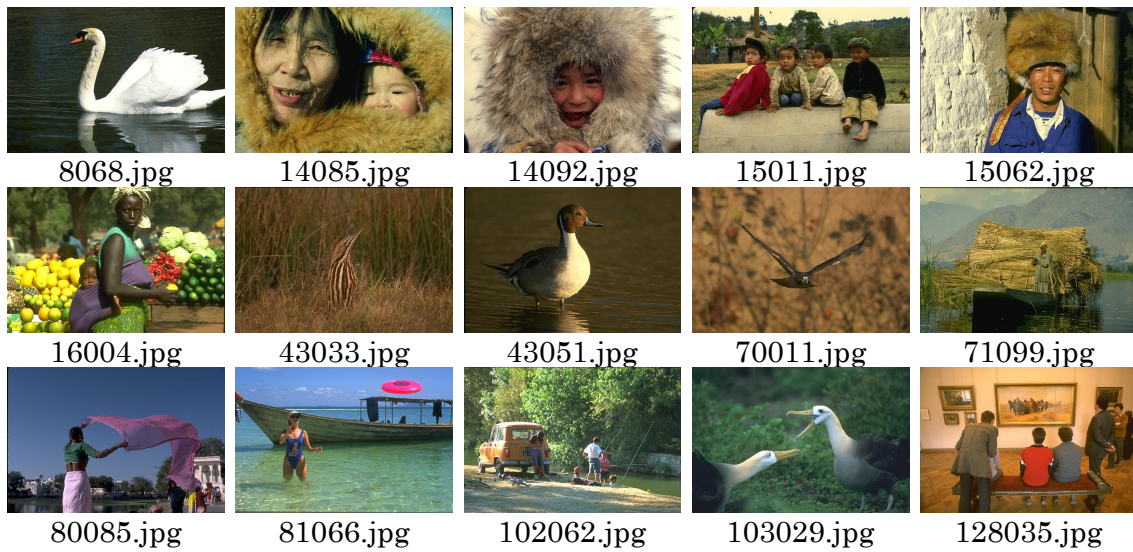


Figura 3.17. Muestra de imágenes del repositorio BSDS500 para prueba.

En el anexo 2 de este documento se muestran todas las imágenes que se utilizaron para el entrenamiento, validación y pruebas.

3.7. Entrenamiento y pruebas del modelo empleando el repositorio BSDS500

El proceso general seguido durante los experimentos, fue el que se muestra en la Fig 3.18, en la cual se resume los procesos anteriormente mencionados. En esta figura se muestra que tanto las imágenes de entrenamiento como las de validación fueron seleccionadas conforme las dos clases predominantes en el repositorio BSDS500, que son persona, ave y una más que es el fondo, para luego ser introducidas a la Red Convolutiva junto con la media de todas las imágenes de entrenamiento seleccionadas. En cuanto a sus respectivas etiquetas, antes de que fueron seleccionadas por las clases mencionadas, primeramente se convirtieron de archivos MAT a archivos PNG para ser visualizadas, y así fuera más fácil seleccionar sólo una etiqueta por imagen. Después, se les asignó un mapa de color específico para poder ser utilizadas en el entrenamiento junto con los pesos pre-entrenados con otro repositorio; luego pasaron por la misma selección que sus correspondientes imágenes para determinar qué etiquetas entrarían a la red. Una vez que se entrenó la red en el entorno de trabajo llamado CAFFE, se creó automáticamente un archivo de tipo CAFFEMODEL, el cual es el modelo FCN-8s ya entrenado. A este modelo se le pasan las imágenes de prueba, que también fueron seleccionadas conforme a las clases mencionadas, para segmentarlas y luego evaluarlas con las métricas PRI, VI y GCE, tomando como referencia las etiquetas de las imágenes de prueba, obteniendo de este modo, una evaluación cuantitativa y cualitativa de las segmentaciones realizadas con el modelo creado.

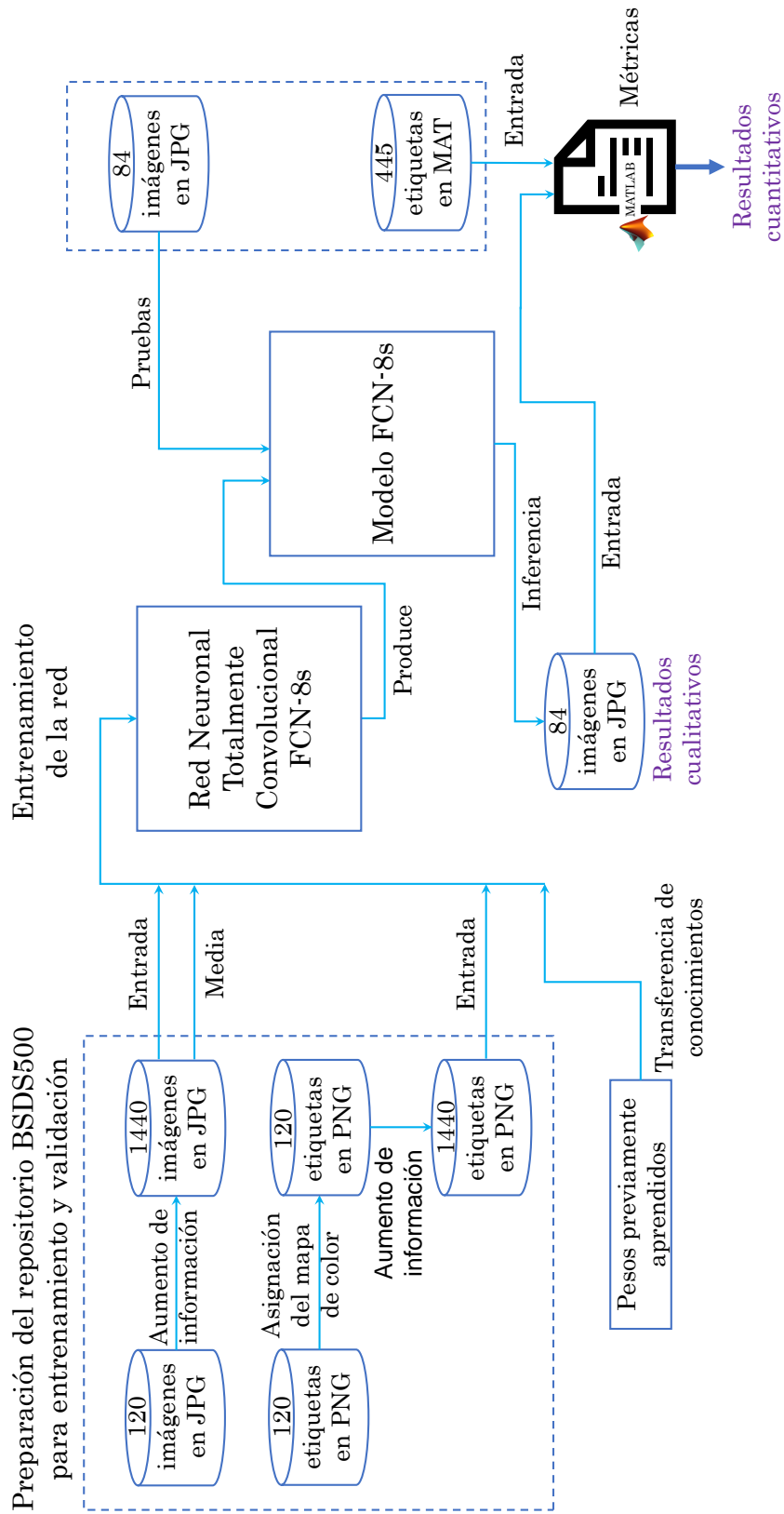


Figura 3.18. Proceso general seguido en la experimentación.

Sintonización original

Antes de pasar propiamente a la información obtenida de los diferentes experimentos realizados, se procede a mostrar la sintonización (configuración) original de los hiperparámetros empleados en las diferentes arquitecturas utilizadas. Así mismo, conviene hacer una breve aclaración en cuanto al significado de hiperparámetro y la diferencia que existe con el parámetro, ya que según (Ramasubramanian, 2017), el hiperparámetro es aquel valor que no es aprendido en el entrenamiento de la red, sino configurado manualmente antes del entrenamiento, mientras que el parámetro sí es aprendido durante el entrenamiento; además, los hiperparámetros influyen en el rendimiento del modelo, en otras palabras, los hiperparámetros dedican las reglas de entrenamiento del modelo por el cual los parámetros son estimados.

La sintonización original de los hiperparámetros para la arquitectura FCN-Alexnet se muestra en la Tabla 3.2, que viene a continuación:

Tabla 3.2. Sintonización de hiperparámetros de la arquitectura FCN-Alexnet

Hiperparámetro	Valor	Hiperparámetro	Valor
test_iter:	736	momentum:	0.9
test_interval:	999999999	iter_size:	20
display:	20	max_iter:	100000
average_loss:	20	weight_decay:	0.0005
lr_policy:	“fixed”	snapshot:	4000
base_lr:	1e-4	test_initialization:	false

En cuanto a la arquitectura FCN-8s, el cual viene en el proyecto de *Shelhamer*, él utilizó la misma arquitectura pero con distintas sintonizaciones para los repositorios de NYUD V2, PASCAL Context, PASCAL VOC 2012 y SIFT FLOW, siendo estos dos últimos repositorios los más parecidos, en cuanto al tipo de imágenes y de clases halladas, con el BSDS500. Por esta razón, se presenta en la Tabla 3.3 la sintonización de los hiperparámetros del FCN-8s que *Shelhamer* utilizó para ser entrenados con el PASCAL VOC, pero que en los subsecuentes experimentos de esta tesis se entrenó con el BSDS500.

Tabla 3.3. Sintonización de hiperparámetros originalmente para PASCAL VOC

Hiperparámetro	Valor	Hiperparámetro	Valor
test_iter:	736	momentum:	0.99
test_interval:	999999999	iter_size:	1
display:	20	max_iter:	100000
average_loss:	20	weight_decay:	0.0005
lr_policy:	“fixed”	snapshot:	4000
base_lr:	1e-14	test_initialization:	false

Después, se muestra en la Tabla 3.4 la sintonización del FCN-8s que originalmente era para el repositorio SIFT FLOW, pero que de igual manera se entrenó con el repositorio BSDS500.

Tabla 3.4. Sintonización de hiperparámetros originalmente para SIFT FLOW

Hiperparámetro	Valor	Hiperparámetro	Valor
test_iter:	200	momentum:	0.99
test_interval:	999999999	iter_size:	1
display:	20	max_iter:	300000
average_loss:	20	weight_decay:	0.0005
lr_policy:	“fixed”	snapshot:	4000
base_lr:	1e-12	test_initialization:	false

El significado o función de los diversos hiperparámetros configurados se pueden observar en el anexo de este documento.

Según el artículo de (Murugan, 2017), la sintonización de los hiperparámetros es un paso crucial, ya que la precisión y el rendimiento de la red dependen totalmente de los hiperparámetros. De los cuales, los más importantes son el número de capas convolucionales y de otros tipos de capas, así como la tasa de aprendizaje, siendo los primeros definidos por la arquitectura seleccionada, mientras que la tasa de aprendizaje se puede cambiar sin tener que modificar la estructura de la red para poder encontrar el valor que mejor resultado otorgue. Es por eso que, en los siguientes experimentos se entrenó sólo con diferentes grados de aprendizaje, ya que los otros hiperparámetros (como el *momentum*, el *weight_decay*, entre otros) estaban configurados de manera similar en todos los modelos, a excepción del grado de aprendizaje y otros. Otros hiperparámetros como el *test_iter*, *iter_size* y *max_iter* no influyen significativamente, según algunas pruebas realizadas empíricamente. También fueron probados los diferentes métodos de optimización existentes en el entorno de trabajo llamado Caffe, siendo el Gradiente Descendente Estocástico (SGD) el que mejor resultado dio.

Para estos experimentos, se realizó un entrenamiento con el modelo FCN-Alexnet y ocho más con el modelo FCN-8s y con el repositorio BSDS500, los cuales se describen a continuación.

Experimentos: 1, 2 y 3 (determinación de una sintonización base)

Para no entrenar con todas las clases encontradas en el BSDS500 y disminuir el tiempo de entrenamiento, se entrenó sólo con las tres clases de objetos más representativas del repositorio a trabajar, las cuales son: persona, ave y fondo.


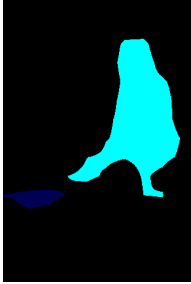



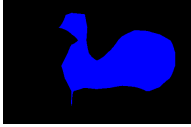
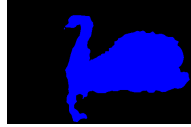

Se buscaron los diferentes tipos de Redes Neuronales Convolucionales que se han utilizado para segmentar imágenes naturales a color, esto con el fin de encontrar una arquitectura base de donde partir, encontrándose disponibles los siguientes: FCN-Alexnet y FCN-8s tanto para el repositorio PASCAL VOC como para SIFT FLOW.

El primer experimento o entrenamiento se hizo utilizando la arquitectura FCN-Alexnet junto con el repositorio BSDS500. En cuanto la arquitectura FCN existen de tres tipos: una con paso de 32 (32 *stride*, 32s), otro con paso de 16 (16 *stride*, 16s) y otro más con paso de 8

(8 *stride*, 8s), siendo esta última la que segmenta con mayor precisión en comparación con las otras dos, por lo que se tomó el FCN-8s para los siguientes experimentos.


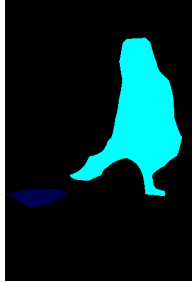



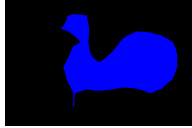
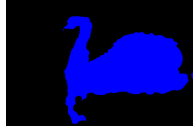

El segundo entrenamiento se hizo con la sintonización definida como si fuera para PASCAL VOC, mientras que el tercer entrenamiento se hizo con la sintonización definida como si fuera para SIFT FLOW. Luego se evaluaron cualitativa y cuantitativamente estos tres entrenamientos, siendo la segunda sintonización la que dio un mejor resultado en cuanto a la métrica PRI (promediando la precisión de las 84 imágenes de prueba) y siendo la tercera sintonización mejor en cuanto a los hiperparámetros VI y GCE, esto sin tomar en cuenta la segmentación manual (real), como se aprecia en la Tabla 3.5, en la cual, también se muestra una imagen (de las 84 probadas) por clase para evaluar cualitativamente su segmentación.

Tabla 3.5. Comparación entre la sintonización de FCN-Alexnet y FCN-8s para PASCAL VOC y SIFT FLOW (con todas las imágenes de prueba)

	Segmentación manual (real)	Con modelo FCN-Alexnet	Con modelo FCN-8s para PASCAL VOC	Con modelo FCN-8s para SIFT FLOW
Clase Persona				
Clase Ave				
PRI	0.555	0.540	0.542	0.538
VI	2.183	2.600	2.516	2.488
GCE	0.020	0.132	0.100	0.091

Como no hubo una clara ventaja de una sintonización con respecto a otra, se decidió aplicar las métricas a una sola imagen de persona y a otra más de un ave, dando como resultado los valores que se presentan en la Tabla 3.6, la cual muestra mejores resultados en todos las métricas con la sintonización definida para SIFT FLOW.

Tabla 3.6. Comparación entre la sintonización de FCN-Alexnet y FCN-8s para PASCAL VOC y SIFT FLOW (con una imagen de prueba)




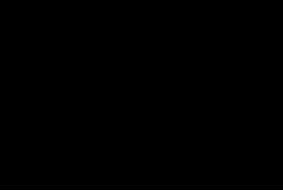



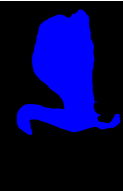
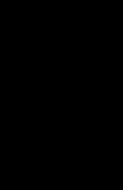

	Segmentación manual (real)	Con modelo FCN-Alexnet	Con modelo FCN-8s para PASCAL VOC	Con modelo FCN-8s para SIFT FLOW
Segmentación de persona				
PRI	0.833	0.781	0.805	0.810
VI	1.311	1.613	1.522	1.493
GCE	0.010	0.070	0.046	0.041
Segmentación de ave				
PRI	0.797	0.723	0.753	0.760
VI	0.838	1.298	1.280	1.176
GCE	0.015	0.102	0.111	0.079

Con los experimentos anteriores se vio un poco más marcado el aumento de precisión en la segmentación al configurar la red con los hiperparámetros como si fuera para el repositorio SIFT FLOW que para PASCAL VOC, por lo que se decidió dejar como configuración base la que era para el repositorio SIFT FLOW. Posteriormente, se modificó sólo el grado de aprendizaje base (lr_base) para ver como influía en las inferencias.

Experimentos: 4, 5 y 6 (ajustes en el grado de aprendizaje)



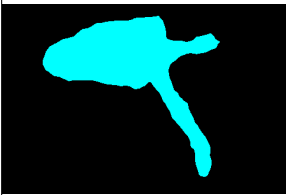



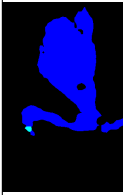
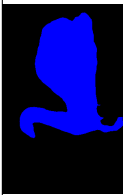

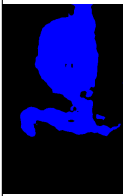
En estos experimentos se cambió únicamente el grado de aprendizaje base para ver como influía en la calidad de la segmentación, y así, poder determinar su mejor valor para segmentar las imágenes del BSDS500. Entonces, en el cuarto entrenamiento se configuró el grado de aprendizaje base en $1e - 16$, en el quinto entrenamiento se cambió el $base_lr$ a $1e - 10$ y en el sexto entrenamiento se cambió el $base_lr$ de forma gradual desde $1e - 12$ hasta $1e - 16$, siendo el primero el valor de $base_lr$ original para SIFT FLOW ($1e - 12$) y el segundo un valor menor al $base_lr$ original para PASCAL VOC ($1e - 14$), tomando así todo el rango de valores del hiperparámetro lr_base donde da mejores resultados. En la Tabla 3.7 se puede ver esta comparación de resultados.

Tabla 3.7. Comparación entre diferentes sintonizaciones (con todas las imágenes de prueba)

	Segmentación manual (real)	Con hiperparámetros para SIFT FLOW	Con $base_lr$ de $1e-16$	Con $base_lr$ de $1e-10$	Con $base_lr$ desde $1e-12$ hasta $1e-16$
Clase Persona					
Clase Ave					
PRI	0.555	0.538	0.549	0.267	0.536
VI	2.183	2.488	2.462	2.712	2.503
GCE	0.020	0.091	0.092	0	0.094

De la evaluación anterior se aprecia que mejoró la precisión de la segmentación, en cuanto a las métricas PRI y VI, al disminuirle el grado de aprendizaje base a $1e - 16$, mientras que al subirle a $1e - 10$ a este mismo hiperparámetro, la red no segmentó alguna de las dos clases con la que fue entrenada y sólo mostró una imagen en negro; tampoco dio el mejor de los resultados a cambiarle gradualmente el grado de aprendizaje base desde $1e - 12$ hasta $1e - 16$. Luego se procedió a realizar la misma comparación pero tomando en consideración una imagen de persona y una de ave, para poder observar más puntualmente la diferencia de precisión entre una sintonización y otra, como se aprecia en la Tabla 3.8.

Tabla 3.8. Comparación entre diferentes sintonizaciones (con una imagen)

	Segmentación manual (real)	Con hiperparámetros para SIFT FLOW	Con $base_lr$ de $1e-16$	Con $base_lr$ de $1e-10$	Con $base_lr$ desde $1e-12$ hasta $1e-16$
Segmentación de persona					
PRI	0.833	0.810	0.814	0.556	0.808
VI	1.311	1.493	1.466	1.892	1.503
GCE	0.010	0.041	0.036	0	0.043
Segmentación de ave					
PRI	0.797	0.760	0.753	0.461	0.762
VI	0.838	1.176	1.249	1.485	1.181
GCE	0.015	0.079	0.103	0	0.080

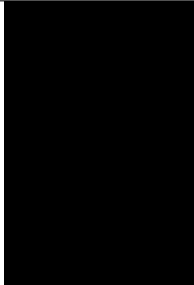
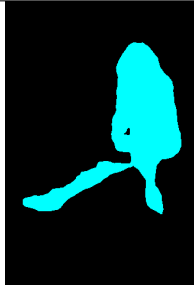
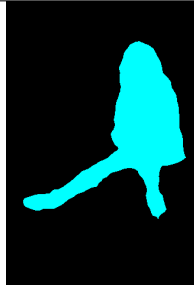
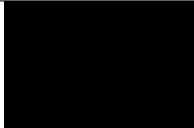

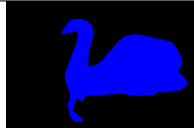
En la tabla comparativa anterior se ve claramente que el *base_lr* en $1e - 16$ funciona mejor en la segmentación de la clase persona, al ser el mejor evaluado en las tres métricas; sin embargo, en la clase ave, no fue el mejor en alguna de las tres métricas.

Experimentos: 7, 8 y 9 (pruebas con métodos alternos)

La finalidad de estos últimos experimentos fue implementar y evaluar qué tanto influye, en el rendimiento de la segmentación de imágenes, los diversos métodos que se mencionaron en el estado del arte de la tesis titulada “Sintonización de una Red Totalmente Conectada para segmentación de dos clases de objetos en imágenes” y así poder tomar el método que funcione para aumentar la precisión de la segmentación.

Posteriormente, para el séptimo entrenamiento no se aplicó el método de transferencia del conocimiento, ya que en todos los entrenamientos anteriores sí se había realizado este método. En el octavo entrenamiento se implementó, a parte de la transferencia de conocimiento, el método de aumento de información, con las transformaciones que se explicó anteriormente en este documento. Por último, se realizó el noveno entrenamiento estableciendo el valor de la media del repositorio BSDS500 en el archivo de configuración de la red. Estos tres últimos entrenamientos se hicieron con la sintonización de la red como si fuera para SIFT FLOW, con el grado de aprendizaje base en $1e - 16$.

Tabla 3.9. Comparación de resultados con sintonización para SIFT FLOW (con todas las imágenes de prueba)

	Hiperparámetros para SIFT FLOW sin transferencia del conocimiento	Hiperparámetros para SIFT FLOW con transferencia y aumento de la información	Hiperparámetros para SIFT FLOW con media del BSDS500
Clase Persona			
Clase Ave			
PRI	0.267	0.539	0.550
VI	2.712	2.524	2.460
GCE	0	0.102	0.093

De estos últimos experimentos se determinó usar la media del repositorio BSDS500 ya que, cuantitativamente por medio de las tres métricas implementadas, dio mejores resultados al utilizarlo en la red.

En la Figura 3.19 se aprecia más claramente que la precisión del FCN-8s modificado en este trabajo (FCN 1 con valor de 0.55) fue la segunda mejor respecto a la métrica PRI, después de la segmentación manual (con valor de 0.555). Cabe aclarar que el FCN 2 (con valor de 0.538) representa al modelo FCN-8s creado originalmente para SIFT FLOW, el FCN 3 (con valor de 0.542) es el modelo FCN-8s originalmente para PASCAL VOC y el FCN 4 (0.54) es el FCN-Alexnet entrenado originalmente con PASCAL VOC.

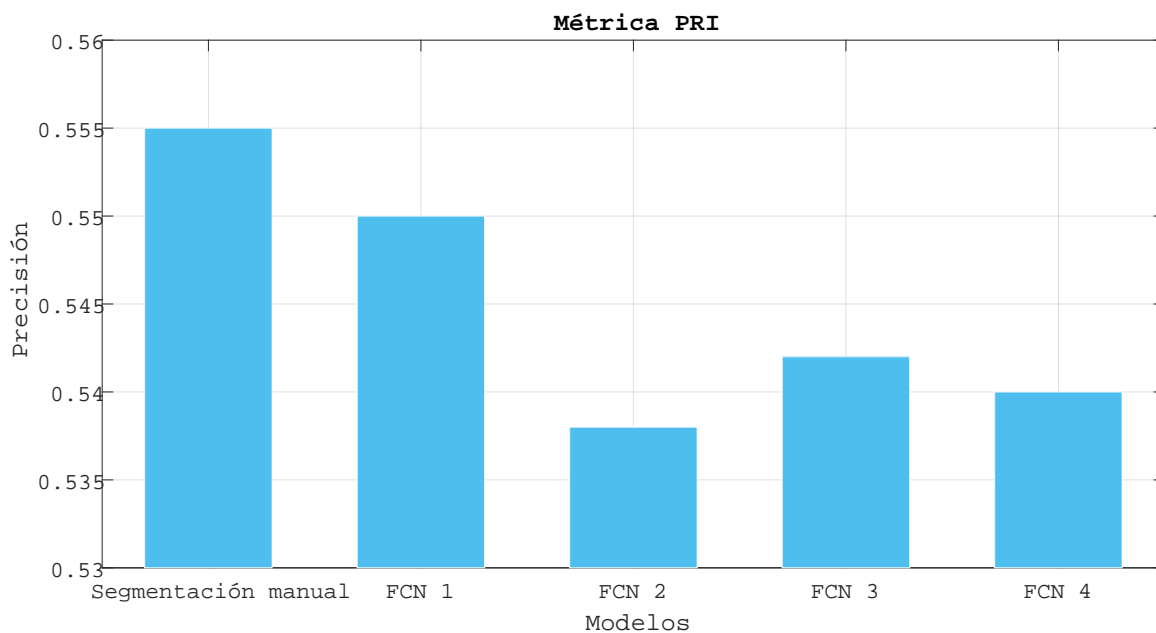


Figura 3.19. Gráfica comparativa entre los resultados obtenidos con la métrica PRI.

También en la Figura 3.20 se puede observar que la precisión del FCN-8s (FCN 1 con valor de 2.46) fue la segunda mejor respecto a la métrica VI, después de la segmentación manual (con valor de 2.183). El FCN 2 (con valor de 2.488) es el modelo 3 (con valor de 2.516) y el FCN 4 (con valor de 2.6)

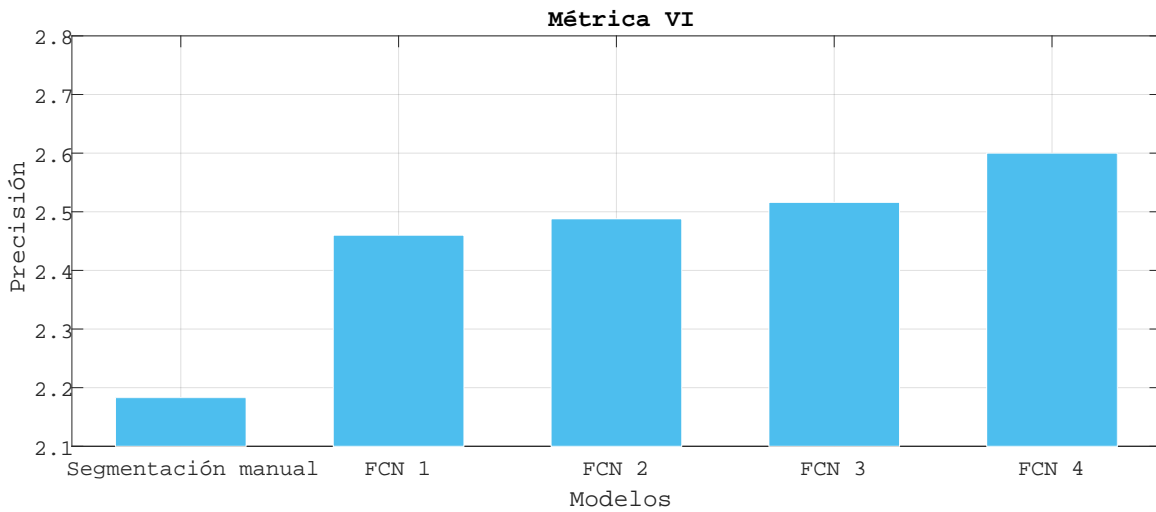


Figura 3.20. Gráfica comparativa entre los resultados obtenidos con la métrica VI.

Por último, en la Figura 3.21 se puede ver que la precisión del FCN-8s modificado (FCN 1 con valor de 0.093) en este trabajo fue la tercera mejor respecto a la métrica GCE, después de la segmentación manual (con valor de 0.02) y del modelo FCN-8s originalmente para SIFT FLOW (FCN 2 con valor de 0.091). Cabe aclarar que el FCN 3 (con valor de 0.1) es el modelo FCN-8s originalmente para PASCAL VOC y el FCN 4 (con valor de 0.132) es el FCN-Alexnet entrenado originalmente con PASCAL VOC.

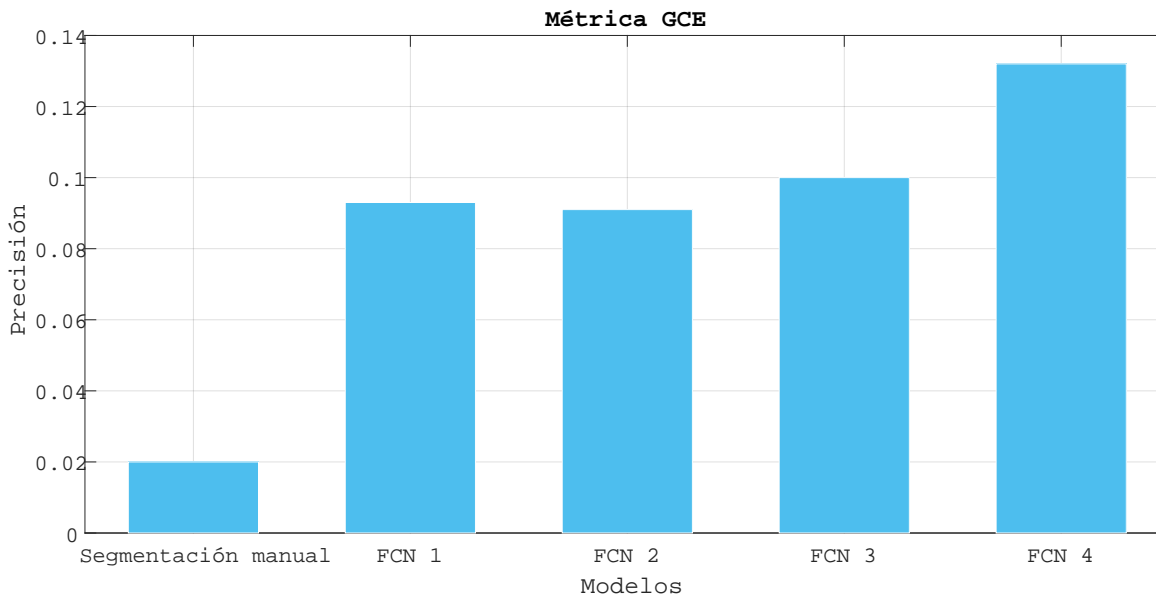


Figura 3.21. Gráfica comparativa entre los resultados obtenidos con la métrica GCE.

Análisis de significancia estadística

Por último, en este capítulo se realizó una prueba de hipótesis para analizar su significancia estadística, es decir, aceptar o rechazar que la configuración realizada en el experimento 9 con la media del BSDS500 produce mejores segmentaciones que al no utilizarla y que estas mejoras son producidas de forma causales y no debidas al azar, en cuanto a la segmentación de imágenes a colores se refiera, dependiendo de la evidencia proporcionada por una muestra de datos de 200 imágenes (las imágenes de prueba del BSDS500).

Para llevar a cabo este análisis, primero se definen dos valores: p – *valor* (probabilidad de cometer un error de tipo 1) y α (nivel de significancia, es decir, la máxima cantidad de error que se está dispuesto aceptar para dar como válida una hipótesis). Cabe señalar que un error de tipo 1 (falso positivo) es aquel que ocurre cuando se acepta una hipótesis, siendo que la proposición (hipótesis) era falsa. Convencional e internacionalmente, se da un valor de 0.05 a α , este valor fue determinado por el estadístico Fisher en su libro (Fisher, 1934) y retomado por trabajos como el de (Sterne y Smith, 2001) y (Monterrey Gutiérrez, 2012). En cuanto al valor de p – *valor*, primero se define la hipótesis nula o de trabajo (H_0) y la hipótesis alterna o del investigador (H_1) para cada métrica, esto con el fin de obtener posteriormente el valor de p – *valor* y así determinar si se acepta o rechaza la hipótesis nula.

Hipótesis para las tres métricas (PRI, VI y GCE):

H_0 : $\mu_1 = \mu_2$, es decir, que no hay una diferencia estadísticamente significativa entre las medias de las métricas PRI, VI y GCE aplicadas a las imágenes de prueba del repositorio BSDS500 con el modelo FCN-8s, con media y sin media aritmética.

H_1 : $\mu_1 \neq \mu_2$, es decir, que sí hay una diferencia estadísticamente significativa entre las medias de las métricas PRI, VI y GCE aplicadas a las imágenes de prueba del repositorio BSDS500 con el modelo FCN-8s, con media y sin media aritmética.

siendo μ_1 las medias de las métricas PRI, VI y GCE aplicadas a las segmentaciones de las imágenes de prueba (del BSDS500) realizadas por el modelo FCN-8s que utilizó la media del BSDS500 para entrenar. Por el contrario, el μ_2 son las medias de las métricas PRI, VI y GCE aplicadas a las segmentaciones de las imágenes de prueba (del BSDS500) realizadas por el modelo FCN-8s que no utilizó la media del BSDS500 para entrenar.

Posteriormente, en Matlab se obtuvieron los siguientes valores con la función `ttest2(x,y)`:

- A un nivel de significancia de 0.05, se obtuvo un p – *valor* de $6,262e - 10$ para la métrica PRI, lo cual indica que se puede rechazar la hipótesis nula para tomar la hipótesis alterna. Además, también resultó que existe una mejora estadísticamente significativa, según la métrica PRI, al utilizar la media del BSDS500 para entrenar el modelo FCN-8s.
- A un nivel de significancia de 0.05, se obtuvo un p – *valor* de 0,272 para la métrica VI, lo cual indica que no se puede rechazar la hipótesis nula. Por lo tanto, no hay evidencia

estadísticamente significativa, según la métrica VI, de que haya una diferencia entre utilizar y no la media del BSDS500 para entrenar el modelo FCN-8s.

- A un nivel de significancia de 0.05, se obtuvo un p -valor de 0,123 para la métrica GCE, lo cual indica que no se puede rechazar la hipótesis nula. Por lo tanto, no hay evidencia estadísticamente significativa, según la métrica GCE, de que haya una diferencia entre utilizar y no la media del BSDS500 para entrenar el modelo FCN-8s.

3.8. Discusión

La configuración más exitosa, según cualitativa y cuantitativamente, quedó de la siguiente manera: configurando los hiperparámetros como si fuera para el repositorio SIFT FLOW, a excepción del grado de aprendizaje base ($base_lr$), que se debe configurar en $1e - 16$ y también se debe de utilizar la media del repositorio BSDS500 para que se mejore el rendimiento de la segmentación en un 3,43 % con respecto al modelo FCN-Alexnet y un 1,23 % con respecto al modelo FCN-8s, al promediar las tres métricas PRI, VI y GCE de forma global (por las 84 imágenes de prueba). En cuanto a los métodos de optimización, también se experimentó con todos los tipos disponibles en el entorno de trabajo de Caffe, siendo el gradiente descendiente estocástico (*Stochastic Gradient Descent*, SGD) el que mejor desempeño tuvo.

Por otra parte, no se pudo comparar de forma cuantitativa los resultados obtenidos con otros trabajos del estado del arte debido a que esos trabajos se enfocan más a reducir el tiempo de segmentación y no a mejorar la precisión de ésta, y otros trabajos más evalúan la precisión mediante otras métricas que no se implementaron en esta tesis. Tampoco se evalúa el tiempo de entrenamiento debido a que no forma parte de los objetivos específicos establecidos al principio de esta tesis.

En los nueve experimentos realizados, se utilizaron más imágenes (para entrenamiento) de la clase persona que de la clase ave en una relación de 71 : 15, sin embargo, al medir la precisión de la segmentación por clase, se observa que hay una mayor precisión en la clase ave que en la clase persona en todos los experimentos efectuados, como se muestra en la Figura 3.22. Donde la métrica PRI es mayor, lo cual resulta mejor, en la clase ave que en la clase persona; por lo contrario, las métricas VI y GCE son menores, lo cual resulta mejor, en la clase ave que en la clase persona, teniendo estos mismos comportamientos en los nueve experimentos que se llevaron a cabo.

Esto se debe posiblemente a que los pesos transferidos a los modelos implementados fueron aprendidos originalmente mediante el repositorio ILSVRC, el cual contiene 1242 imágenes de personas y 2126 imágenes de aves, lo cual pudo influir a que segmentaran mejor las imágenes de aves que de personas. Otro factor que pudo influir fue el hecho que las personas utilizan una gran diversidad de accesorios, la cual interfiere con la textura, color y forma original de ellas; mientras las aves no utilizan accesorios, lo que hace que mantenga su textura, color y forma original.

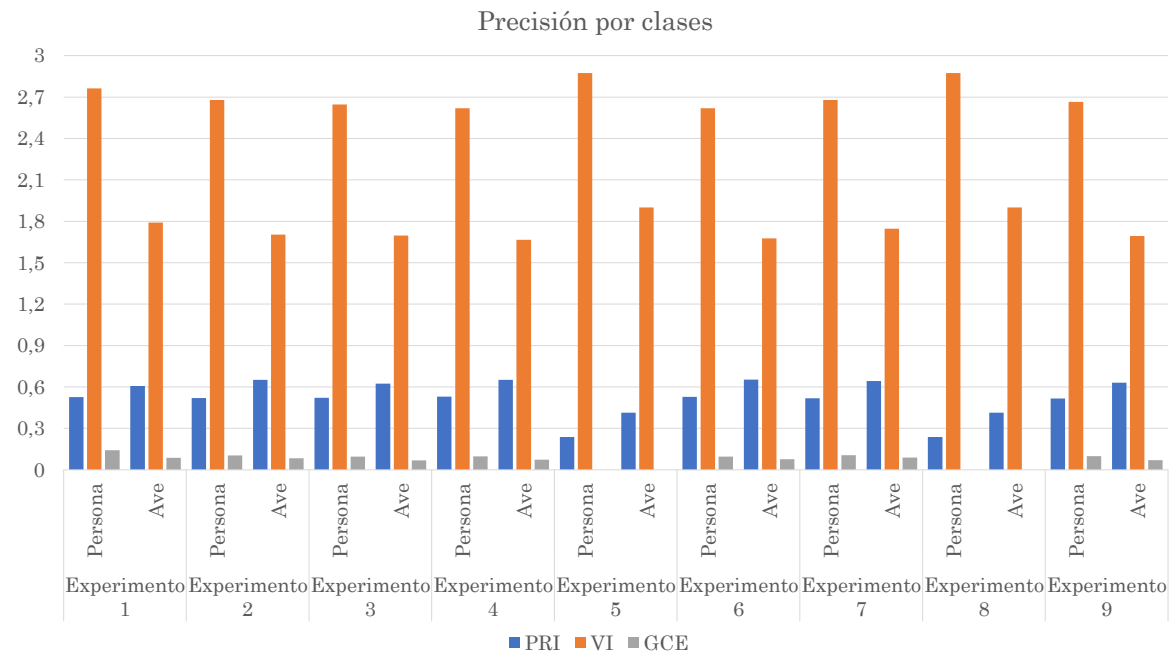


Figura 3.22. Precisión por clases.

Esta disparidad entre el número de imágenes por clases también se observa en otros repositorios, como el PASCAL VOC 2011, en donde existen 316 imágenes de la clase persona y 84 de la clase ave, teniendo una relación de 79 : 21.

Capítulo 4

Conclusiones

En la presente sección se muestra a detalle todo lo que se logró del trabajo de tesis, como son los objetivos y alcances logrados, resultados (productos, aportaciones y conclusiones) y trabajos a futuro.

4.1. Objetivos y alcances logrados

En la Tabla 4.1 se pueden ver los objetivos alcanzados con sus respectivas tareas.

Tabla 4.1. Objetivos alcanzados.

Objetivos	Actividades
Comprender los conceptos básicos de segmentación de imágenes.	Se estudió e incluyó el tema de segmentación de imágenes en el reporte del marco conceptual, así como las diferentes técnicas de segmentación.
Comprender los conceptos básicos de las Redes Neuronales Convolucionales.	Se analizó a fondo todas las capas de las Redes Neuronales Convolucionales en el reporte del marco conceptual, mostrando su funcionamiento y la ecuación matemática que sustenta a cada capa.
Comprender cómo procesa información una GPU.	Se incluyó el concepto básico de Unidad de Procesamiento Gráfico (GPU) y su funcionamiento de manera general en el reporte del marco conceptual.
Implementar un modelo basado en una Red Neuronal Convolutional que realice la segmentación de imágenes a color.	En la sección de experimentación se detallan los dos modelos de Redes Neuronales Convolucionales que se implementaron para segmentar imágenes a color sobre una GPU, los cuales fueron: FCN-Alexnet y FCN-8s.
Cuantificar la calidad de la segmentación y comparar los resultados con los obtenidos en otros trabajos mencionados en el estado del arte.	Se utilizaron las métricas de PRI, VI y GCE sobre las segmentaciones inferidas con los modelos creados para evaluar cuantitativamente la precisión en la segmentación de imágenes, también se hicieron comparaciones entre las segmentaciones inferidas para la evaluación cualitativa.

También, en la Tabla 4.2 se pueden apreciar los alcances logrados en este trabajo con sus respectivas tareas.

Tabla 4.2. Alcances logrados.

Alcances	Actividades
Trabajar con imágenes a color del repositorio BSDS500.	Se utilizaron las imágenes y etiquetas de entrenamiento, validación y pruebas del repositorio BSDS500, tal como se menciona en la sección de Experimentación.
Realizar el procesamiento de imágenes con una GPU NVIDIA con 640 núcleos.	Todos los experimentos y pruebas se realizaron mediante una GPU NVIDIA GeForce GTX 960M con 640 núcleos desde Linux.
Evaluar conforme a las métricas analizadas proporcionadas por el Grupo de Visión por Computadora de la Universidad de Berkeley.	Se evaluó cuantitativamente el rendimiento de la segmentación con las métricas PRI, VI y GCE, todas ellas proporcionadas por el Grupo de Visión por Computadora de la Universidad de Berkeley.
Comparar los resultados obtenidos con los resultados reportados en otros trabajos del estado del arte, sólo los que muestran el rendimiento de la segmentación realizada o los valores de las métricas de evaluación de dichas segmentaciones.	De las segmentaciones inferidas por el modelo FCN-8s se evaluó con las métricas mencionadas anteriormente, para después compararlas con los resultados de otros trabajos, como se muestra en la sección no. 7: Comparaciones con otros métodos.

4.2. Productos

Los productos que se mencionan a continuación serán entregados en un disco compacto anexo a la tesis.

1. Reporte del estado del arte

Se redactó un reporte del estado del arte, en el cual se detallan los trabajos analizados, así como también se muestra una tabla comparativa entre los métodos utilizados en cada uno de ellos, y los cuales son el sustento de este trabajo.

2. Reporte del marco conceptual

Se elaboró un reporte del marco conceptual, donde se describen los conceptos que comprende este trabajo de tesis, como son: la segmentación de imágenes, el Aprendizaje Profundo, las Redes Neuronales Convolucionales y la computación acelerada.

3. Modelo de Red Neuronal Convolutacional entrenado con el BSDS500

Se produjo un modelo de Red Neuronal Totalmente Convolutacional de paso 8 (FCN-8s) entrenado con las imágenes y etiquetas del repositorio BSDS500, junto con los pesos

transferidos que fueron aprendidos con el repositorio ILSVRC. El modelo está en formato CAFFEMODEL y se puede implementar al tener instalado el software necesario descrito en el reporte de avances, en la sección 4.2.2 Arquitectura de software. La arquitectura FCN-8s fue creado por (Long *et al.*, 2015) al fusionar las redes: VGG-16, AlexNet y GoogLeNet, y convolucionar algunas capas de clasificación y poder así segmentar imágenes.

4. **Reporte de resultados**

Se hizo este documento como parte de los productos entregables mencionados en la propuesta de tesis aceptada, la cual comprende la descripción de los repositorios, el modelo implementado, las métricas, la experimentación y los resultados.

5. **Repositorio BSDS500 ya preparado**

Se entregarán todas las imágenes y etiquetas originales del repositorio BSDS500, así como las que fueron seleccionadas, pre-procesadas (sólo etiquetas) y aumentadas con las transformaciones geométricas vistas en la sección de Experimentación de este documento.

6. **Artículo “Segmentación semántica de imágenes naturales utilizando Redes Neuronales Convolucionales”**

Se escribió un artículo para el Congreso Internacional en Tecnología, Innovación y Docencia (CITID 2017) que se realizó en el Instituto Tecnológico de Zacatepec (ITZ) en marzo de 2017, como lo muestra la Figura 4.1 en donde se aprecia la primera página del artículo y la Figura 4.2 en donde se observa la constancia de participación.

SEGMENTACIÓN SEMÁNTICA DE IMÁGENES NATURALES UTILIZANDO REDES NEURONALES CONVOLUCIONALES

D. G. Suárez-Santiago, D. Mújica-Vargas y M. Mejía-Lavalle

¹*Departamento de Ciencias Computacionales del Centro Nacional de Investigación y Desarrollo Tecnológico, Interior Internado Palmira S/N, Col. Palmira Cuernavaca, Morelos, México - diegogabriel@cenidet.edu.mx, dantemv@cenidet.edu.mx, mlavalle@cenidet.edu.mx*

Área de participación: *Ciencias de la Computación / Inteligencia artificial*

Resumen: En este trabajo de investigación se segmentó semánticamente imágenes a color de los repositorios de PASCAL VOC 2012 y BSDS500 de Berkeley, utilizando las Redes Neuronales Convolucionales. Se realizó con este tipo de redes debido a que poseen las cualidades de ser invariante a traslación, escala y distorsión, así como por obtener de forma automática las características más relevantes para cada conjunto de imágenes, y también por tener implícitas las ventajas del Aprendizaje Profundo: robustez, generalizable y escalable. También se muestran las herramientas que se instalaron para entrenar, validar y probar la Red Neuronal Convolutiva mediante la Unidad de Procesamiento Gráfico. Por último, se da a conocer una evaluación visual y subjetiva de los resultados, con los cuales se demostró su capacidad de la red para la tarea asignada, debido a la precisión y tasa de error mostradas en la segmentación semántica realizada.

Introducción.

La Red Neuronal Convolutiva ha liderado la Inteligencia Artificial en los últimos años [15] debido a los buenos resultados obtenidos en las tareas de clasificación y reconocimiento de objetos [3 y 9]. Esos resultados se han obtenido gracias a la extracción de características significativas que realiza ese tipo de red mediante las operaciones de convolución. Las características obtenidas son útiles para el proceso de segmentación, la cual es la tarea medular en el proceso de reconocimiento visual, pero esta tarea se complica cuando se presentan en las imágenes sombras y reflexión, variación de intensidad o color en las regiones, texturas parecidas, iluminación no uniforme, traslapamiento de imágenes, oclusión, objetos con bordes suavizados o difuminados con el fondo, deformación y desorden del fondo [10].

En años recientes, se ha introducido el concepto de segmentación semántica, la cual ha sido hecha mediante diversas técnicas de Aprendizaje Profundo para contrarrestar los factores antes mencionados, como en el caso de [14], donde se realizó una segmentación semántica de imágenes a color mediante Redes Totalmente Convolucionales (*Fully Convolutional Networks*, FCN), junto a capas de deconvolución para obtener una salida precisa de píxeles (densificación de las regiones segmentadas). También en el trabajo de [12] se implementó una Red Neuronal Deconvolutiva después de las últimas capas de una CNN muy profunda de 16 capas, obteniendo mejores resultados que el trabajo anterior.

- 200 -




Figura 4.1. Primera página del artículo realizado para el CITID 2017.



Figura 4.2. Constancia de participación en el CITID 2017.

7. Póster para la Escuela de Inteligencia Artificial y Robótica

Se realizó un póster y una presentación con el tema de tesis de este trabajo para la Escuela de Inteligencia Artificial y Robótica llevada a cabo en la Universidad Tecnológica Emiliano Zapata (UTEZ) en octubre de 2017. En la Figura 4.3 se presenta el póster que se realizó, así como el reconocimiento obtenido en la Figura 4.4.



Segmentación de imágenes a color utilizando técnicas de Aprendizaje Profundo

Suárez Santiago Diego Gabriel,
Asesor: Dr. Dante Mújica Vargas
Centro Nacional de Investigación y Desarrollo Tecnológico

RESUMEN

En el presente trabajo se entrenó una Red Neuronal Convocional, de tipo Totalmente Convocional de paso 8 (FCN-F8), con el repositorio BSDS500 para crear un modelo capaz de segmentar imágenes naturales a color. Este repositorio se distingue por tener imágenes difíciles de segmentar y cuyas imágenes no han sido utilizadas con este tipo de técnica de Aprendizaje Profundo. El entrenamiento, validación y prueba de la red se hizo mediante la Unidad de Procesamiento Gráfico (GPU) para minimizar el tiempo y hacer factible su implementación. Las segmentaciones inferidas se evaluaron con las métricas de Índice Rand Probabilístico (Probabilistic Rand Index, PRI), Variación de Información (Variation of Information, VI) y Error de Consistencia Global (Global Consistency Error, GCE), donde se obtuvo una evaluación cuantitativa aceptable y competitiva con los modelos originales para segmentar los repositorios de PASCAL VOC y SIFT FLOW, así como con el modelo original FCN-Alexnet.

INTRODUCCIÓN:

La segmentación de imágenes es una de las tareas más complejas del procesamiento digital de imágenes, debido a que existe una serie de factores que la dificultan. Por ejemplo: las variaciones de intensidad o color, iluminación no uniforme, texturas parecidas, sombras y reflexión, objetos con bordes suavizados o difuminados con el fondo, traslapamiento de objetos y oclusión. Por consiguiente, la segmentación es una de las partes más importantes en la cadena de procesamiento de imágenes (Jähne, 2002), ya que determina el eventual éxito o fracaso de todo el proceso de análisis de la imagen (Marfin, 2004).

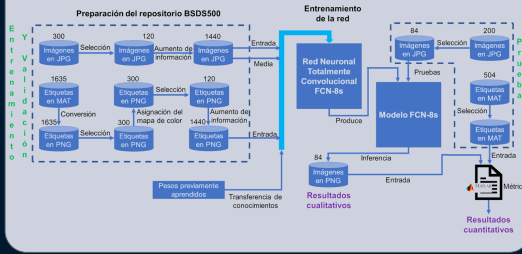
Dado lo anterior, se propuso la implementación de una Red Neuronal Convocional basada en técnicas de Aprendizaje Profundo para segmentar imágenes a color. Este tipo de redes tiene la ventaja de ser robustos a ciertas invarianzas, según lo menciona (LeCun *et al.*, 1998), (Zeiler y Fergus, 2013) y (Girshick *et al.*, 2014).

OBJETIVO:

- Segmentar imágenes a color utilizando una Red Neural Convocional implementada en una Unidad de Procesamiento Gráfico.

METODOLOGÍA:

Primero se preparó el repositorio BSDS500, como se aprecia en diagrama de abajo. Posteriormente, se utilizó dichas imágenes y etiquetas de entrenamiento y validación para entrenar la Red Neuronal Totalmente Convocional, llamado FCN-8s, probando diferentes grados de aprendizaje y métodos de optimización. También se probó con diferentes transformaciones geométricas aplicadas a las imágenes y etiquetas de entrenamiento y validación para aumentar la información, así como también se implementó la transferencia de conocimientos y se trabajó con la media del repositorio BSDS500, para obtener una mejora en la precisión de la segmentación.



RESULTADOS:

Imagen original

Segmentación verdadera

Segmentación por Gotthwal *et al.*, 2014

Segmentación inferida

Imagen original

Segmentación verdadera

Nekrasov *et al.*, 2016

FCN-8s

De forma cualitativa, como se aprecia en la imagen de arriba, las imágenes segmentadas con el modelo FCN-8s, el cual fue entrenado con las imágenes del repositorio BSDS500, realiza una segmentación buena y un poco mejor, visualmente, que los métodos propuesto por (Gotthwal *et al.*, 2014) y (Nekrasov *et al.*, 2016). Ahora bien, ya una vez probado con la media del BSDS500, con una tasa de aprendizaje base de $1e-16$ y con el método de optimización del Gradiente Descendiente Estocástico (Stochastic Gradient Descent, SGD), de forma cuantitativa alcanzó una precisión de 0.550 en la métrica PRI, de 2.460 en la métrica VI y 0.093 en la métrica GCE, tal como se aprecia en la tabla de abajo, en donde se compara con los modelos originales para PASCAL VOC y SIFT FLOW, resultando un poco mejor que los modelos mencionados.

Clase pájaro y persona	Modelo	Métrica PRI	Métrica VI	Métrica GCE
	Segmentación manual (verdadera)	0.555	2.183	0.020
	Modelo FCN-Alexnet	0.540	2.600	0.1324
	Modelo original para PASCAL VOC	0.542	2.516	0.100
	Modelo original para SIFT FLOW	0.538	2.488	0.091
	Modelo FCN-8s con media	0.550	2.460	0.093

REFERENCIAS:

- Girshick, R., Donahue, J., Darrell, T., and Malik, J. (2014). Rich feature hierarchies for accurate object detection and semantic segmentation. 21.
- Gotthwal, R., Gupta, S., Gupta, D., and Kumar, A. (2014). Color image segmentation algorithm based on rgb channels. p. 5.
- Jähne, B. (2002). Digital Image Processing. Springer.
- LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. (1998). Gradient-based learning applied to document recognition. Proc. of the IEEE, p. 46.
- Martin, D., Fowlkes, C., Tal, D., and Malik, J. (2001). A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. In Computer Vision, 2001. ICCV 2001. Proceedings. Eighth IEEE International Conference on, volume 2, pp. 416-423. IEEE.
- Nekrasov, V., Ju, J., and Choi, J. (2016). Global deconvolutional networks for semantic segmentation. arXiv preprint arXiv:1602.03930.
- Zeiler, M. and Fergus, R. (2013). Visualizing and understanding convolutional networks. Dept. of Computer Science, Courant Institute, p. 11.

Figura 4.3. Póster presentado en la Escuela de Inteligencia Artificial y Robótica.

UNIVERSIDAD TECNOLÓGICA
EMILIANO ZAPATA DEL ESTADO DE MORELOS
ORGANISMO PÚBLICO DESCENTRALIZADO DEL GOBIERNO DEL ESTADO DE MORELOS

otorga el presente

RECONOCIMIENTO

A: Suárez Santiago Diego Gabriel

Por su participación como ponente, con la conferencia:
**"Segmentación de Imágenes a Color Utilizando
Técnicas de Aprendizaje Profundo"**,
en el marco de actividades de la Escuela de Inteligencia Artificial
y Robótica 2017, realizada los días 19 y 20 de octubre del
presente año en las instalaciones de esta Universidad Tecnológica.

Emiliano Zapata, Mor., octubre 2017



M.C. Jaime Vázquez Colín
Director de la División Académica de
Mecánica Industrial



Figura 4.4. Reconocimiento por participar en la Escuela de Inteligencia Artificial y Robótica.

4.3. Aportaciones y mejoras realizadas


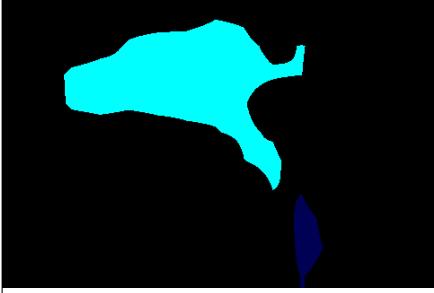



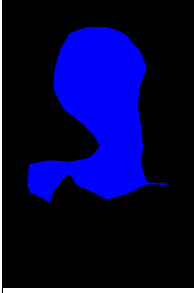
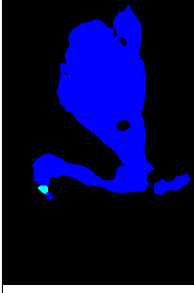
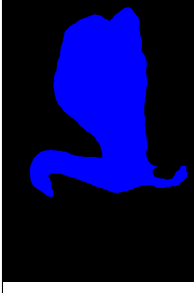
Una de las aportaciones de esta tesis es ofrecer un antecedente para aquellos trabajos en los que se requieren entrenar una Red Neuronal Convolutiva a partir de un repositorio propio, ya que se describe los procedimientos a seguir para poder hacerlo, además de proveer de otros métodos necesarios para aumentar la precisión en la segmentación de imágenes a color, junto con las métricas para evaluarla.

Otra de las aportaciones de este trabajo fue el de haber seleccionado una de entre todas las etiquetas disponibles por imagen para entrenamiento y validación, tomando como criterio que los objetos estuvieran lo menos particionado posible y que no se asignaran regiones que no le pertenecieran a un objeto. También se creó una lista de 116 objetos encontrados en el repositorio BSDS500. Así mismo, también se hizo una recopilación, tanto de las imágenes como de las etiquetas, que contuvieran las clases persona y ave, siendo estos más fácilmente disponibles para aquellos trabajos que requieran utilizar estas clases de objetos.

Por último, se adjunta a esta tesis los programas en MATLAB que contienen las métricas PRI, VI y GCE para evaluar la precisión en la segmentación de imágenes del repositorio BSDS500.

En cuanto a las mejoras realizadas con respecto a los modelos originales, el FCN-Alexnet y el FCN-8s, se puede ver en la Tabla 4.3, en donde se aprecia que las segmentaciones realizadas a las dos clases utilizadas, persona y pájaro, con el modelo FCN-Alexnet son muy burdas. Mientras tanto, con el modelo FCN-8s se mejora considerablemente la forma de las segmentaciones, aunque presenta cierta rigidez en los bordes curvos de las regiones segmentadas. Por último, con el FCN-8s modificado y propuesto en este trabajo para segmentar específicamente las imágenes del BSDS500, se ven más suaves los bordes curvos y sobre todo en las medidas cuantitativas se observa más claramente las mejoras en comparación a los dos modelos base originales, superándolos en las tres métricas tomadas en cuenta para evaluar la segmentación.

Tabla 4.3. Mejoras con respecto a los dos modelos originales utilizados

	Segmentación manual (real)	FCN-Alexnet	FCN-8s original	FCN-8s propuesto
Clase persona				
Clase pájaro				
PRI	0.555	0.540	0.542	0.550
VI	2.183	2.600	2.516	2.460
GCE	0.020	0.132	0.100	0.093

4.4. Conclusiones generales

Se concluye que las Redes Neuronales Convolucionales necesitan un repositorio con un gran número de imágenes por clase para ser capaz de aprender. También, estas redes son mayormente implementadas y alcanza su mayor efectividad en la tarea de clasificación de imágenes, y en los trabajos en donde se utilizan para la segmentación, se implementan en conjunto con otros métodos, como son los Campos Aleatorios Condicionales de (Noh *et al.*, 2015), las hipercolumnas propuesta por (Hariharan *et al.*, 2015), el Algoritmo Auto-Adaptativo de (Wang *et al.*, 2017), entre otros más.

En cuanto al pre-procesamiento del repositorio BSDS500 mencionado en la sección 3.6, se concluye que se debe conseguir o crear un repositorio con más de mil imágenes, con un número pequeño de clases, a lo mucho veinte clases de objetos, para que haya un número considerable de imágenes por clase de objetos. Además, si se tienen varias etiquetas (segmentaciones manuales) por imagen, también se debe de elegir una con la cual se entrenará la red, o tener una sola etiqueta por imagen aunque se evaluara la segmentación con otras métricas distintas a las presentadas en este trabajo de tesis, así como obtener etiquetas con una paleta de colores similar a la presentada en la Tabla 3.1 y en formato PNG, ya que este formato es el que se utiliza mayormente en los diferentes entornos de trabajo para Aprendizaje Profundo.

La sintonización realizada en la experimentación, específicamente la realizada en el experimento 9, fue la que mejor resultado dio para entrenar la Red Neuronal Convolutional FCN-8s con el repositorio BSDS500, tomando sólo las clases de ave y persona para entrenar, validar y probar el modelo mencionado. Si se requiere utilizar el modelo FCN-8s con otros repositorios que contengan imágenes naturales a colores, habrá que obtener la media del repositorio a utilizar para asignárselo a los archivos de configuración de la red y así mejorar los resultados de la segmentación y para corroborar el rendimiento se podría cambiar los grados de aprendizaje entre un rango de $1e - 16$ a $1e - 12$.

El método de aumento de información no aumentó significativamente la precisión en la segmentación. También, se pudo observar una diferencia entre usar y no la transferencia de conocimiento, ya que al no utilizarlo sólo mostraba imágenes en negro como inferencias del modelo.

Por último, según la prueba de significancia estadística realizada en el capítulo anterior, los resultados obtenidos en la experimentación sí son estadísticamente significativos como para decir que aumentó la precisión al utilizar la media del BSDS500 con el modelo FCN-8s, pero sólo con la métrica PRI, mientras que con las otras dos métricas (VI y GCE) no hubieron evidencias estadísticamente significativa que aumentara la precisión al utilizar la media del BSDS500 durante el entrenamiento del modelo FCN-8s.

4.5. Trabajos futuros

A continuación, se mencionan los posibles trabajos que puede desarrollarse a partir de este trabajo de tesis:

- Se podría trabajar en conjunto con otro método que aumente la precisión en la segmentación de imágenes, como algunos de los mencionados anteriormente.
- Otra mejora muy significativa sería aumentar la velocidad de procesamiento o rendimiento de las redes, reduciendo aún más el tiempo de entrenamiento, validación y pruebas, mediante la paralelización desde software; ya que según la Ley de Amdahl (Patterson *et al.*, 2004), es el algoritmo el que decide la mejora de velocidad, no el número de procesadores, debido a que estos últimos sólo aumenta la velocidad de la parte del sistema que está diseñado para trabajar en paralelo, mientras que el resto trabajaría de forma secuencial. La siguiente ecuación describe lo anterior:

$$\text{Incremento de velocidad} = \frac{1}{(1 - P)} \quad (4.1)$$

Donde P es el porcentaje del código que sí se puede paralelizar. Por ejemplo, supongamos que tenemos un problema y que un 30 % del algoritmo puede ser paralelizado, mientras el 70 % restante sigue siendo secuencial, la ley de Amdahl dice que al paralelizar al máximo el programa la nueva versión será $1/(1 - 0,3) = 1,4285$ veces más rápida que la versión no paralelizada.

- También podría ser posible simplificar algunas de las capas que conforman a la red, para disminuir la cantidad de operaciones que realiza.
- Este trabajo se puede tomar de base para experimentar y definir una propia capa que realizará una tarea en específico, así como implementar propias métricas en alguna capa, para no tener que ser evaluado desde afuera con otras clases, lenguajes o aplicaciones; o una capa que pre-procesara el repositorio BSDS500 sin tener que utilizar MATLAB, como se hizo en este trabajo.
- Así mismo, se podría trabajar en implementar los modelos creados en dispositivos móviles o en drones, para tareas específicas.
- Igualmente, se podría crear una página web para implementar el modelo creado.
- Incluso, se podría crear un repositorio propio desde cero, para ver el rendimiento de las Redes Neuronales Convolucionales y resolver un problema específico.
- Además, se podría trabajar en automatizar la sintonización de los hiper-parámetros que conforman a la Red Neuronal Convolutiva, para encontrar los valores que mejor resultados den al segmentar imágenes de cierto repositorio.
- Por último, se podría retomar este trabajo e implementar alguna de las posibles mejoras descritas anteriormente, para realizar el doctorado.

Bibliografía

- Amaratunga, T. (2017). How deep should it be to be called deep learning? <http://www.codesofinterest.com/2017/04/how-deep-should-deep-learning-be.html>.
- Amin, V. (2016). Data-first machine learning. <https://insidebigdata.com/2016/09/01/data-first-machine-learning/>.
- Araujo, L. (2016a). Convolutional neural networks. https://leonardoaraujosantos.gitbooks.io/artificial-inteligence/content/convolutional_neural_networks.html.
- Araujo, L. (2016b). Pooling layer. https://leonardoaraujosantos.gitbooks.io/artificial-inteligence/content/pooling_layer.html.
- Arbeláez, P., Pont-Tuset, J., Barron, J. T., Marques, F., y Malik, J. (2014). Multiscale combinatorial grouping. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 328–335.
- Behnke, S. (2016). Deep learning group. https://www.ais.uni-bonn.de/deep_learning/.
- Berkeley (2011). Repositorio bsds500. http://www.eecs.berkeley.edu/Research/Projects/CS/vision/grouping/BSR/BSR_bsds500.tgz.
- Berkeley (2013). Contour detection and image segmentation resources. <http://www.eecs.berkeley.edu/Research/Projects/CS/vision/grouping/resources.html#bench>.
- Britz, D. (2015). Deep learning glossary. <http://www.wildml.com/deep-learning-glossary/>.
- Bull, G., Gao, J., y Antolovich, M. (2015). Image Segmentation Using Random Feature. *Fifth International Conference on Graphic and Image Processing*, p. 8.
- BVLC (2016). Diy deep learning for vision with caffe. https://docs.google.com/presentation/d/1UeKXVgRvvxg9OUdh_UiC5G71UMscNPlvArsWER41PsU/preview?pref=2&pli=1&slide=id.g129385c8da_651_0.
- Cardenas (2015). Implementación y evaluación de redes neuronales artificiales tipo “Pulse-Couple Neural Network” (PCNN) aplicadas a visión artificial. Tesis de Maestría, Centro Nacional de Investigación y Desarrollo Tecnológico.

- Cervantes, S. (2006). Metodología para la estructuración y uso de conocimiento en segmentación de imágenes digitales. Tesis de Maestría, Centro Nacional de Investigación y Desarrollo Tecnológico.
- Cervantes, S. (2014). Localización de regiones para el reconocimiento de objetos en imágenes. Tesis de Maestría, Centro Nacional de Investigación y Desarrollo Tecnológico.
- Chen, T., Goodfellow, I., y Shlens, J. (2015). Net2net: Accelerating learning via knowledge transfer. *arXiv preprint arXiv:1511.05641*.
- Deeplearning4j (2016). Convolutional networks in java. <http://deeplearning4j.org/convolutionalnets.html>.
- Duarte, A. (2006). Segmentación Evolutiva de Imágenes utilizando Algoritmos Sociales Jerárquicos. *ResearchGate*, p. 9.
- Eckroth, J. (2014). Image convolutions and transformations. <http://cse3521.artifice.cc/image-convolutions-transformations.html>.
- Fergus, R. (2013). Deep learning for computer vision. <http://media.nips.cc/Conferences/2013/Video/Tutorial1A.pdf>.
- Fisher, R. A. (1934). Statistical methods for research workers.
- Garcia-Peraza, L., Li, W., Gruijthuijsen, C., Devreker, A., Attilakos, G., Deprest, J., Vander Poorten, E., Stoyanov, D., Vercauteren, T., y Ourselin, S. (2016). Real-Time Segmentation of Non-Rigid Surgical Tools based on Deep Learning and Tracking. In *Lecture Notes in Computer Science*. Springer Verlag (Germany).
- Gibiansky, A. (2014). Convolutional neural networks. <http://andrew.gibiansky.com/blog/machine-learning/convolutional-neural-networks/>.
- Ginzburg, B. (2014a). Lecture 2. caffe: getting started forward propagation. http://courses.cs.tau.ac.il/Caffe_workshop/Bootcamp/pdf_lectures/Lecture%202%20Caffe%20-%20getting%20started.pdf.
- Ginzburg, B. (2014b). Lecture 5. cnn: Regularization. http://courses.cs.tau.ac.il/Caffe_workshop/Bootcamp/pdf_lectures/Lecture%205%20CNN%20-%20dropout.pdf.
- Girshick, R., Donahue, J., Darrell, T., y Malik, J. (2014). Rich feature hierarchies for accurate object detection and semantic segmentation. *21*.
- Gothwal, R., Gupta, S., Gupta, D., y Dahiya, A. K. (2014). Color image segmentation algorithm based on RGB channels. In *Reliability, Infocom Technologies and Optimization (ICRITO)(Trends and Future Directions), 2014 3rd International Conference on*, pp. 1–5. IEEE.
- Gutstein, S., Fuentes, O., y Freudenthal, E. (2008). Knowledge transfer in deep convolutional neural nets. *International Journal on Artificial Intelligence Tools*, 17(03):555–567.

- Hariharan, B., Arbeláez, P., Girshick, R., y Malik, J. (2014). Simultaneous detection and segmentation. In *European Conference on Computer Vision*, pp. 297–312. Springer.
- Hariharan, B., Arbeláez, P., Girshick, R., y Malik, J. (2015). Hypercolumns for object segmentation and fine-grained localization. *arXiv*, p. 10.
- Heaton, J. (2015). *Artificial Intelligence for Humans, Volume 3: Deep Learning and Neural Networks*. Heaton Research.
- Hernández, J. (2003). Implementación de una red neuronal holográfica para el control de un brazo robot articulado. Tesis de Maestría, Centro Nacional de Investigación y Desarrollo Tecnológico.
- Hu, S., Williams, C., y Todorovic, S. (2015). Tree-Cut for Probabilistic Image Segmentation. *Servidor de artículos científicos arXiv*, p. 9.
- Huang, J. (2015). Convolutional neural networks. <http://www.slideshare.net/jbhuang/lecture-29-convolutional-neural-networks-computer-vision-spring2015>.
- Ikeuchi, K. (2014). *Computer vision: A reference guide*. Springer Publishing Company, Incorporated.
- Jähne, B. (2002). *Digital Image Processing*. Springer.
- Jia, Y. y Shelhamer, E. (2014a). Layers of caffe. <http://caffe.berkeleyvision.org/tutorial/layers.html>.
- Jia, Y. y Shelhamer, E. (2014b). Loss in caffe. <http://caffe.berkeleyvision.org/tutorial/loss.html>.
- Jiang, Y. y Ma, J. (2014). Fast and effective image segmentation via superpixels and adaptive thresholding. In *International Symposium on Neural Networks*, pp. 568–575. Springer.
- Johnson, J. (2016). Cs231n convolutional neural networks for visual recognition. <http://cs231n.github.io/convolutional-networks/>.
- Kishore, A., Jindal, S., y Singh, S. (2015). Designing Deep Learning Neural Networks using Caffe. *Manipal University*, p. 17.
- Krizhevsky, A., Sutskever, I., y Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pp. 1097–1105.
- Lü, C., Yang, X., y Qi, S. (2015). Color Image Segmentation Based on the Ant Colony Algorithm. *8th International Congress on Image and Signal Processing (CISP 2015)*, p. 2015.
- LeCun, Y., Bottou, L., Bengio, Y., y Haffner, P. (1998). Gradient-Based Learning Applied to Document Recognition. *Proc. of the IEEE*, p. 46.

- Lee, H., Grosse, R., Ranganath, R., y Ng, A. Y. (2009). Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations. In *Proceedings of the 26th annual international conference on machine learning*, pp. 609–616. ACM.
- Leónard, N. (2013). Preprocess the ground. <http://dp.readthedocs.io/en/master/preprocess/>.
- Liang, M. y Hu, X. (2015). Recurrent convolutional neural network for object recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3367–3375.
- Liu, Y., Stultz, C., Gutttag, J., Chuang, K.-T., Liang, F.-W., y Su, H.-J. (2016). Transferring Knowledge from Text to Predict Disease Onset. In *Machine Learning for Healthcare Conference*, pp. 150–163.
- Long, J., Shelhamer, E., y Darrell, T. (2015). Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3431–3440.
- Martin, D., Fowlkes, C., Tal, D., y Malik, J. (2001). A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. In *Computer Vision, 2001. ICCV 2001. Proceedings. Eighth IEEE International Conference on*, volume 2, pp. 416–423. IEEE.
- Martín, M. (2004). Técnicas Clásicas de Segmentación de Imagen. *Laboratory of Mathematics in Imaging*, p. 23.
- MathWorks (2016). softmax. <https://www.mathworks.com/help/nnet/ref/softmax.html>.
- MATLAB (2017a). Colormap. <https://www.mathworks.com/help/matlab/ref/colormap.html>.
- MATLAB (2017b). Transfer learning using convolutional neural networks. <https://www.mathworks.com/help/nnet/examples/transfer-learning-using-convolutional-neural-networks.html>.
- Matuz, M. (2016). Análisis y detección de anormalidades en mamografías utilizando redes neuronales convolucionales. Tesis de Maestría, Centro Nacional de Investigación y Desarrollo Tecnológico.
- Meilă, M. (2005). Comparing clusterings: an axiomatic view. In *Proceedings of the 22nd international conference on Machine learning*, pp. 577–584. ACM.
- Monterrey Gutiérrez, P. (2012). π_0 , 05 , ¿Criterio mágico para resolver cualquier problema o leyenda urbana? *Universitas Scientiarum*, 17(2).
- Morales, E. (2016). Deep learning. <http://ccc.inaoep.mx/~emorales/Cursos/Aprendizaje2/Acetatos/deeplearning.pdf>.

- Muñoz, J. (2016). Capítulo 6. segmentación de imágenes. http://www.lcc.uma.es/~munozp/documentos/procesamiento_de_imagenes/temas/pi_cap6.pdf.
- Murugan, P. (2017). Hyperparameters optimization in deep convolutional neural network/bayesian approach with gaussian process prior. *arXiv preprint arXiv:1712.07233*.
- Ng, A. (2016). Deep learning. <http://cs229.stanford.edu/materials/CS229-DeepLearning.pdf>.
- Nielsen, M. (2015). Neural networks and deep learning. <http://neuralnetworksanddeeplearning.com/chap6.html>.
- Noh, H., Hong, S., y Han, B. (2015). Learning deconvolution network for semantic segmentation. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 1520–1528.
- NVIDIA (2016a). Qué es computación acelerada por la gpu. <http://la.nvidia.com/object/what-is-gpu-computing-la.html>.
- NVIDIA (2016b). Using digits to train a semantic segmentation neural network. <https://github.com/NVIDIA/DIGITS/tree/master/examples/semantic-segmentation>.
- Ontiveros, N. (1995). Desarrollo de un sistema diagnosticador general de señales gráficas basado en la tecnología de redes neuronales. Tesis de Maestría, Centro Nacional de Investigación y Desarrollo Tecnológico.
- Oquab, M., Bottou, L., Laptev, I., y Sivic, J. (2014). Learning and transferring mid-level image representations using convolutional neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1717–1724.
- Patterson, D., Hennessy, J., Larus, J., y Corretger, R. (2004). *Estructura y diseño de computadores: interficie circuitería-programación*. Number v. 1 in Estructura y diseño de computadores. Reverté.
- Pérez, J. (2000). Compresión de imágenes usando redes neuronales artificiales recurrentes. Tesis de Maestría, Centro Nacional de Investigación y Desarrollo Tecnológico.
- Ramasubramanian, K. (2017). *Machine learning using R*. Apress, Place of publication not identified.
- Rand, W. M. (1971). Objective criteria for the evaluation of clustering methods. *Journal of the American Statistical association*, 66(336):846–850.
- Rocha, M. (1999). Reconocimiento de patrones de fallas en generadores eléctricos empleando redes neuronales artificiales. Tesis de Maestría, Centro Nacional de Investigación y Desarrollo Tecnológico.
- Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., Berg, A. C., y Fei-Fei, L. (2015). ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, 115(3):211–252.

- Serna, N. L. y Román, U. (2009). Técnicas de Segmentación en Procesamiento Digital de Imágenes. *Revista de Ingeniería de Sistemas e Informática*, p. 8.
- Shan, S. (2016). Fully convolutional networks for semantic segmentation. <http://sintalk.cn/2016/11/01/Fully-Convolutional-Networks/>.
- Shelhamer, E., Long, J., y Darrell, T. (2017). Fully convolutional networks for semantic segmentation. *IEEE transactions on pattern analysis and machine intelligence*, 39(4):640–651.
- Sigut, J., Fumero, F., Nuñez, O., y Sigut, M. (2014). Automatic marker generation for watershed segmentation of natural images. *Electronics Letters*, 50(18):1281–1283.
- Simonyan, K. y Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*.
- Sánchez, O. (2009). Modelado del Comportamiento de Conducción de Vehículos de Transporte y/o Carga Aplicando Redes Neuronales Artificiales y Visión Artificial. Tesis de Maestría, Centro Nacional de Investigación y Desarrollo Tecnológico.
- Sterne, J. A. y Smith, G. D. (2001). Sifting the evidence—what’s wrong with significance tests? *Physical Therapy*, 81(8):1464–1469.
- Stutz, D., Hermans, A., y Leibe, B. (2014). Superpixel segmentation using depth information. *RWTH Aachen University, Aachen, Germany*, 4.
- Unnikrishnan, R. y Hebert, M. (2005). Measures of similarity. In *Application of Computer Vision, 2005. WACV/MOTIONS’05 Volume 1. Seventh IEEE Workshops on*, volume 1, pp. 394–394. IEEE.
- Vedaldi, A. y Lenc, K. (2015). Matconvnet: Convolutional neural networks for matlab. In *Proceedings of the 23rd ACM international conference on Multimedia*, pp. 689–692. ACM.
- Velarde, A. (1998). Sistema de visión artificial para la verificación del llenado de recipientes no opacos utilizando redes neuronales artificiales. Tesis de Maestría, Centro Nacional de Investigación y Desarrollo Tecnológico.
- Vélez, J., Moreno, A., Sánchez, ., y Sánchez, J. (2003). *Visión por computador*. José Vélez.
- Wang, M. (2015). Multi-path Convolutional Neural Networks for Complex Image Classification. *Dalhousie University*, p. 8.
- Wang, Y., Huang, Y., Zheng, W., Zhou, Z., Liu, D., y Lu, M. (2017). Combining convolutional neural network and self-adaptive algorithm to defeat synthetic multi-digit text-based CAPTCHA. In *Industrial Technology (ICIT), 2017 IEEE International Conference on*, pp. 980–985. IEEE.
- Xiaoyang, H. (2016). Deep learning and computer vision series (10) on convolutional neural network. http://prog3.com/sbdm/blog/han_xiaoyang/article/details/50542880.

-
- Yang, A., Wright, J., Ma, Y., y Sastry, S. (2007). Unsupervised segmentation of natural images via lossy data compression. https://people.eecs.berkeley.edu/~yang/software/lossy_segmentation/.
- Zeiler, M. y Fergus, R. (2013a). Visualizing and Understanding Convolutional Networks. *Dept. of Computer Science, Courant Institute*, p. 11.
- Zeiler, M. D. y Fergus, R. (2013b). Stochastic pooling for regularization of deep convolutional neural networks. *arXiv preprint arXiv:1301.3557*.

Anexo A. Hiperparámetros de la red

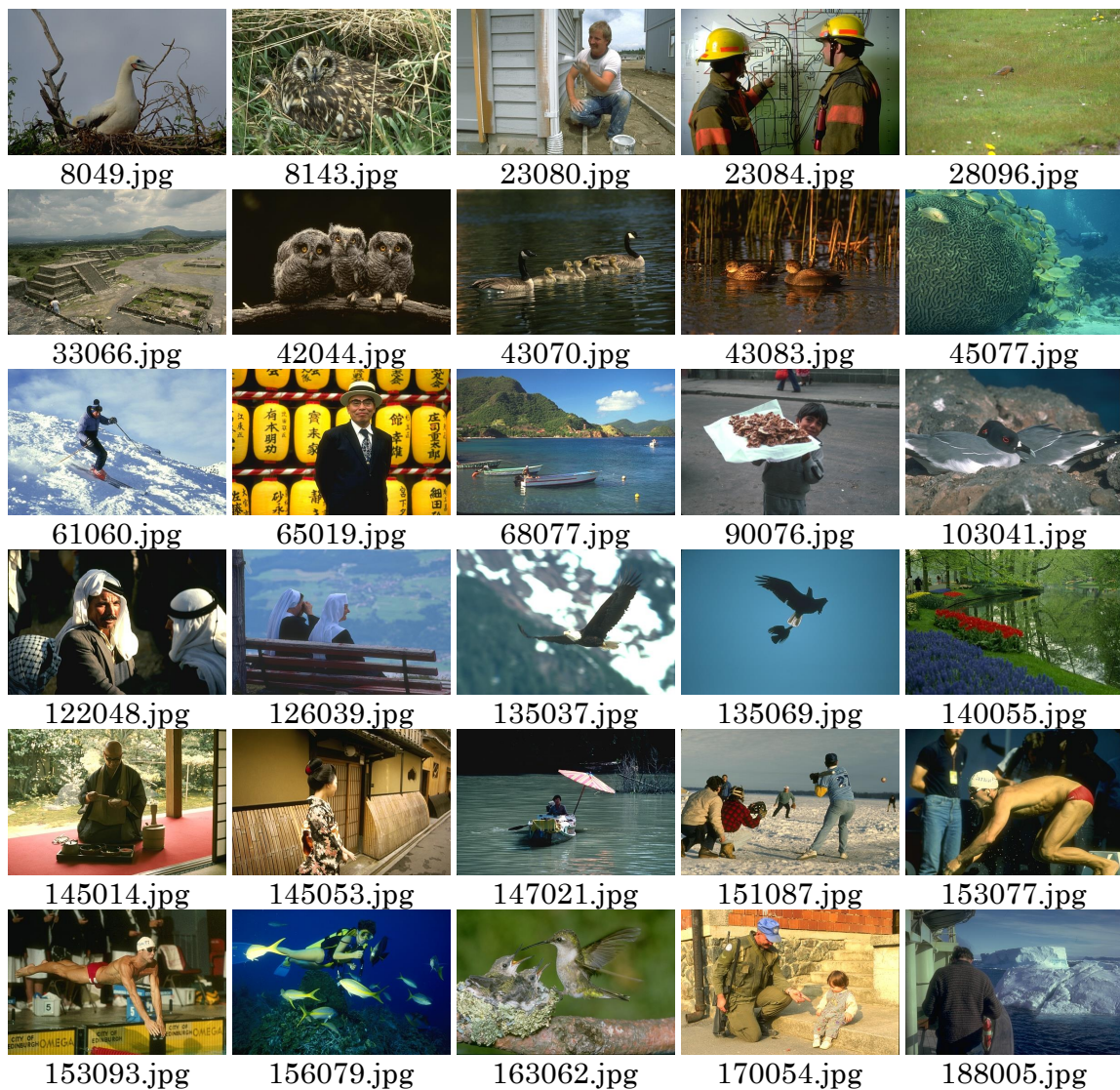
Significado de los hiperparámetros configurables para entrenar una red neuronal convolucional en Caffe:

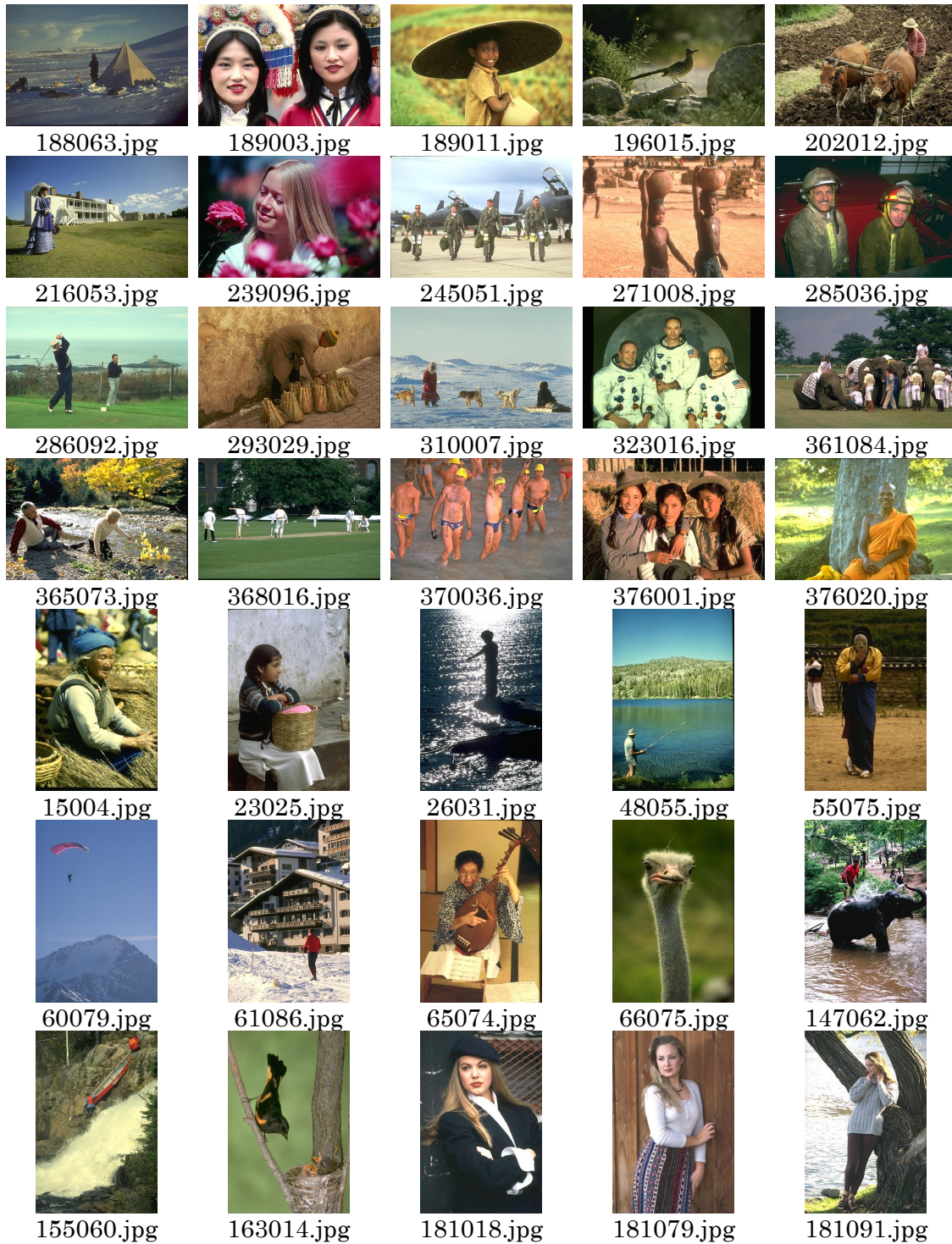
- **base_lr:** Este hiperparámetro indica la tasa de aprendizaje base (inicial) de la red. El valor es un número real (punto flotante).
- **lr_policy:** Este hiperparámetro indica cómo la tasa de aprendizaje debe cambiar con el tiempo. Este valor es una cadena entre comillas dobles.
- **gamma:** Este hiperparámetro indica cuánto debe cambiar la tasa de aprendizaje cada vez que se alcance el siguiente “paso”. El valor es un número real y se puede multiplicar la tasa de aprendizaje actual por dicho número para obtener una nueva tasa de aprendizaje.
- **stepsize:** Este hiperparámetro indica la frecuencia en iteraciones con la que debemos pasar al siguiente “paso” del entrenamiento. Este valor es un entero positivo.
- **stepvalue:** Este hiperparámetro indica uno de los potencialmente muchos conteos de iteración que debe pasar al siguiente “paso” de entrenamiento. Este valor es un entero positivo. A menudo, hay más de uno de estos hiperparámetros presentes, cada uno indica la siguiente etapa de iteración.
- **max_iter:** Este hiperparámetro indica cuando la red debe detener el entrenamiento. El valor es un entero que indica qué iteración debe ser la última.
- **momentum:** Este hiperparámetro indica cuánto del peso anterior será retenido en el nuevo cálculo. Este valor es una fracción real.
- **weight_decay:** Este hiperparámetro indica el factor de penalización (regularización) de grandes pesos. Este valor es a menudo una fracción real.
- **solver_mode:** Este hiperparámetro indica qué modo se utilizará para entrenar la red. Las opciones puede ser CPU o GPU
- **snapshot:** Este hiperparámetro indica la frecuencia en iteraciones con la que Caffe debe generar una copia del modelo (*caffemodel*) que se está entrenando junto con su archivo de configuración (*solverstate*). Este valor es un entero positivo.

- **snapshot_prefix:** Este hiperparámetro indica cómo se debe nombrar a un modelo y archivo de configuración generado.
- **net:** Este hiperparámetro indica la ubicación de la red a ser entrenada (ruta del archivo prototxt). Este valor es una cadena entre comillas dobles.
- **test_iter:** Este hiperparámetro indica cuántas iteraciones de prueba deben producirse por cada *test_interval*. Este valor es un entero positivo.
- **test_interval:** Este hiperparámetro indica la frecuencia con la que se ejecutará la fase de prueba de la red.
- **display:** Este hiperparámetro indica la frecuencia en iteraciones con la que Caffe debe mostrar resultados en consola. Este valor es un entero positivo y especifica un conteo de iteraciones.
- **type:** Este hiperparámetro indica el algoritmo de propagación del error hacia atrás que se utilizará para entrenar la red. Este valor es una cadena entre comillas.

Anexo B. Imágenes del BSDS500

86 imágenes para entrenamiento del repositorio BSDS500.



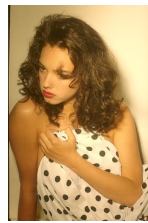




187029.jpg



187083.jpg



198004.jpg



198023.jpg



198054.jpg



225017.jpg



239007.jpg



242078.jpg



246016.jpg



246053.jpg



260081.jpg



268002.jpg



274007.jpg



292066.jpg



302003.jpg



311068.jpg



365025.jpg



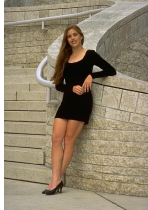
368078.jpg



372047.jpg

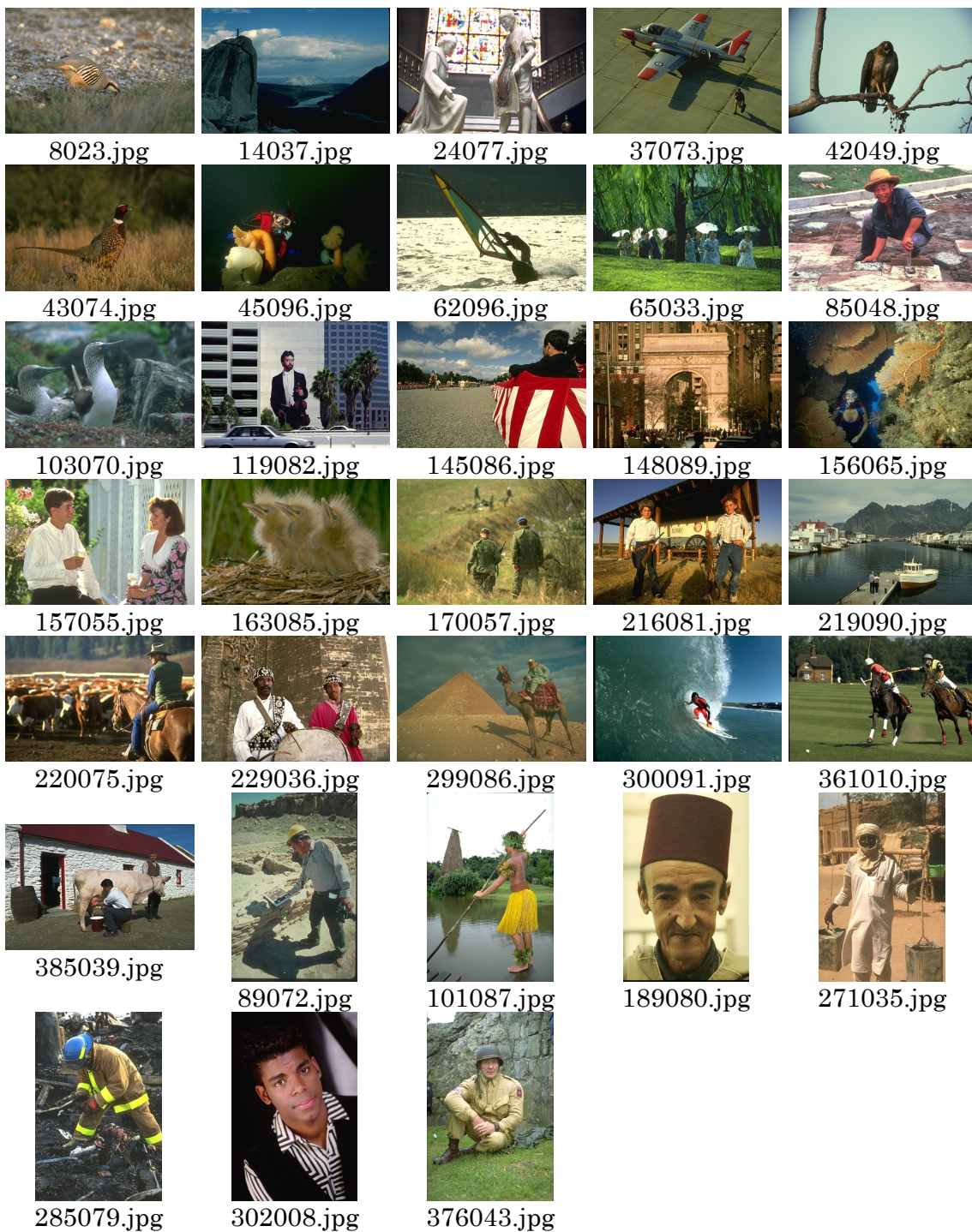


374067.jpg

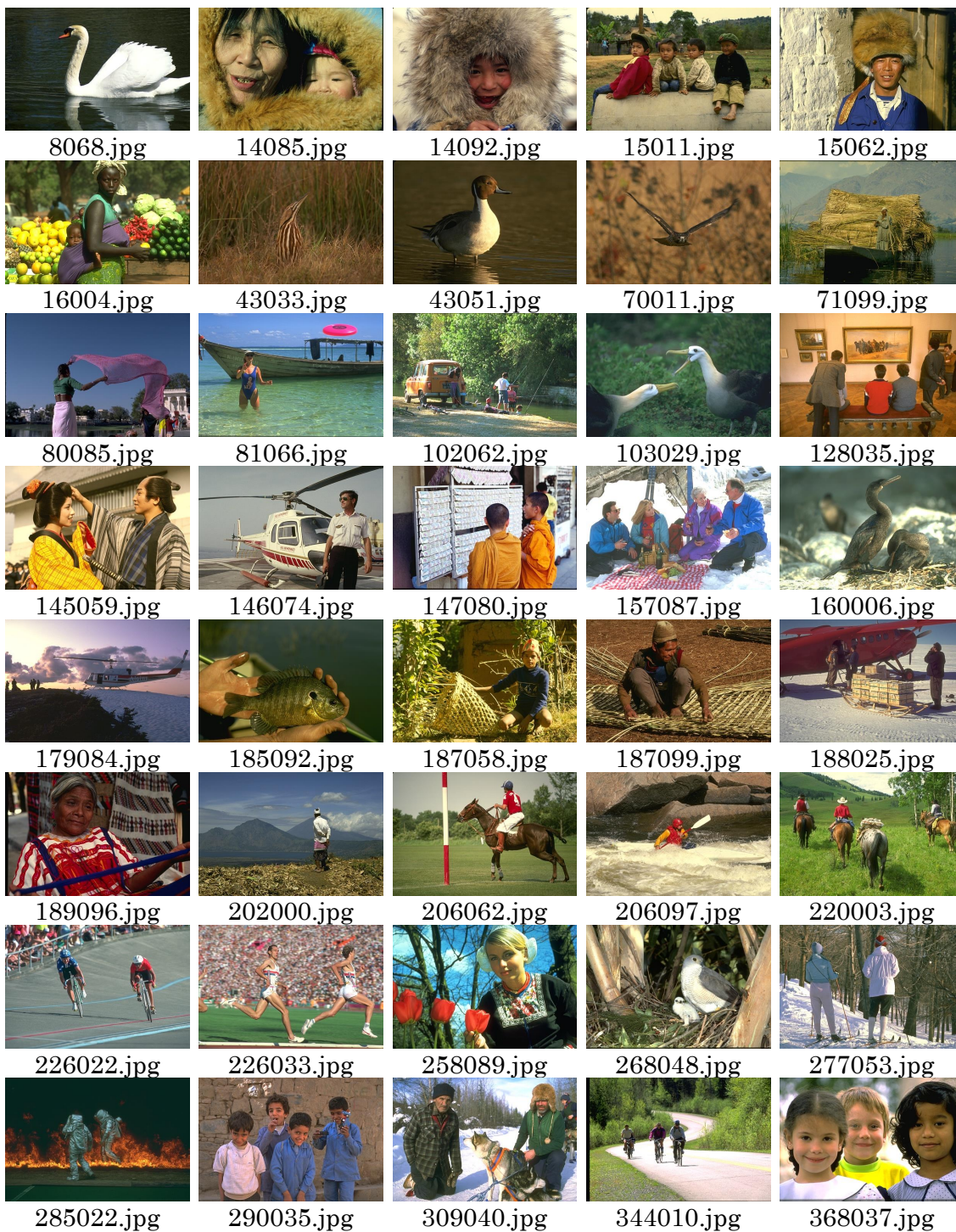


388016.jpg

33 imágenes para validación del repositorio BSDS500.



84 imágenes para prueba del repositorio BSDS500.





376086.jpg



20069.jpg



23050.jpg



64061.jpg



65084.jpg



70090.jpg



80090.jpg



81090.jpg



140006.jpg



147077.jpg



156054.jpg



163004.jpg



163096.jpg



181021.jpg



189006.jpg



189013.jpg



189029.jpg



196027.jpg



196040.jpg



198087.jpg



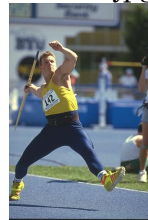
217013.jpg



225022.jpg



226043.jpg



226060.jpg



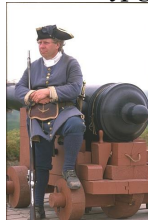
230063.jpg



230098.jpg



238025.jpg



243095.jpg



246009.jpg



249021.jpg



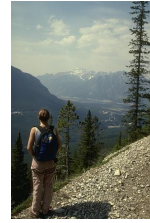
250047.jpg



250087.jpg



259060.jpg



267036.jpg



268074.jpg



279005.jpg



281017.jpg



289011.jpg



302022.jpg



365072.jpg



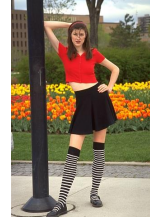
384089.jpg



388006.jpg



388018.jpg



388067.jpg

cenidet[®]
*Centro Nacional de Investigación
y Desarrollo Tecnológico*