UNIVERSIDAD DE CHILE
FACULTAD DE CIENCIAS FÍSICAS Y MATEMÁTICAS
DEPARTAMENTO DE INGENIERÍA ELÉCTRICA

SEGMENTACIÓN SEMÁNTICA Y RECONOCIMIENTO DE LUGARES USANDO CARACTERÍSTICAS CNN PRE-ENTRENADAS

TESIS PARA OPTAR AL GRADO DE MAGÍSTER EN CIENCIAS DE LA INGENIERÍA, MENCIÓN ELÉCTRICA

PEDRO IGNACIO ORELLANA RUEDA

PROFESOR GUÍA:
JAVIER RUIZ DEL SOLAR SAN MARTÍN

MIEMBROS DE LA COMISIÓN:
FELIPE TOBAR HENRÍQUEZ
RODRIGO VERSCHAE TANNENBAUM

SANTIAGO DE CHILE
2019

RESUMEN DE LA TESIS PARA OPTAR AL GRADO DE MAGÍSTER EN CIENCIAS DE LA INGENIERÍA, MENCIÓN ELÉCTRICA
POR: PEDRO IGNACIO ORELLANA RUEDA
FECHA: 2019
PROF. GUÍA: JAVIER RUIZ DEL SOLAR SAN MARTÍN

En el presente trabajo de tesis se propone e implementa un sistema de segmentación semántica y reconocimiento de interiores domésticos utilizando características pre-entrenadas extraídas de una red neural convolucional profunda. En particular este tipo de enfoque ha entregado buenos resultados en el problema de clasificación de imágenes, aunque no ha sido muy explotado en el problema de la segmentación semántica.

El problema de segmentación semántica es uno de los más desafiantes en el campo de la visión por computador, dada la complejidad técnica del problema en sí, y a la dificultad agregada de generar una base de datos etiquetada para poder entrenar los modelos. Debido a esto, en algunas de bases de datos pequeñas, como NYU Depth v1, métodos basados en características "*hand crafted*" aún siguen superando a enfoques basados en aprendizaje profundo "end-to-end". Es por este motivo que este trabajo de tesis busca explorar enfoques que permitan transferir el conocimiento aprendido por una red profunda a un problema con otra base de datos diferente. Esto permitirá entrenar modelos de segmentación semántica en bases de datos muy pequeñas como para obtener buenos resultados utilizando aprendizaje profundo.

En particular en este trabajo se extraen características de las últimas capas de una red Segnet entrenada en la base de datos de interiores domésticos SUN RGBD. El método propuesto para trabajar con estas características estaba basado en los trabajos "*hand crafted*" que han entregado mejores resultados a la fecha. Este método está compuesto por varias etapas, la primera es un algoritmo que divide en la imagen en múltiples zonas, a partir de la información de un detector de contornos. Luego cada una de estas zonas pasa a ser un segmento desde donde se calcula un vector de características a partir de la información extraída de la las últimas capas de Segnet. Finalmente existe una etapa de clasificación, a nivel de segmentos, compuesta por un SVM. De forma paralela se construye un reconocedor de lugares, utilizando también características extraídas de una red profunda. La idea de este reconocedor de lugares es que aporte con información de contexto de alto nivel al sistema de segmentación semántica.

Las contribuciones de este trabajo de tesis son específicamente tres. La primera de ellas es el uso transferencia de conocimiento de una red profunda aplicado al problema de segmentación semántico. La metodología utilizada para extraer las características es novedosa, pues hasta la realización de este trabajo no se habían utilizado enfoques como el propuesto para abordar el problema de segmentación semántica. Este enfoque es de especial utilidad en bases de datos muy pequeñas como para aplicar aprendizaje *profundo end-to-end* o *fine-tuning*. La segunda contribución es el uso de la información de un detector de contornos en conjunto con las características extraídas de una red profunda. Las características generadas por una red profunda tienen gran poder de separabilidad, incluso para realizar clasificación a nivel de pixeles de forma efectiva. Sin embargo en la frontera de dos objetos es donde se comente más errores de clasificación. El detector de contornos contribuye a disminuir los errores en las fronteras entre objetos. La última contribución de este trabajo de tesis es el uso de la información de un clasificador de lugares, información de contexto de alto nivel, para mejorar el resultado de la segmentación semántica. En conjunto estas tres contribuciones permitente mejorar los resultados a la fecha en la base de datos de segmentación semántica NYU Depth v1.

Agradecimientos

Agradecimientos a todos lo que permitieron de alguna u otra forma la realización de este trabajo de tesis, desde profesores hasta personal administrativo de la universidad. A mis compañeros y amigos del laboratorio de visión, desde su ayuda académica hasta los momentos de ocio jugando "robotitos". Gracias por la oportunidad que se me ha dado de realizar este postgrado, de aprender cosas que ni siquiera sabía que no sabía y de las cuales nunca voy a dejar de maravillarme. Gracias en especial a mis padres, que sin ellos nada de esto sería posible (pues técnicamente ellos crearon mi existencia (y unicidad)).

Tabla de contenido

1. Introducción 1
   1.1 Motivación 1
   1.2 Definición del problema y objetivos 2
      1.2.1 Objetivos generales 3
      1.2.2 Objetivos específicos 3
   1.3 Hipótesis 3
   1.4 Estructura del documento 4
2. Estado del arte 5
   2.1 Segmentación semántica 5
      2.1.1 Segmentación semántica y deep-learning 6
   2.2 Clasificación de lugares 9
      2.2.1 Rankings en bases de datos 10
      2.2.2 Transferencia de conocimiento desde redes CNN pre-entrenadas 18
      2.2.3 Descripción base de datos 21
   2.3 Discusión 25
3. Sistema propuesto 28
   3.1 Sub-sistema de segmentación semántica 28
      3.1.1 Etapa generación de superpixeles 30
      3.1.2 Extracción de características RGB 30
      3.1.3 Extracción de características Depth 35
      3.1.4 Clasificación 36
      3.1.5 Resultados 37
      3.1.6 Análisis de resultados 39
   3.2 Sub-sistema de clasificación de lugares y generación de contexto 39
      3.2.1 Clasificador de lugares 41
      3.2.2 Clasificación 43
      3.2.3 Resultados 44
      3.2.4 Análisis de resultados 45
      3.2.5 Uso de contexto de lugar en segmentación semántica 47
4. Pruebas, resultados y análisis 49
   4.1 Metodología para pruebas 49
   4.2 Resultados 49
      4.2.1 Segmentación semántica y uso de contexto 49
      4.2.2 Clasificación de lugares 50

4.2.3 Resultados cualitativos 51
4.3 Análisis de resultados 55
5. Conclusiones 58
6. Bibliografía 60

1. Introducción

Este trabajo de tesis trata del desarrollo de un sistema de segmentación semántico y clasificación de lugares, utilizando características extraídas desde redes *deep* previamente entrenadas. Para esto se explora la transferencia de conocimiento desde redes entrenadas en bases de datos distintas a la de problema a tratar y las posibles formas de extracción de características desde estas redes. Específicamente se aborda el problema de segmentación semántica en interiores domésticos, en la base de datos NYU Depth v1. Además se busca explorar la integración de la información de lugar en el problema de segmentación semántica, para mejorar los resultados de esta última. Tanto la información que entrega la segmentación semántica como la del clasificador de lugares, son cruciales para algún día desarrollar un robot de uso doméstico totalmente autónomo.

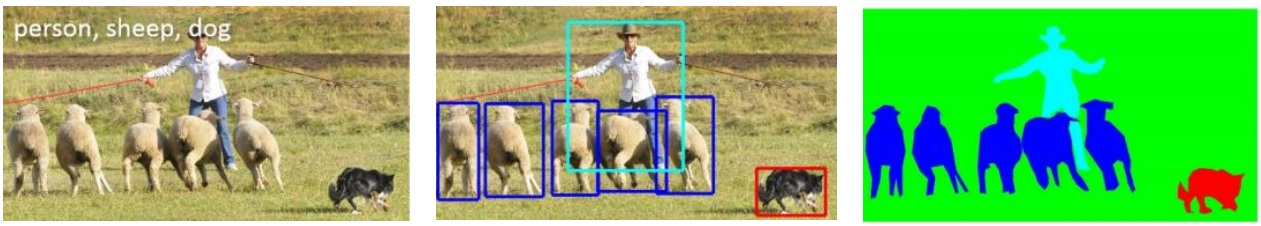
1.1 Motivación

Mientras los humanos pueden interpretar sin problemas complejas escenas, los algoritmos más sofisticados aún tienen dificultades para hacerlo. Sin embargo esto es un requisito si algún día se quiere alcanzar la implementación de robots totalmente autónomos. Este es un problema especialmente complejo en el caso de robots domésticos, donde existe una gran variabilidad entre la estructura física del interior de un hogar y otro, a diferencia por ejemplo, de la variabilidad entre caminos pavimentas de distintas carreteras.

Este trabajo se enfoca en el problema de segmentación semántica y en la clasificación de lugares, ambos en interiores domésticos. Estos sistemas son necesarios si se quiere un robot que sea capaz entender y realizar órdenes de alto nivel, como por ejemplo “ve a la living y trae el libro que esta sobre el sillón”. Por un lado existe el problema de navegación a nivel geométrico dentro del lugar. Para esto es necesario que el robot se localice y haga un mapa del entorno a nivel geométrico, lo cual corresponde al problema de SLAM. Este problema ha sido ampliamente estudiado y tratado, existiendo soluciones con un grado elevado de efectividad y aplicabilidad a situaciones reales. Sin embargo, para que el robot sea capaz de seguir la orden dada, y dejando de lado la parte de manipulación de objetos, aún es necesario que sea capaz de ubicarse a un nivel más alto de abstracción. Esto significa que sea capaz de “entender” que un área determinada corresponde a un living y que a su vez pueda reconocer distintas clases semánticas en la escena, como por ejemplo para este caso particular, un sillón. La solución ideal no solo debería ser capaz de reconocer estas clases semánticas, sino además determinar sus contornos, para poder interactuar con ellos. Esto último corresponde al problema de segmentación semántica. A su vez, para reconocer el living, debe ser capaz de clasificar lugares. Estos últimos dos problemas son claves para algún día llegar a obtener un robot de servicio doméstico totalmente autónomo capaz de seguir instrucciones de alto nivel.

1.2 Definición del problema y objetivos

Los principales focos de este trabajo son el problema de segmentación semántica y clasificación de lugares. El problema de segmentación semántica corresponde a asignar una determinada etiqueta o categoría de cada uno de los pixeles de una imagen. Este problema es más complejo que la clasificación o reconocimiento de una imagen completa (asignar una misma etiqueta a todos los pixeles de una imagen) o que la detección y reconocimiento de objetos mediante *bounding box* (donde existe un algoritmo para generar *proposals* de regiones rectangulares a evaluar, para luego tratar estas zonas de la misma forma que un problema de reconocimiento de imágenes). En el problema de segmentación semántica, además de detectar y reconocer los objetos, se busca segmentar sus fronteras de forma exacta.



a) *b)* *c)*

Figura 1-1 : a) reconocimiento de imágenes, b) detección y reconocimiento c) segmentación semántica

En cuanto al problema de clasificación de lugares en interiores domésticos, este es abordado como un problema de clasificación de imágenes. Esto es, asignar una etiqueta de una clase a una imagen determinada.

De forma intuitiva, el conocer en qué tipo de habitación uno se encuentra ayuda a saber qué objetos son más probables de encontrar. Es decir, si uno está en un dormitorio, a priori sabe que la probabilidad de encontrar una cama es más alta que la de encontrar un sillón, por ejemplo. Siguiendo esta idea, en este trabajo de tesis también se busca utilizar la información del clasificador de lugares para ayudar al sistema de clasificación semántica. De esta forma se espera mejorar el resultado de la segmentación semántica, que es el problema más complejo de entre los dos.

Un tercer foco de este trabajo es el uso de redes CNN pre-entrenadas como extractores de características. Existen múltiples trabajos en el área de visión que son capaces de obtener muy buenos resultados en una base de datos determinada, pero al ser probados en otras bases de datos, o en el mundo real, su desempeño disminuye de forma dramática [1] [2]. Este es un punto crítico para una aplicación destinada al mundo real, como podría ser un robot de servicio doméstico. Por este motivo son de particular interés las características extraídas a partir de redes CNN pre-entrenadas, pues han demostrado ser capaces de generalizar y entregar un alto desempeño en múltiples bases de datos [3]. Esto sin requerir ningún tipo de ajuste o entrenamiento extra para generar las características, a diferencia de los métodos *hand-crafted*. Los métodos *hand-crafted*, para entregar un alto rendimiento, generalmente necesitan aprender un codebook y/o ajustar múltiples parámetros, al momento de generar sus características. El uso de características CNN pre-entrenadas ha sido ampliamente estudiado y ha entregado muy buenos resultados [4][5][6] en el problema de clasificación de imágenes, sin embargo aún no se han visto resultados igual de buenos en el área de segmentación semántica utilizando este enfoque.

1.2.1 Objetivos generales

El objetivo de este trabajo es implementar un sistema de segmentación semántica usando características CNN pre-entrenadas e información de contexto de alto nivel, esta última obtenida de un clasificador de lugares.

1.2.2 Objetivos específicos

Los objetivos específicos de este trabajo de tesis son los siguientes:

- Extraer características útiles para el problema de segmentación a partir de alguna red *deep* previamente entrenada en alguna base de datos distinta a la del problema a tratar en esta tesis. Al mismo tiempo se busca validar la efectividad para generalizar de estas características.

- Implementar sistema de segmentación semántica basado en un enfoque híbrido, utilizando características extraídas de red *deep* CNN + clasificador SVM.

- Diseñar sistema para clasificar lugares utilizando características *deep* CNN pre entrenadas.

- Integrar la información del clasificador de lugares al sistema de segmentación semántica.

- Integrar la información de un sistema de detección de bordes para mejorar los resultados en los contornos de los objetos al momento de realizar la segmentación semántica.

- Evaluar el rendimiento del sistema propuesto en frente a otras las alternativas existentes, tales como utilizar características *hand-crafted*, red *deep* desde cero o realizar *fine tunning* en alguna red ya entrenada. Evaluar también el aporte de cada una de las etapas propuestas.

1.3 Hipótesis

Las hipótesis planteadas en este trabajo de tesis son las siguientes:

- La transferencia de conocimiento en problemas de segmentación semántica es un enfoque efectivo, análogo al utilizar esta misma técnica en problemas de clasificación de imágenes.
En el ámbito de clasificación de imágenes, diferentes trabajos han demostrado que es posible extraer características desde capas intermedias de una de una red *deep* previamente entrenada [4] [5] [6] y utilizarlas en un problema distinto, obteniendo resultados *state-of-the-art* [4]. De forma similar, las capas intermedias de una red *deep* para segmentar pueden usarse como características en otros problemas de segmentación semántica. La transferencia de conocimiento es de particular interés en bases de datos que no son muy grandes, donde no es posible utilizar métodos basados en *deep* learning con entrenamiento desde cero. Dado que la base de datos donde se trabajará es pequeña, se espera que estas características sean mejor, en relación al resultado del clasificador, que entrenar desde cero, o que el mejor método *hand-crafted* existente.

-La información de contexto de alto nivel generada por un clasificador de lugares, permite mejorar los resultados de la segmentación semántica.
La gran mayoría de los sistemas de segmentación semántica buscan explotar de alguna forma la información de contexto de bajo nivel, correspondiente a la información de los pixeles en la vecindad del pixel o segmento a clasificar. Sin embargo, la información de contexto de alto nivel, correspondiente al tipo de habitación de la imagen a segmentar, no ha sido igual de explotada. De forma intuitiva, si se conoce el tipo de habitación (cocina, dormitorio, etc…) de la imagen que se está segmentado, obtengo información relacionada con la probabilidad a priori de encontrar ciertos elementos. Es decir, si sé que la imagen que estoy segmentado corresponde a un comedor, la probabilidad de encontrar una mesa es más alta que la de encontrar una cama, por ejemplo. Esto sin duda es de ayuda para un sistema de segmentación semántica.

- La información que entrega un sistema detector de bordes es de ayuda incluso en sistemas basados en *deep* learning.
Una gran parte de los sistemas de segmentación semántica basados en métodos *hand crafted*, utilizan de alguna forma la información de algún sistema para detectar bordes. Por otro lado, los métodos basados en redes *deep* learning *end to end* han demostrado ser eficaces incluso sin usar este tipo información [7] [8] [9] [10]. Sin embargo, la frontera entre dos objetos distinto sigue siendo una de las partes donde se comenten mayores errores de clasificación [10]. Es por esto que se busca rescatar la etapa de detección de bordes utilizadas en los métodos basados en features *hand crafted* y utilizara en métodos basados con características *deep learning*.

1.4 Estructura del documento

Este documento tiene la siguiente estructura. En el Capítulo 2 se muestra la revisión bibliográfica y el estado del arte del problema de segmentación semántica, sobre todo los trabajos enfocados en interiores, y más específicamente los trabajos en NYU Depth V1. También se muestran el estado del arte respecto a la extracción de características de redes CNN, para la clasificación de imágenes. El Capítulo 3 detalla la implementación propuesta, tanto para el problema de segmentación semántica, como para el clasificador de lugares y la generación de contexto. En este capítulo, además, se hacen pruebas preliminares tanto para el problema de segmentación semántica como el de clasificación de lugares, con el objetivo de definir bien estas etapas. El Capítulo 4 muestra un análisis del rendimiento completo del sistema, en su conjunto, y lo compara con otros trabajos e implementaciones. Finalmente, el Capítulo 5 corresponde a las conclusiones en este trabajo.

2. Estado del arte

En este capítulo se muestra resumidamente el estado del arte del problema de segmentación semántica y clasificación de lugares. En particular para el caso de segmentación semántica, se pone énfasis en los métodos que trabajan con bases de datos de interiores domésticos, y se revisan con particular detalle los que están más relacionados con este trabajo de tesis.

2.1 Segmentación semántica

El primer enfoque en entregar buenos resultados en el problema de segmentación semántica es el que se utiliza en trabajos como los de Zemel *et al.* [11] y Shotton *et al.* [12] [13]. En estos trabajos se calculan características *hand-crafted* en patches rectangulares de la imagen, y luego se etiqueta el *patch* o el pixel central del *patch* con algún clasificador, como una red neuronal multicapa [11], boosting [12] o random forest [13]. El resultado de esta primera etapa de clasificación es bastante ruidoso, por lo que es necesario refinarlo. En la etapa de refinamiento se utiliza un modelo probabilístico en forma de grafo, el cual analiza la información de las vecindades para suavizar el resultado del clasificador. Esta última etapa corresponde generalmente a un MRF o un CRF [12] [13].

.

Figura 2-1 : Imágenes de resultados, con sus respectivos *inputs*, del trabajo de Shotton *et al.*[13]

Enfoques posteriores mejoraron la etapa de clasificación al etiquetar directamente todo un segmento de la imagen [14] [15] en vez de solo un patch o el pixel central. Estos *patch* son generados de forma que tengan relación con los contornos de los objetos a clasificar. Una de las formas de obtener estos segmentos es mediante algún método que genere *superpixeles*. Los *superpixeles* son grupos de pixeles similares entre sí, según criterios de color y gradiente. Con respecto a las características, los últimos trabajos *hand-crafted* han utilizados las que han demostrado ser más efectivas y populares en problemas de clasificación de imágenes, como LBP, HOG, SIFT [15], obteniendo también buenos resultados. En la búsqueda de características más discriminativas, se ha utilizado aprendizaje tanto supervisado como no supervisado para la generación de características. Estas etapas de aprendizaje dependen de etapas previas de

características *hand-crafted* calculadas de forma densa y son métodos en los cuales deben ajustarse varios hiperparámetros.

La tendencia en últimos trabajos de segmentación semántica es a utilizar enfoques basados en *deep* learning [10] [16] [7] [17] [9] [18] [19], dado al gran éxito del trabajo de Alex Krizhevsky *et. al*. (2012) [20] en el área de clasificación de imágenes. Las arquitecturas diseñadas para clasificar imágenes presentan una gran invariancia ante traslaciones y cambios de escalas en la entrada, lo que es algo muy deseable en problemas de alto nivel. Sin embargo, es una desventaja para problemas como el de segmentación semántica, pues se pierde la información espacial. Esto se debe a que las arquitecturas para clasificar imágenes van reduciendo la dimensionalidad de la imagen de entrada, a medida que pasa por cada capa de subsampling, eliminando la información espacial, y haciéndola imposible recuperar solo a partir del vector de salida.

Uno de los primeros enfoques corresponde a aplicar una red *deep* al problema de segmentación semántica, es el de Couprie *et al.* (2013) [18], donde utilizan una red CNN con la misma arquitectura a la de las redes para clasificar imágenes y un enfoque similar al de los trabajos basados en características *hand-crafted*. Para solucionar el problema de la pérdida de información espacial, se entrena la red, de forma supervisada, en regiones cuadradas de la imagen, obteniendo un vector de características para el pixel central de esa región. De esta forma se logra obtener a la salida características densas de la imagen de entrada. Lo anterior se realiza para varias copias de la red a distintas escalas, compartiendo los pesos entre cada copia. En este trabajo se generan regiones mediante superpixeles gPb[21] y luego se calculan las características para cada una de estas regiones, a partir de la características densas ya generadas. Finalmente se refina este resultado mediante CRF. Trabajos similares, como el de Gupta *et al.*[22], siguen el mismo enfoque, obtenido buenos resultados incluso sin utilizar CRF.

Trabajos posteriores se enfocan en crear arquitecturas de redes *deep* que fueran capaces de entregar de forma inmediata el etiquetado semántico denso, mediante entrenamiento end-to-end [10] [16] [7] [17] [9]. La siguiente sección revisa las principales arquitecturas de redes *deep* para segmentación semántica utilizando un enfoque end-to-end.

2.1.1 Segmentación semántica y deep-learning

Dada la gran eficacia de los métodos basados en deep-learning para clasificar imágenes, surgió rápidamente el interés de utilizar este tipo de redes para problemas de procesamiento de imágenes más complejos, como lo son la detección y clasificación de objetos o la segmentación semántica. Con respecto al problema de segmentación semántica, el primer problema que se presenta es que las redes CNN para clasificar imágenes van reduciendo su dimensionalidad, con la consiguiente pérdida de la información espacial, a medida información va avanzando a través de las capas. Específicamente las capas que tienen la función de subsampling o pooling son las que provocan este efecto. Si bien este efecto es deseado al momento de clasificar imágenes, pues reduce la dimensión espacio de características y la complejidad de la red, es contraproducente para el problema de segmentación semántica. Esto se debe a la perdida de resolución de la imagen de entrada que va produciendo en cada capa de pooling, lo que no permite tener como salida una imagen con la misma resolución de la entrada. Debido a esto es que existen múltiples trabajos que proponen distintas arquitecturas [7][10][23][9] que logran solucionar este problema de diferentes formas. Principalmente se distinguen 3 tipos de arquitectura, fully-convolutional

networks(FCN) [7], redes tipo encoder-decoder [10][23], y redes con convolucionales dilatadas (o también llamadas convoluciones "atrous") [9][24].

Las redes tipo FCN network corresponde a los primeros trabajos que permitieron abordar el problema de segmentación semántica "end-to-end" mediante *deep* learning. La idea principal de esta arquitectura es reemplazarlas ultimas capas "fully connected" de las redes para clasificar imágenes por capas convolucionales, de forma de obtener una matriz 2D en la salida, en vez de un vector unidimensional con las clases.

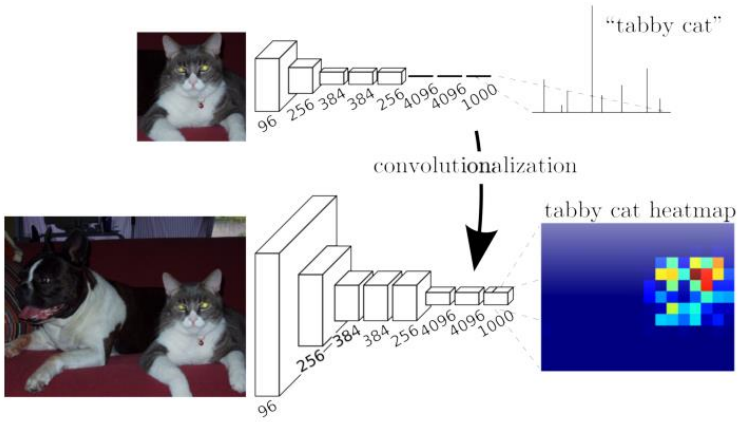


Figura 2-2: Transformación de capas "fully-connected" a convolucionales [7]

Al reemplazar las últimas capas "fully-connected" por convolucionales, el problema de la baja resolución de la imagen de salida (debido a las capas de "pooling"), sigue presente. El trabajo [7] logra mitigar esto en alguna medida al agregar al final una capa extra de "upsampling" mediante deconvoluciones, pero la resolución de salida sigue siendo baja.

Las arquitecturas tipo encoder-decoder, como Segnet [10], logran solucionar el problema de la baja resolución de salida, al guardar los índices de los máximos encontrados en las capas de maxpoling. Estos índices luego son utilizados para realizar "upsampling". De esta forma se logra almacenar la información espacial que se pierde en la redes CNN para clasificar imágenes.

El tercer tipo de arquitectura corresponde a la que propuso el trabajo de Deeplab [9]. Para abordar el problema de la baja resolución de la imagen de salida, provocado por las capas de max-pooling, introducen la convolución dilatada [24] (también llamada "atrous") .Deeplab también utiliza como base la red VGG, solamente que las dos últimas capas de pooling son remplazadas por la convolución dilatada. La convolución dilatada corresponde a una operación de convolución tradicional, solo que el kernel tiene elementos nulos al interior de la máscara. La Figura 2-3 muestra 3kernel de convolución dilatada. El primero con un "rate" igual a 1, donde el kernel resultante es un kernel de convolución de 3x3 tradicional. Para un "rate" igual a 2, existe una separación de un pixel entre los elementos de la máscara del kernel. La cantidad de pesos a calcular sigue siendo la misma, 9, pero el área efectiva que se está abarcado es de 6x6 pixeles. Para un "rate" igual a 3, existe una separación de 2 pixeles. Con los mismos 9 pesos se está abarcando un área de 7x7 pixeles.

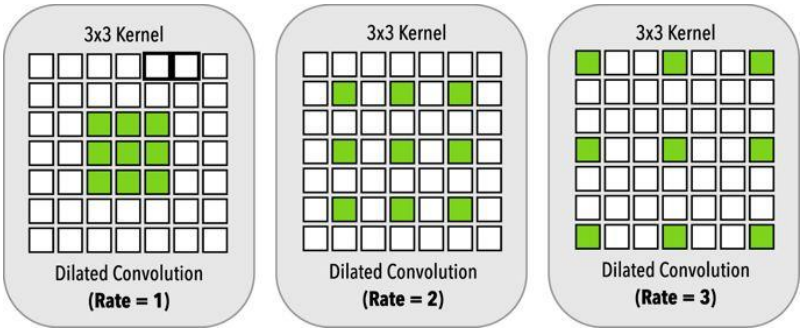


Figura 2-3: kernels de convolución dilatada.

La idea de la convolución dilatada es disminuir la cantidad de parámetros que la red debe aprender, que es uno de los efectos que se busca en las capas de max-pooling, pero sin la consiguiente pérdida de resolución.

En particular se revisa con mayor detalla la red Segnet, que es la que se utiliza en este trabajo de tesis para extraer las características pre-entrenadas para el problema de segmentación semántico. El criterio para escoger esta red es que la red que presenta mejor rendimiento entro los modelos pre-entrenados en SUN-RGBD disponibles al momento de realizar este trabajo de tesis (ver Tabla 2-5), solamente superado por bayesian-segnet. SUN-RGBD es la base de datos de segmentación semántica en interiores más grandes que existe, por lo que es la mejor opción para poder extraer características pre-entrenadas y transferirlas a otra base de datos más pequeña.

Segnet

Una red *deep* que ha cobrado gran relevancia en el problema de segmentación semántica es Segnet. Esta red corresponde a uno de los primeros trabajos en el área de segmentación semántica en proponer una red *deep* completamente convolucional y entrenable end-to-end. La arquitectura de esta red se inspira en trabajos relacionados con autoencoders entrenable de forma no supervisado [25] y distintos métodos para poder visualizar capas intermedias de redes CNN [26].

Segnet está compuesta por una red que cumple la función de encoder, seguida por una red que hace de decoder, y al final posee una capa para clasificar a nivel de pixeles. La red encoder corresponde a VGG16, descartando sus capas “fully-conected” y dejando solo sus 13 capas convolucionales. Las capas de max-pooling, de 2x2, adicionalmente almacenan los índices donde ocurre están activaciones. Esta información espacial, que se pierde en las redes para clasificar imágenes, es almacenada para poder realizar el upscaling de la salida.

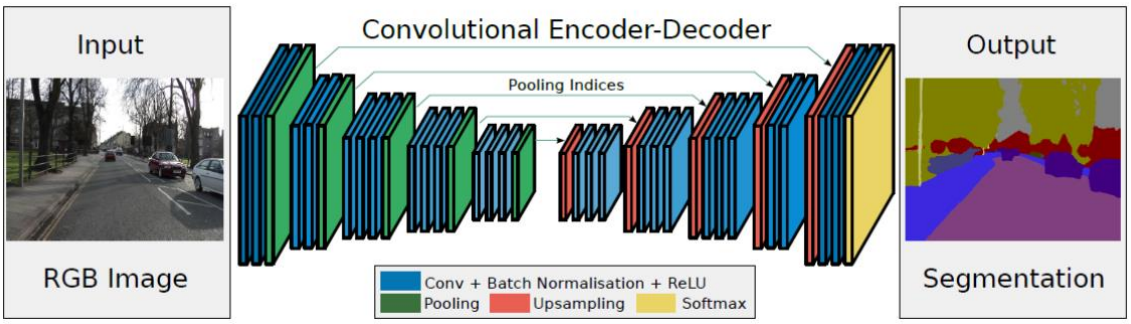


Figura 2-4: Arquitectura de Segnet [10]

2.2 Clasificación de lugares

El problema de clasificación de lugares corresponde a asignar una etiqueta de lugar (por ejemplo dormitorio, montaña o bosque) a una imagen [27] [28] [29]. El enfoque clásico a este problema está basado en sistemas para clasificar imágenes utilizando características *hand-crafted* como SIFT-BOW, Gist, HOG o LBP [30] [31] [32] [27] [28] [33] [29].

Con respecto a los sistemas para clasificar imágenes, estos sufrieron un gran cambio de paradigma a partir del año 2012, gracias al trabajo de Krizhevsky *et al.* [20]. La red *deep* propuesta en este trabajo, conocida como *Alexnet* logró superar por amplio a todos los métodos conocidos en el problema de categorizar imágenes. La característica principal de este enfoque se basa en la idea de que red neuronal multicapa aprenda de forma automática tanto a generar las características como a clasificarlas. Esto se diferencia del enfoque clásico en que las características son diseñadas a mano (*hand-crafted*) de forma muy laboriosa, para luego pasar a la etapa de clasificación. Este paradigma se conoce como *deep learning* y el precursor en esta área aplicada a visión corresponde al trabajo de LeCun *et al*. [34]. En este trabajo se utiliza una red convolucional multicapa (CNN, por sus siglas en inglés) entrenada mediante back-propagation, para poder reconocer dígitos en imágenes. La arquitectura propuesta por Krizhevsky *et al.* [20] es similar a la del trabajo de LeCun *et al*. [34], salvo que es más profunda, introduce una etapa de regularización mediante "*dropout*", para evitar el sobre aprendizaje [20] y además utiliza la función de activación no lineal "*Relu*" en vez de las tradicionales. La función de activación "*Relu*" es mucho más eficiente al momento de entrenar redes deep, permitiendo reducir los tiempos de entrenamiento hasta 6 veces [20] comparadas con funciones de activación tradicionales, como sigmoidea o tangente hiperbólica. Al momento de ser entrenadas este tipo de redes requiere de mucho poder de cómputo en modernas GPU y de bases de datos muy grandes y etiquetadas, para poder entrenar miles de parámetros y evitar el sobre-aprendizaje. Como referencia la base de datos donde fue entrenada Alexnet corresponde a Imagenet, que posee del orden de millones de ejemplos. Estos 3 factores combinados (mejoras en la arquitectura como Relu, dropout y mayor profundidad, mayor poder de cómputo disponible y la aparición de grandes bases de datos) permitieron el éxito de Alexnet en la tarea de clasificar imágenes en Imagenet. Actualmente existen varias arquitecturas *deep* CNN que han mejorado los resultados de Alexnet, como GoogleNet [35] o VGG [36], aunque todas estas se inspiran en la arquitectura de red Alexnet y sus mejoras introducidas.

Un trabajo de gran interés es el clasificación de lugares mediante Deep-learning es el de Zhou *et al.*[37], donde crean una base de datos de lugares con millones de imágenes etiquetadas, de forma de poder entrenar redes CNN. Esta base de datos la llaman Place205, pues posee 205 clases de lugares, que van desde playas, montañas, hasta oficina o dormitorio. En este trabajo se prueban tres arquitecturas de redes CNN, correspondientes a Alexnet, Googlenet y VGG16. Los mejores resultados los obtienen que la red VGG16.

Sin embargo, cuando se dispone de una cantidad muy limitada de ejemplos en una base de datos, el entrenamiento de estas redes *deep* no entrega buenos resultados [20]. En varias tareas de reconocimiento de imágenes solo se dispone de bases de pequeñas en comparación a Imagenet (con 2 a 3 órdenes de magnitud inferiores respecto a Imagenet), donde no es posible entrenar una red como Alexnet. Es para este caso donde varios trabajos [6][38][4][5][36][26] han explorado la trasferencia de conocimiento de redes pre-entrenadas a problemas con bases de datos muy pequeñas. Estos trabajos muestran que extrayendo características desde las últimas capas de redes CNN se obtiene un descriptor muy potente para discriminar imágenes. La base de datos NYUv1 corresponde a este caso, donde es muy pequeña (solo posee 2284 imágenes), por lo que se revisaran con mayor detalles los trabajos que buscan traspasar el conocimiento de una red pre-entrenada otro problema.

En la siguiente sección se muestran los resultados más relevantes de segmentación semántica en bases de datos de interiores domésticos.

2.2.1 Rankings en bases de datos

Este trabajo de tesis se enfoca en el problema de segmentación semántica en interiores domésticos, por lo que se hace una recopilación de resultados en esa área. Hasta la fecha de realización de este trabajo solo existen tres bases de interiores domésticos, NYU Depth v1, NYU Depth v2, y SUN-RGBD. Las Tabla 2-1, Tabla 2-2, Tabla 2-3, Tabla 2-4 y Tabla 2-5 recopilan los resultados de distintos trabajos en estas tres bases de datos.

Para cada trabajo se muestra su nombre junto con su año de publicación y su rendimiento. La columna "*método*" muestra los métodos más relevantes usados en el trabajo, la columna "*Depth"* indica si se utiliza información de la imagen de profundidad, la columna *"CRF/MRF"* indica si se utiliza alguna de estas dos técnicas o variantes, la columna "ME/IT" indica si se utiliza análisis multi-escala y por último la columna *"Deep Lear."* indica si es un método basado en *deep* learning o no.

Tabla 2-1. Ranking base de datos NYUv1 – 13 clases

Trabajo	Media Clases	Método	Depth	CRF/ MRF	ME/ IT	Deep Lear.
Wang *et al.* (2012) [39]	54.8%	MUFL-SVM	X	-	-	-
Silberman *et al.* (2011) [40]	56.6%	SIFT-SVM	X	X	-	-
Hermnans *et al.* (2014) [41]	59.5%	RPF-RF	X	X	-	-
Wolf *et al.* (2015) [42]	68.2%	Feat. Point clo.-RF	X	X	X	-
Ren *et al.* (2012) [43]	76.1%	KDE-SVM	X	X	X	-

Tabla 2-2. Ranking base de datos NYUv2 – 13 clases

Trabajo	Media Clases	Método	Depth	CRF/ MRF	ME/ IT	Deep Lear.
Couprie *et al.* (2014) [44]	36.2%	Convnet	X	-	X	X
Wang *et al.* (2012) [39]	42.5%	MUFL-SVM	X	-	-	-
Khan *et al.* (2014) [45]	45.1%	SDP HOP	-	X	-	-
Hermnans *et al.* (2014)[41]	48.0%	RPF-RF	X	X	-	-
Handa *et al.(*2015) [46]	52.5%	SceneNet-DHA	X	-	-	X
Wolf *et al.* (2015) [42]	56.9%	Feat. Point clo.-RF	X	X	-	-
Eigen *et al.* (2015) [17]	66.9%	MultiEsc-CNA	X	-	X	X

Tabla 2-3. Ranking base de datos NYUv2 – 4 clases

Trabajo	Media Clases	Método	Depth	CRF/ MRF	ME/ IT	Deep Lear.
Silberman *et al.* (2012[47]	59.6%	SIFT-3D priors	X	X	-	-
Couprie *et al.(*2013) [44]	63.5%	Convnet	X	-	X	X
Wang *et al.* (2012) [39]	65.3%	MUFL-SVM	X	-	-	-
Khan *et al.* (2014) [45]	65.6%	SDP HOP	-	X	-	-
Hermnans *et al.* (2014) [41]	69,0%	RPF-RF	X	X	-	-
Muller *et al.* (2014) [48]	71.9%	SP-RF	X	X	-	-
-Wolf *et al.* (2015) [42]	72.6%	Feat. Point clo.-RF	X	X	-	-
Eigen *et al.*(2015) [17]	82.0%	MultiEsc-CNA	X	-	X	X

Tabla 2-4. Ranking base de datos NYUv2 – 40 clases

Trabajo	Media Clases	Método	Depth	CRF/ MRF	ME/ IT	Deep Lear.
Gupta et al.(2012) [49]	28.4%	feat hc+SVM	X	-	X	-
Gupta *et al.*(2014) [22]	35.1%	Feat CNN+SVM	X	-	-	X
Badrinarayanan [10]	36.0%	Segnet	-	-	-	X
Shelhamer *et al.*(2015) [7]	44.8%	FCN	X	-	X	X
Kendal *et al.*(2015) [16]	45.8%	Baye. Segnet	-	-	X	X

Tabla 2-5. Ranking base de datos SUN RGBD

Trabajo	Media Clases	Método	Depth	CRF/ MRF	ME/ IT	Tipo-características
Hong et al. (2015) [23] [16]	32.3%	DeconvNet	-	-	-	X
Chaurasia et al. (2016) [50]	32.6%	Enet	-	-	-	X
Ren et al. (2012) [43]	36.3%	KDE	X	X	X	-
Shelhamer et al. (2016) [7][16]	38.4%	FCN	-	-	X	X
Chen et al. (2016) [9] [16]	42.2%	DeepLab	-	X	-	X
Shelhamer et al. (2016) [20] [16]	42.5%	FCN-RGBD	X	-	X	X
Badrinarayanan et al. (2015)[10]	44.7%	Segnet	-	-	-	X
Kendal et al. (2016) [16]	45.8%	Baye. Segnet	-	-	X	X

Entre los trabajos basados en redes tipo encoder-decoder sobresale en particular Segnet [10] y Bayesian Segnet [16], debido a su rendimiento y al ser consideras unas arquitecturas tipo por otros trabajos para el problema de segmentación semántica.

El trabajo de Ren *et al.* [43] es de particular interés, pues es el mejor resultado en la base datos NYUv1 hasta la fecha de publicación de este documento. En el trabajo de Ren *et al.* [43] realizan pruebas en dos bases de datos, NYU v1 y Stanford Background Dataset. La primera base de datos corresponde a lugares en interiores domésticos, mientras que la segunda a imágenes de ciudades y paisajes en exteriores.

De forma muy general, el trabajo Ren *et al.* [43] sigue con el esquema clásico propuesto por los trabajos anteriores, aunque utiliza métodos distintos en cada etapa. Ver Figura 2-5. Para generar los segmentos de la imagen a clasificar, utiliza un detector de fronteras y un sistema de segmentación jerárquico, gPb-ucm [21]. Las características que extrae son KDE (Kernel Descriptors), que corresponde a distintas mascaras convolucionales y transformaciones no lineales, para extraer información de gradiente, color y normales de las superficies. Estas características se extraen de forma densa en las imágenes RGBD.

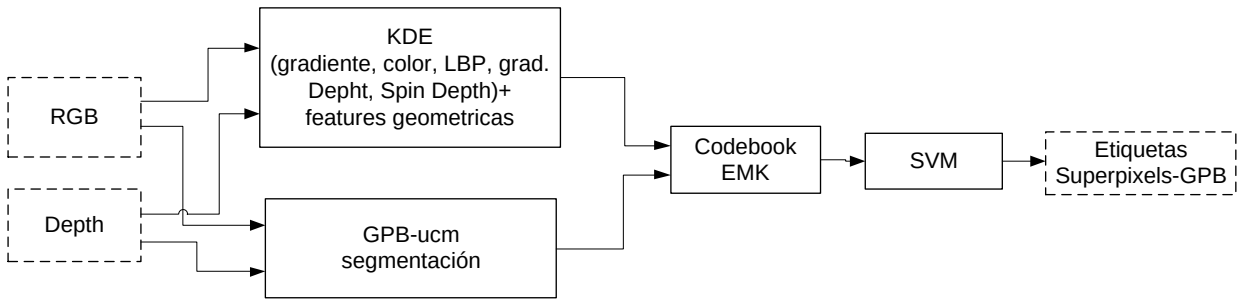


Figura 2-5: Arquitectura Segmentación semántica mediante KDE

A continuación se describe con mayor detalle el trabajo de Ren *et al.* [43], pues se utiliza como una parte del sistema propuesto en este trabajo de tesis.

Módulo KDE

Kernels descriptors [51][52] (KDE) corresponde a un método para extraer características de bajo nivel en una ventana (zonas de *n* x *n* pixeles) dentro de una imagen, bajo el enfoque de kernels. Esto con el fin de comparar la similitud entre dos ventanas distintas. Descriptores como SIFT u HOG discretizan los atributos del pixel en intervalos discretos para luego generar histogramas, permitiendo compararlos en bases a estos histogramas. Una desventaja de esto es que la generación de intervalos introduce errores de discretización. Los kernels descriptors reducen este error de discretización al trabajar bajo un enfoque de kernels.

La expresión que describe a los KDE se deriva directamente de expresar descriptores como SIFT u HOG bajo el enfoque de kernels. Estos dos descriptores corresponden a dos casos particulares de KDE, al utilizar un kernel que discretiza las orientaciones. A continuación se muestra el desarrollo que permite obtener la expresión genérica de los KDE.

Sea $\theta(z)$ la orientación del gradiente y $m(z)$ la magnitud del gradiente, para un pixel z dentro de una imagen. En un descriptor como SIFT u HOG, el gradiente de cada pixel es discretizado en d direcciones, que pueden expresarse dentro de un vector $\delta(z)$ de la siguiente forma:

$$\delta(z) = [\delta_1(z), \dots, \delta_d(z)] \tag{2-1}$$

Los elementos del vector $\delta(z)$, se genera con la siguiente expresión:

$$\delta_i(z) = \begin{cases} 1 & si \ \left\lfloor \frac{d\theta}{2\pi} \right\rfloor = i - 1 \\ 0 & otro \end{cases} \tag{2-2}$$

Gracias a esta función, la orientación del gradiente queda representada dentro de uno de los intervalos discretos. El vector de características de cada pixel z, $F(z)$ se obtiene al ponderar el vector discretizado del gradiente por la magnitud del gradiente. Esto es:

$$F(z) = \delta(z)\, m(z) \tag{2-3}$$

Luego, para obtener el histograma de gradientes orientados en una ventana P de la imagen, se realiza la sumatoria de los vectores de características de cada pixel dentro de P:

$$F_h(P) = \sum_{z \in P} \delta(z)\, \widetilde{m}(z) \tag{2-4}$$

Donde $\widetilde{m}(z)$ es el gradiente normalizado en relación al tamaño de la ventana P:

$$\widetilde{m}(z) = \frac{m(z)}{\sqrt{\sum_{z \in P} m(z)^2}} \tag{2-5}$$

El tamaño de esta ventana P es típicamente de 4×4 y para HOG de 8×8. Para medir la similitud entre dos ventanas, se utilizada una medida de distancia relacionada con la norma L2. Esto se puede expresar mediante un kernel lineal. Sean P y Q dos ventanas, típicamente de distintas imágenes, la expresión para medir similitud entre ellas queda expresada por:

$$K_h(P,Q) = F_h(P)^T F_h(Q) = \sum_{z' \in Q} \sum_{z \in P} \widetilde{m}(z') \widetilde{m}(z) \delta(z')^T \, \delta(z) \tag{2-6}$$

Reemplazando $\widetilde{m}(z')\widetilde{m}(z)$ por $k_{\widetilde{m}}(z,z')$ y $\delta(z')^T \, \delta(z)$ por $k_\delta(z,z')$:

$$K_h(P,Q) = \sum_{z' \in Q} \sum_{z \in P} k_{\widetilde{m}}(z,z') k_\delta(z,z') \tag{2-7}$$

En la ecuación (2-7), tanto $k_{\widetilde{m}}(z,z')$ como $k_\delta(z,z')$ corresponden a un producto interno punto, resultando en kernels definidos positivos. Podemos decir que $K_h(P,Q)$ corresponde a un match kernel sobre el set, compuesto por las ventanas P y Q. La ecuación (2-7), muestra el enfoque basado en kernels para realizar un calce entre ventanas usando las características HOG.

Notar que el kernel $k_\delta(z,z')$ mide la similitud entre la orientación del gradiente entre dos pixeles, siendo 1 si ambas orientaciones calzan en el mismo intervalo, y 0, en cualquier otro caso (*hard binning).* Esto introduce errores de discretización que pueden provocar resultados sub-óptimos. Una forma de remediar esto en cierta medida es utilizar un kernel que exprese el grado de pertenencia en cada intervalo de las orientaciones (*soft binning*), mediante una función delta como la siguiente:

$$\delta_i(z) \; = \; max(cos(\theta(z) - a_i)^9, 0) \tag{2-8}$$

Aunque esto sigue sufriendo de los inconvenientes de la discretización. Sin embargo, el enfoque basado KDE permite buscar y aplicar todo tipo de kernels. De aquí nacen ideas como utilizar un kernel de orientación $k_O(\tilde{\theta}(z), \tilde{\theta}(z'))$ en vez de $k_\delta(z,z')$, que utilicen un kernel gausssiano directamente sobre la diferencia de las orientaciones, en vez de comparar la discretización de estas. Este nuevo kernel queda definido de la siguiente manera:

$$k_O(\tilde{\theta}(z), \tilde{\theta}(z')) \; = \exp(-\gamma_O \left\| \tilde{\theta}(z) - \tilde{\theta}(z') \right\|^2) \tag{2-9}$$

Donde $\tilde{\theta}(z)$ es un vector con la orientación normalizada:

$$\tilde{\theta}(z) = [\sin(\theta(z)) \cos(\theta(z))] \tag{2-10}$$

El trabajo [52] propone como primer kernel match el de gradiente, K_{grad}, donde además de añadir el kernel de orientación, k_O se añade un kernel de posición $k_p(z,z')$, para tener una medida en relación a la distancia espacial de los pixeles a comparar. El kernel match de gradiente, K_{grad}, queda definido de la siguiente forma:

$$K_{grad}(P,Q) = \sum_{z' \in Q} \sum_{z \in P} k_{\widetilde{m}}(z,z') k_O \left(\tilde{\theta}(z), \tilde{\theta}(z') \right) k_p(z,z') \tag{2-11}$$

Donde $k_O(z, z')$ es un kernel gaussiano sobre la distancia euclidiana de los dos pixeles a comparar (normalizada entre 0 y 1), tal como se muestra a continuación:

$$k_P(z, z') = \exp(-\gamma \|z - z'\|^2) \tag{2-12}$$

A continuación, se define una forma para calcular un descriptor denso en una ventana de la imagen a partir del enfoque de kernels, pues los resultados hasta ahora son para comparar dos ventanas distintas. Continuado con el ejemplo anterior, se reescribe las expresiones del kernel de orientación (2-9) y el kernel de posición (2-12), bajo la forma de producto punto:

$$k_O(\tilde{\theta}(z), \tilde{\theta}(z')) = \emptyset_O\left(\tilde{\theta}(z)\right)^T \emptyset_O(\tilde{\theta}(z')) \tag{2-13}$$

$$k_P(z, z') = \emptyset_P(z)^T \emptyset_P(z') \tag{2-14}$$

A partir de las expresiones (2-13) y (2-14) se puede derivar la expresión para calcular las características sobre una sola ventana P, tal como se muestra en la ecuación (2-15):

$$F_{grad}(P) = \sum_{z \in P} k_{\widetilde{m}}(z) \emptyset_O\left(\tilde{\theta}(z)\right) \otimes \emptyset_P(z) \tag{2-15}$$

En donde $\otimes$ es el producto de Kronecker. Para esta nueva característica generada, se cumple que:

$$F_{grad}(P)^T F_{grad}(Q) = K_{grad}(P, Q) \tag{2-16}$$

El único inconveniente de la características generada (2-15) es que al tener un kernel gaussiano, $F_{grad}(P)$ se convierte en un vector de dimensión infinita. Una forma de reducir la dimensión, es muestrear de forma uniforme vectores bases de la región de soporte, los suficientes hasta obtener una buena aproximación del kernel. De esta forma se pueden obtener características compactas de baja dimensión. Para obtener estos vectores bases, se muestrea en forma de grilla sobre la región de soporte, de forma densa y uniforme. Luego la aproximación del kernel se obtiene al proyectar este en un espacio de menor dimensión, compuesto por los vectores bases muestreados.

Por ejemplo, considerando el kernel de orientación, $k_O(\tilde{\theta}(z), \tilde{\theta}(z'))$, y dado un set de vectores bases $\{\emptyset_O(x_i)\}_{i=1}^{d_O}$, donde x_i son vectores de gradiente muestreados de forma uniforme sobre las orientaciones posibles, se puede aproximar el vector de dimensión infinita $\emptyset_O\left(\tilde{\theta}(z)\right)$ a un espacio de dimensión $d_O$al proyector en los vectores bases $\{\emptyset_O(x_i)\}_{i=1}^{d_O}$.

Siguiendo la formulación del trabajo [51], el procedimiento anterior equivale a utilizar un kernel de dimensión finita, aproximado, de la siguiente forma.

$$\tilde{k}_O\left(\tilde{\theta}(z), \tilde{\theta}(z')\right) = k_O\left(\tilde{\theta}(z), \tilde{\theta}(z'), X\right)^T \left[\boldsymbol{K_O^{-1}}\right]_{ij} k_O\left(\tilde{\theta}(z'), X\right)^T = \left[\boldsymbol{G} k_p(z, X)\right]^T \left[\boldsymbol{G} k_p(z', X)\right] \tag{2-17}$$

Así como el kernel match de gradiente (2-11) también se puede definir kernels que extraigan información de color, o de forma. En particular el trabajo [43], que es el que obtiene los mejores resultados en segmentación semántica en la base de datos NYU Depth v1 a la fecha, utiliza los kernels de gradiente y color para la imagen RGB y profundidad y los vectores normales para la imagen de profundidad.

Módulo gPb-ucm

En esta sección se describe el método para generar los segmentos a clasificar. Para referirse a estos segmentos se utiliza el término superpixeles, que si bien generalmente solo se usa para agrupaciones de pixeles por criterios de similitud de vecindad (como color o gradiente), es el más adecuado para describir estos segmentos de imagen sin dejar espacio a ambigüedades. Esta etapa de generación de superpixeles mediante gPb-ucm corresponde a la utilizada en el trabajo Ren et al. [43].

Detección de contornos

El bloque básico para detector de contornos Pb es el cálculo del gradiente orientado $G(x, y, \theta)$ para una imagen I. Este cálculo se realiza colocando un disco en la ubicación (x, y) dividido en dos mitades con un diámetro de ángulo θ. Para cada mitad, se calcula un histograma con la intensidad de la imagen. La magnitud del gradiente G en la ubicación (x, y) se define como la distancia x^2 entre los histogramas de cada mitad del disco, g y h.

$$x^2(g,h) = \frac{1}{2}\sum_i \frac{(g(i) - h(i))^2}{g(i) + h(i)} \tag{2-18}$$

La Figura 2-6 muestra un ejemplo de este cálculo y su resultado.

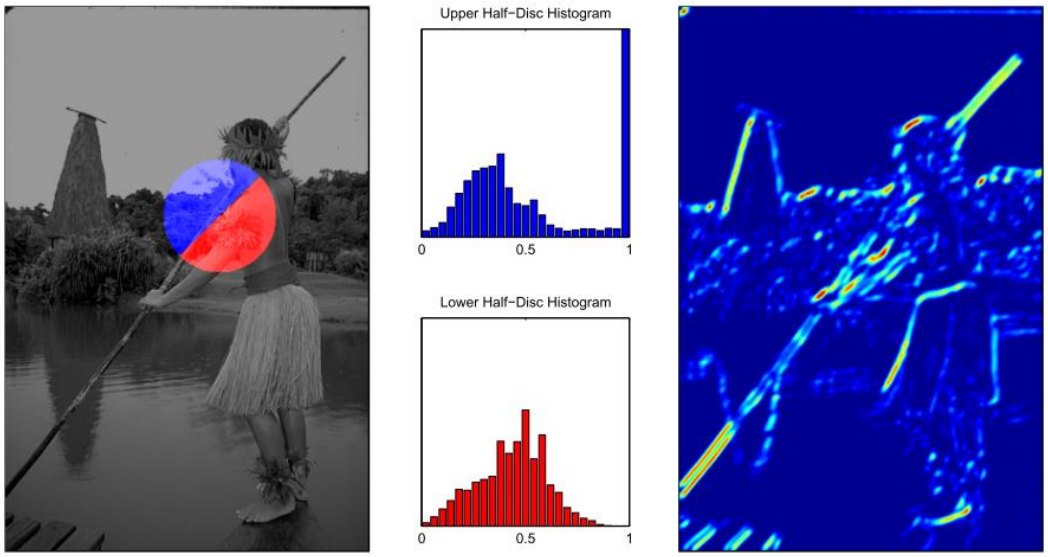


Figura 2-6: Izquierda, calculo histogramas sobre un disco orientado. Figura derecha, imagen del gradiente orientado

El detector de bordes *Pb* combina la información del gradiente orientado de cuatro canales distintos de la imagen a analizar. Tres canales del espacio de color CIE Lab (brillo, color a y color b), y un cuarto canal de textura (obtenido mediante un banco de filtros de textons). Lo anterior se realiza para tres escalas distintas $[\frac{\sigma}{2}, \sigma, 2\sigma]$, para cada uno de los 4 canales.

Luego se combinan de forma lineal estos cuatro canales, para las tres escalas definidas. La expresión general para el detector de bordes es:

$$mPb(x, y, \theta) = \sum_{s} \sum_{i} \alpha_{i,s} G_{i,\sigma(i,s)}(x, y, \theta) \tag{2-19}$$

Donde s es el índice de la escala, i el índice del canal con las características (brillo, color a, color b y textura) y $G_{i,\sigma(i,s)}$ mide las diferencias entre los histogramas de cada semi círculo de radio $\sigma(i, s)$, centrado en el punto (x, y) y divididos con un diámetro de ángulo θ. Los pesos $\alpha_{i,s}$ son ponderaciones para cada una de las señales de gradiente y aprendidos mediante aprendizaje supervisado. Este entrenamiento se realiza utilizando la base de datos *Berkeley Segmentation Dataset* (BSDS), optimizando el F-score sobre el conjunto de entrenamiento.

MPb extrae información local sobre la probabilidad de encontrar un contorno. Para agregar información global, se utiliza clustering espectral. Para la etapa de clustering espectral, se utiliza la matriz de afinidad W, que contiene información del valor máximo de mPb en una línea que conecta dos pixeles, j e i. Se conectan todos los pixeles j e i en un radio menor a $r = 5$ pixeles, y se define la expresión para W_{ij} como:

$$W_{ij} = \exp(-max_{p \in \overline{ij}}\{mPb(p)\}/\rho) \tag{2-20}$$

Donde $\overline{ij}$ es el segmento de línea que conecta el punto *i* con el punto *j*, y ρ es una constante igual a 0.1. Con el objetivo de agregar información global, se define $D_{ii} = \sum_j W_{ij}$ y se resuelve el sistema $(D - W)\boldsymbol{v} = \boldsymbol{\lambda} D\boldsymbol{v}$ para encontrar los vectores propios $\{\boldsymbol{v}_0, \boldsymbol{v}_1 \ldots \boldsymbol{v}_n\}$.

Los vectores propios contienen información sobre los bordes de la imagen. Al tratar cada vector propio $\boldsymbol{v}_k$como una imagen y convolucionarlos con filtros derivativos gaussianos con múltiples orientaciones θ, se obtiene las señales orientadas $\{\nabla_\theta \boldsymbol{v}_k(\boldsymbol{x}, \boldsymbol{y})\}$. La información de los vectores propios es combinada para crear el componente "espectral" del detector de bordes, de la siguiente forma:

$$sPb(x, y, \theta) = \sum_{k=1}^{n} \frac{1}{\sqrt{\lambda_k}} \cdot \nabla_\theta \boldsymbol{v}_k(x, y) \tag{2-21}$$

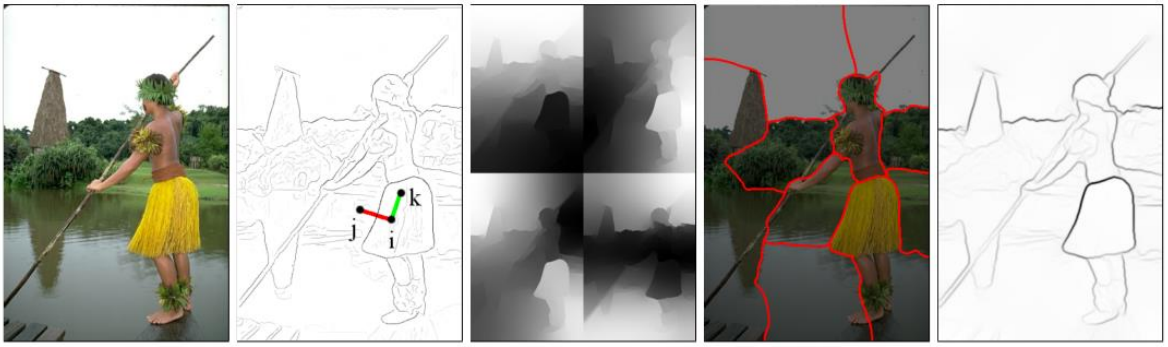


Figura 2-7: Distintas etapas del cálculo de contornos mediante clustering espectral.

Finalmente, una combinación lineal simpe entre mPb y sPb genera el detector de contornos gPb (Global Probability Boundary) :

$$gPb(x,y,\theta) = \sum_s \sum_i \beta_{i,s} G_{i,\sigma(i,s)}(x,y,\theta) + \gamma \cdot sPb(x,y,\theta) \qquad (2\text{-}22)$$

Las constantes $\beta_{i,s}$ e γ son calculadas mediante aprendizaje supervisado utilizando la base de datos *Berkeley Segmentation Dataset* (BSDS).

2.2.2 Transferencia de conocimiento desde redes CNN pre-entrenadas

En esta sección se revisan dos trabajos que se consideraron los más relevantes en el área. La idea general de estos trabajos es transferir el conocimiento aprendido por red CNN ya entrenada extrayendo características de algunas de sus capas para utilizarlas como descriptor genérico en otras bases de datos más pequeñas, como *Caltech-101*, *Caltech-UCSD Birds* y *SUN-397 scene*. Para extraer las características utilizan la red Alexnet con pesos aprendidos en Imagenet y propagan hacia adelante las imágenes de la nueva base de datos. La activación de las neuronas en las distintas capas corresponde a los valores de las características que extraen. Las capas desde donde prueban[6] extraer características corresponden a todas las *fully-conected* que se encuentran antes de la última capa de salida (clasificación *softmax*). Estas corresponden a las capas *fully-conected* 6 7 y 8 respectivamente, las cuales llaman en el trabajo bajo el nombre de descriptor *DeCAF*[6]. Las capas 6 y 7 corresponden a un vector de dimensión 4096 mientras que la capa 8 a un vector de dimensión 1000. Para entrenar en nuevas bases de datos prueban[6] con estos descriptores más clasificadores SVM. Los resultados más relevantes de sus pruebas se pueden ver en la Tabla 2-6 y Tabla 2-7.

Tabla 2-6. Resultados media por clase en Caltech-101 DeCAF contra el mejor resultado [6]

	$\boldsymbol{DeCAF_5}$	$\boldsymbol{DeCAF_6}$	$\boldsymbol{DeCAF_7}$
SVM+*DeCAFn*	77.12%	**86.91%**	83.24%
Yang et al.[53] (2009) (*SIFT+CSIFT+PHOG+SS*)		84.3%	

Tabla 2-7. Resultados media por clase en Caltech-101 DeCAF contra el mejor resultado [6]

Método	**Base de datos**	**Accu. Media**
$DeCAF_6$	Caltech-UCSD bird dataset	**64.96%**
POOF (Berg & Belhumeur, 2013)	Caltech-UCSD bird dataset	56.78%
$DeCAF_6$	SUN-397 scene	**40.94%**
Xiao *et al*. (2010)	SUN-397 scene	38.00%

A partir de las tablas anteriores se puede ver que utilizando que utilizando estas características lograron superar en todas las bases de datos los resultados *state-of-art*, que eran basados en descriptores *hand-crafted*. Algunos de estos trabajos usaban varios tipos de descriptores juntos, como [53], que usan Dense SIFT, Dense Color SIFT, P-HOG y características SS (*self-similarity*). En particular, en la Tabla 2-6 se puede ver que las características extraídas de la capa 7 ($DeCAF_6$) son las que mejor se desempeñan. Este trabajo[6] concluye que las capas fully-conected de la red Alexnet pre-entrenada en Imagenet pueden ser utilizadas para extraer características con un alto poder de separabilidad. En particular se observa que en las 3 bases de datos probadas las características extraídas de una red Alexnet sobrepasan a sofisticados métodos basados en características *hand-crafted*, solamente utilizando estas características más clasificadores SVM. A su vez muestran que estas características son útiles en problemas y bases de datos distintas a donde fue entrenada la red CNN (Imagenet), es decir posen alta capacidad de generalización.

El segundo trabajo de particular interés es el de Razavian *et. al* [4], donde también se explora el uso de redes pre-entrenadas. Este trabajo es muy detallado y prueba en varias bases de datos famosas, donde hasta el momento el mejor resultado lo obtienen métodos *hand-crafted*. En particular extraen características de una red a la que llaman *Overfeat*, que es muy similar a Alexnet solo que varían algunas dimensiones de ciertas capas y otros detalles menores. Esta red es entrenada en Imagenet. En este trabajo definen tres opciones posibles.

- Dejar los pesos de la red pre-entrenada fijos hasta la capa-*n* (generalmente una de las ultimas capas) y entrenar o hacer *fine-tunning* solo a las últimas capas de la red, desde la capa *n+1*.
- Dejar los pesos de la red pre-entrenada fijos hasta la capa-*n* (generalmente una de las ultimas capas) y entrenar con algún clasificador lineal tipo SVM.
- Inicializar la red con los pesos pre-entrenados en Imagenet y entrenar o hacer *fine-tunning* a toda la red completa.

En particular Razavian *et. al* [4] encuentran que los dos primeros casos son útiles cuando la base de datos es pequeña, ya que al aplicar el tercer método en una base de datos pequeña se termina sobre-entrenando toda la red completa. Este último entrega un mejor rendimiento cuando se tienen una cantidad suficiente de ejemplos para ser entrenadas. A su vez, entre los dos primeros casos, encuentran que el clasificador SVM entrega mejores resultados en la mayoría de los casos. En particular el trabajo[4] se centra segundo caso, específicamente extrayendo características de la primera capa *fully-conected* (un vector de dimensión 4096) de la red *Overfeat* y clasificando con SVM. A su vez prueban el mismo caso anterior pero entrenando con *data augmentation*, esto corresponde a aumentar la cantidad de ejemplos del conjunto de entrenamiento mediante copias recortadas, rotadas y/o reflejadas del mismo conjunto. En la Tabla 2-8 se muestran los resultados

de pruebas en 10 bases de datos de imágenes, todas ellas bastantes pequeñas comparadas con Imagenet (que posee 3,5 millones imágenes etiquetadas). La tercera columna muestra el mejor resultado obtenido hasta la fecha (todos ellos basados en características *hand-crafted*), la cuarta columna a características pre-entrenadas CNN-*Overfeat*, la quinta al mismo caso anterior pero con *data augmentation* y la última a *fine-tunning* a toda la red (solo en los casos que disponen de ejemplos suficientes).

Tabla 2-8 Resultados trasferencia de conocimiento desde red CNN *Overfeat*

Base de datos	**Tamaño Base de datos**	**Mejor resultado *state-of-the-art***	**Feat. Extraida CNN+SVM**	**Feat. Extraida CNN+SVM+data aug.**	***Fine tunning* toda la red**
Pascal VOC 2007	~10,000	71.1%	73.9%	77.2%	**77.7%**
MIT-67 indoor scenes	15620	64.0%	58.4%	**69.0%**	68.9%
Caltech-UCSD Birds	11,780	56.8%	53.3%	61.8%	**65.0%**
Oxford 102 Flowers	~10,000	80.7%	74.7%	**86.8%**	-
H3D Human Attributes	~10,000	69.9%	70.8%	73.0%	**79.0%**
UIUC 64 object	~5000	89.5%	89.0%	**91.4%**	-
Oxford5k buildings	5,063	74.9%	65.9%	**79.5%**	-
Paris6k buildings	6,412	67.4%	48.5%	**68.0%**	-
Sculptures6k	6,340	**45.4%**	-	41.3%	-
Holidays dataset	1491	81.9%	64.6%	**84.3%**	80.2%
Ukbench	2250	89.3%	76.3%	**91.1%**	-

El primer resultado importante de este trabajo[4] es que en 9 de las 10 bases de datos se logró superar a los mejores resultado *state-of-the-art* existentes hasta el momento utilizando alguno de los 3 métodos propuestos. Esto demuestra que estas características son capaces de competir y superar a los métodos y descriptores *hand-crafted* más sofisticados y altamente especializados existentes hasta al momento, como VLAD, IFV, BOW, SIFT, HOG, LBP y sus múltiples variantes.

En particular se puede ver que las características extraídas de red CNN más un clasificador SVM resultan en un método competitivo respecto al *state-of-the-art*, superándolo en algunos casos. Incluso en los casos en que se logra superar al mejor resultado *state-of-the-art*, este puede ser aún mejorado mediante *data augmentation* o *fine-tunning*. En particular *fine-tunning* solo es aplicable y entrega buenos resultados en bases de datos donde se dispone de aproximadamente 10,000 ejemplos o más, de lo contrario incluso puede empeorar los resultados en comparación al caso más simple (CNN+SVM).

Algunos otros trabajos también exploran el uso de extracción de características CNN en conjunto con SVM [36] [26] [5], aunque no son tan detallados como los dos trabajos ya revisados. Sin embargo, llegan a las mismas conclusiones. En particular algo importante de destacar los métodos mejores métodos *hand-crafted*, como VLAD, BOW, FIV o HE, es que poseen una etapa de entrenamiento sobre la misma base de datos donde se prueban, para crear sus diccionarios o *codebooks.* Esto se diferencia de las características CNN transferidas, que para ser generadas no requieren de ningún tipo de aprendizaje sobre el conjunto de entrenamiento de la base de datos donde van a ser utilizadas.

Si las bases de datos no tuvieran ningún tipo de sesgo y fueran realmente representativas del mundo real, no tendría mayor relevancia al momento de comparar el hecho que un método ocupara algún tipo de aprendizaje (por ejemplo, para generar su codebook) o no. Sin embargo, en la gran mayoría de las bases de datos de visión (salvo algunas como Imagenet y otras, que intentan remediar esto mediante el uso de millones de ejemplos) esto no es así. Existen varios trabajos que se han dedicado a investigar el sesgo en las bases de datos de visión [2] [3][1]. Estos trabajos parten de la premisa que uno esperaría que un método para detectar autos pudiera generalizar de forma correcta en otras bases de datos, o en mundo real, no que fuera un método útil solo en la base de datos donde se entrenó. Esto se cumpliría siempre y cuando la base de datos de entrenamiento no tuviera ningún sesgo y fuera representativa con respecto al mundo real. De lo contrario, el algoritmo de aprendizaje terminaría aprendiendo de forma inherente algunos sesgos presentes en la base de datos y perdiendo su capacidad de generalizar. Para verificar la existencia de sesgos en algunas de las bases de datos de visión más famosas, Torralba *et al.* [1] realizan la prueba de entrenar detectores de autos y personas, basados en métodos *hand-crafted*, en varias de estas bases de datos y comparar los resultados. En todos los casos, el mejor resultado se obtiene cuando se prueba en el respectivo conjunto de test de la base de datos donde se entrenó. Si bien el resultado anterior era esperable, la baja de rendimiento al momento de probar estos detectores en otras bases es dramática en la mayoría de los casos. En algunos casos, como el del detector de autos entrenado en *"Caltech-101*", que al ser probado en el conjunto de entrenamiento de base de datos "*Caltech-101*" entrega el rendimiento asombroso de 99.7%, llega a caer hasta un 7.6% al ser probado en la base de datos "SUN-09". Esto deja en evidencia la clara existencia de sesgos en varias de las bases de datos de visión. Los métodos probados en el trabajo Torralba *et al.* [1] terminaron aprendiendo algunos de los sesgos propios de cada base de datos donde fueron entrenados y presentan una pobre capacidad de generalización a otras bases de datos, o al mundo real.

Es hecho de que las características pre-entrenadas CNN sean capaces de generalización en múltiples bases de datos, y sin ningún tipo de aprendizaje sobre el conjunto de entrenamiento, demuestran su alto poder separabilidad sin caer en el aprendizaje de los sesgos propios de las bases de datos.

2.2.3 Descripción base de datos

En esta sección se describe detalladamente la base de NYU Depth V1, que es la utilizada en este trabajo de tesis. La base de datos NYU Depth V1 es de particular interés pues hasta la fecha de realización de este trabajo y entre la bibliografía encontrada, es la única donde aún un método basado en características *hand-crafted* es el que obtiene el mejor rendimiento. Esto la hace idónea para probar algunas de las hipótesis planteadas en este trabajo. De igual manera, al final de esta sección se describen las bases NYU Depth V2 y SUN RGBD brevemente.

NYU Depth V1

La base de datos utilizada corresponde a la NYU Depth V1, correspondiente a una base de datos de segmentación semántica en interiores. La base de datos contiene archivos de video grabados con un Kinect 1 de distintos interiores domésticos. Son guardadas las imágenes RGB y de profundidad en formato RAW, y la información del acelerómetro. En total se tienen 108617

cuadros de video, correspondientes a 7 tipos de lugares y obtenidos de 64 habitaciones distintas. La Tabla 2-9 muestra en detalle las estadísticas de las clases de lugares.

Tabla 2-9. Información base de datos NYU Depth V1

Tipo de lugar	**n° de habitaciones**	***Frames***	***Frames* etiquetados**
Baño	6	5588	70
Dormitorio	17	22764	463
Librería	3	27173	781
café	1	1933	47
Cocina	10	12643	276
Sala de estar	13	19262	342
Oficina	14	19254	305
Total	64	108617	2284

Del total de cuadros disponibles, un subconjunto muestreado cada 2-3 segundos posee un etiquetado denso semántico, y una versión procesada de las imágenes de profundidad. La dimensión de este subconjunto etiquetado y con profundidad corregida es de 2284 imágenes. La Figura 2-8 muestra algunas imágenes de este subconjunto.

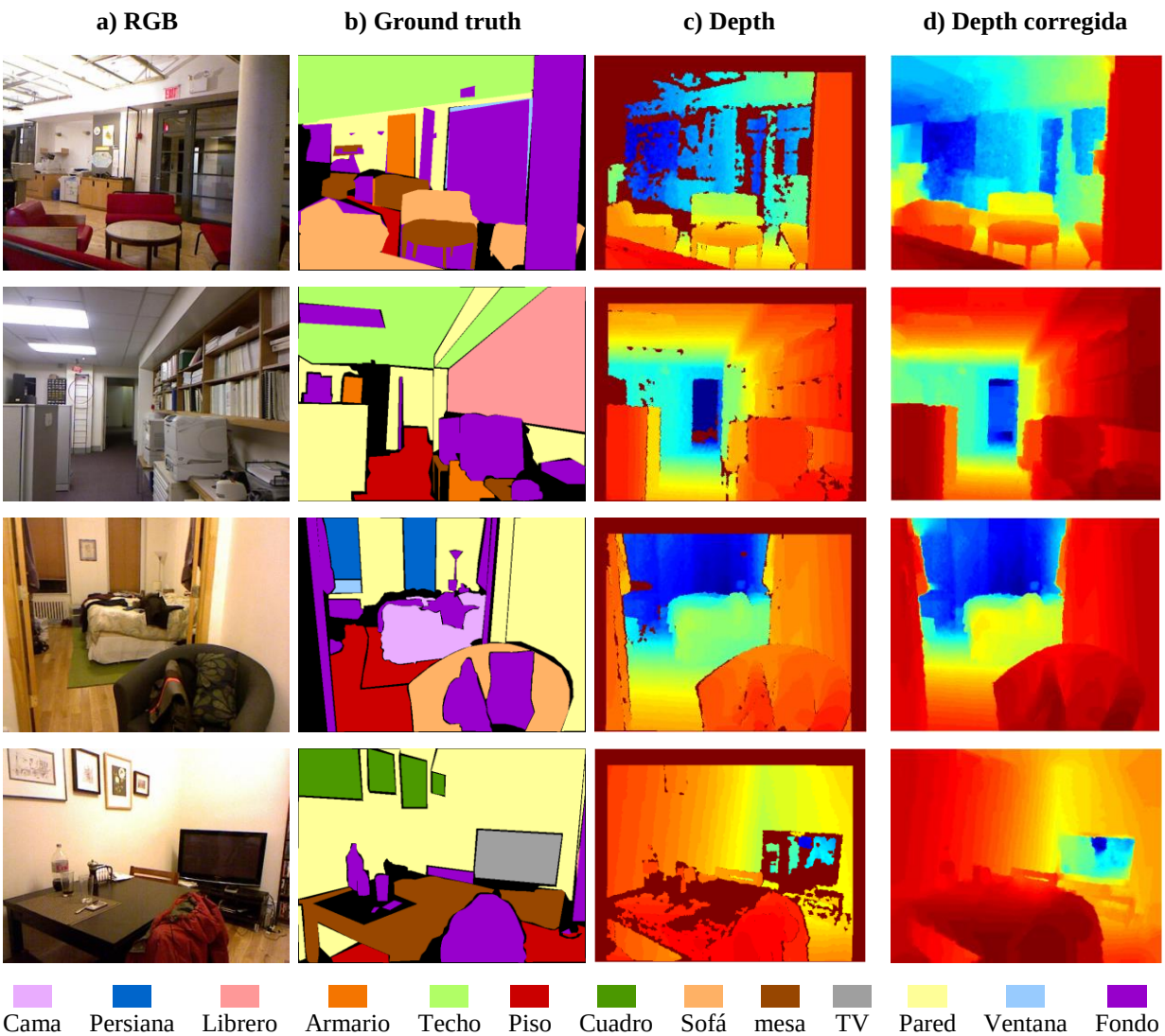


Figura 2-8 Imágenes de la data disponibles en NYU Depth v1. Columna a): Imágenes RGB; columna b): etiquetado obtenido mediante *Amazon Mechanical Turk*; columna c) Imágenes de profundidad, RAW; columna d) Imágenes de profundidad alineadas y corregidas.

El procesado de las imágenes de profundidad corresponde al alineamiento de estas respecto a las RGB, y de un filtro que rellena las zonas marcadas como lecturas inválidas de profundidad del Kinect. Estas zonas invalidas pueden verse Figura 2-8 columna c), en color rojo oscuro. Estas lecturas invalidadas se producen por tres motivos. El primero debido a las zonas donde el patrón de luz estructurada IR del Kinect no llega, debido a la separación entre el proyector IR y la cámara IR (separación necesaria para calcular la disparidad). El segundo motivo se debe a zonas se producen reflejos, y el tercero a zonas donde llega demasiada luz solar (debido a que componente IR presente en la luz solar enmascara el patrón del Kinect). Las imágenes de profundidad alineadas y corregidas pueden verse en la Figura 2-8 columna d).

Para generar el *ground truth* utilizan el servicio web *Amazon Mechanical Turk*, una plataforma donde es posible publicar un trabajo, y luego es llevado a cabo por terceras personas, inscritas en la plataforma de Amazon. Esta plataforma está pensada para trabajos que son simples, repetitivos, y demorosos, pero que no puede realizar una máquina (se requiere cierto grado de inteligencia

humana). El resultado del etiquetado mediante este servicio generó una segmentación semántica con más de 1000 clases distintas, que son reducidas a 13 clases mediante una estructura de sinónimos/antónimos basada en Wordnet. Estas 13 clases pueden verse al pie de la Figura 2-8. La clase fondo (*Background*, en el trabajo original*)* contiene todos los objetos que no calzan en las 12 categorías restantes. A su vez existe un grupo de pixeles que no tiene asignado ninguna etiqueta. Estos pixeles corresponden a las zonas en negro que se ven en el ground truth, Figura 2-8 columna b). Por lo general estas zonas sin etiquetar se encuentran entre las fronteras de un objeto y otro, y varían en tamaño, dependiendo de la prolijidad de la persona que realizo el etiquetado. Este grupo de pixeles sin etiquetar es ignorado al momento de entrenar y de calcular las métricas de rendimiento.

El trabajo de los creadores de la base de dato NYUDv1, además de proponer las 13 categorías semánticas para evaluar rendimiento, define un conjunto de entrenamiento y test generados al azar en una proporción de 60% y 40 % respectivamente. La métrica usada en este trabajo corresponde a la media de la diagonal principal de la matriz de confusión, calculada a partir de la clasificación a nivel de pixel sobre las 13 clases en el conjunto de test. Estos mismos conjuntos y métricas son usados en los trabajos posteriores y son usados en este trabajo también.

Un punto importante a destacar es que existe cierto grado de inconsistencia, o error, en el etiquetado en la base de datos. Esto se debe a que la tarea de etiquetado es realizada por cientos de individuos distintos, y a la dificultad para llevar a cabo un minucioso control de calidad desde la plataforma *Amazon Mechanical Turk*. Estos errores de etiquetados o inconsistencia se pueden ver la Figura 2-9. El primer caso, Figura 2-9 b), es cuando grandes partes de la imagen están sin etiquetar. Este error no es tan grave, pues estos segmentos son ignorados tanto para entrenar como para clasificar, pero es información útil que se desperdicia. El otro caso es el de la Figura 2-9 d), que es más grave, donde las etiquetas son asignadas incorrectamente. Para esa imagen se puede ver que la puerta es incorrectamente etiquetada como armario, y que la pizarra de la izquierda es incorrectamente etiquetada como mesa. Esto genera problemas directamente tanto al momento de entrenar como al de evaluar el rendimiento. El problema de los errores o inconsistencia en el etiquetado del ground truth

Este es un problema no menor y bastante común al momento de generar el ground truth de bases de datos de segmentación semántica. Algunas bases de datos han llevado a cabo soluciones como contratar gente in situ y llevar a cabo un exhaustivo control de calidad[54], con el fin de asegurar un etiquetado de buena calidad. Sin embargo el coste en tiempo y dinero es mucho mayor que con *Amazon Mechanical Turk*.

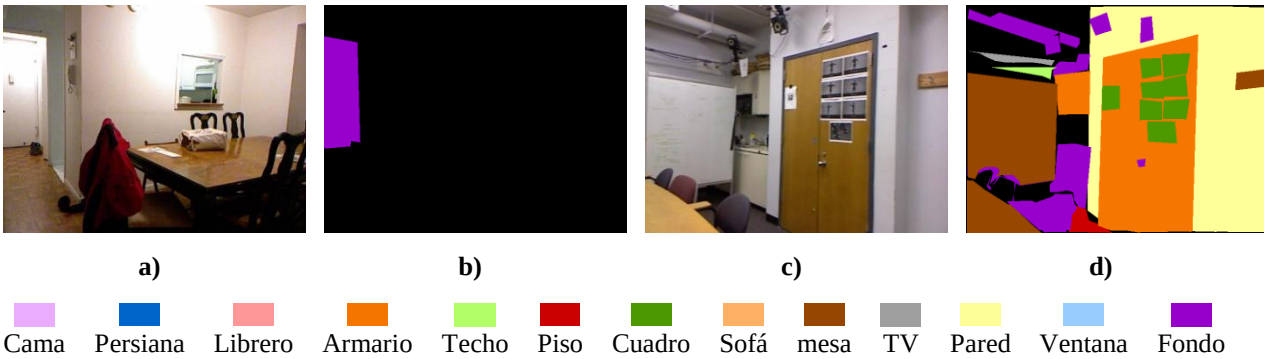


Figura 2-9 Imágenes de casos donde el ground truth esta incorrectamente etiquetado.

2.3 Discusión

En esta sección se discute respecto a las tres partes que componen este trabajo de tesis, y que fueron revisadas en las secciones anteriores. Estas partes son segmentación semántica, clasificación de lugares y uso de contexto de lugar, o alto nivel, en el problema de segmentación semántica.

La aparición de métodos basados en *deep* learning han revolucionado el área de visión computacional, obtenido rendimientos varios puntos porcentuales más arriba que los mejores métodos basados en características *hand-crafted* [20][36][4]. El mejor ejemplo de esto es el área de clasificación de imágenes, con la aparición de *Alexnet* [20], que dio el puntapié a múltiples trabajos basados en este enfoque (VGG16, Googlenet). El requisito principal para poder aplicar este enfoque es disponer de una gran base de datos etiquetada, como por ejemplo Imagenet (que dispone del orden de millones de ejemplos etiquetados). En consideración de la esta gran limitante, múltiples trabajos [6][38][4][36][5][26] se enfocaron en poder transferir el conocimiento de redes entrenadas en bases de datos grandes a problemas con bases de datos mucho más pequeñas. Un enfoque que ha tenido gran éxito en esta búsqueda por transferir el conocimiento, es el uso de características extraídas desde las últimas capas de redes previamente entrenadas [4] [6]. El uso de estas características extraídas en conjunto con clasificadores SVM ha permitido superar el estado del arte en varias bases de datos de clasificación de imágenes como “caltech101”, “birds”[4], entre otras. Desde el estado del arte se desprenden que este es el enfoque indicado para clasificar una pequeña base de datos de imágenes, como es el caso de NYU Depth v1 en el problema de clasificación de lugares.

Con respecto al área segmentación semántica, uno de los primeros trabajos post Alexnet es de Couprie et al. [18], donde usan un enfoque híbrido basado en la generación de superpixel y clasificación mediante SVM, pero con características generadas mediante una red CNN entrenada de forma supervisada. Sin embargo los primeros trabajos en mostrar un rendimiento considerable respecto a estado del arte fueron los basados en un enfoque de aprendizaje profundo “end-to-end”, como FCN[7], Segnet[10] o deeplab[9]. Sin embargo, aún existen bases de datos donde métodos basados en características *hand-crafted* siguen siendo el mejor resultado [43]. Un ejemplo de esto es la base de datos NYU Depth v1, donde el método basado en características *hand-crafted* KDE sigue siendo el mejor resultado hasta la fecha. Además este mismo método sigue superando a algunos métodos basados en *deep learning* end-to-end, en la base de datos SUN-RGBD [55]. Una de las explicaciones para esto es que algunas bases de datos de segmentación semántica (como NYU Depth v1) aún son considerablemente pequeñas, con respecto a bases de datos para clasificar imágenes como Imagenet. Otro factor que podría explicar estos casos es la calidad del etiquetado. Al ser los métodos basados en *deep* learning end-to-end totalmente dependientes de las etiquetas para aprender a generar las características y clasificar, cualquier error en estas etiquetas afectaría de mayor forma todo el proceso de aprendizaje. Esto a diferencia de la mayoría de los métodos basados en características *hand-crafted*, donde las características son generadas de forma manual y/o con enfoques no supervisados. Esto haría las características *hand-crafted* más robustas a errores de etiquetado. En particular este último punto, el error de etiquetado, podría ser el factor que más afecte a la base de datos NYUv1.

Con respecto al problema relacionado con el tamaño de la base de datos en el área de segmentación semántica, se ha intentado imitar [18] el enfoque de transferencia de conocimiento que ha dado muy buenos resultados en el área de clasificación de imágenes[4]. Este consiste en extraer características de redes para clasificar imágenes, como Alexnet, VGG u otras, y luego clasificar mediante el enfoque "clásico" de segmentación semántica. El enfoque "clásico" consiste el uso de algún tipo de superpixeles más un clasificador (generalmente SVM u random forest) y, de forma opcional, un refinamiento mediante algún método basado en modelos de probabilísticos de grafos, como CRFs (conditional random fields) o MRF (Markov random field). En particular para el problema de segmentación semántica en interiores domésticos no se encontraron trabajos que utilizaran características *deep* learning extraídas. Sin embargo existen trabajos en áreas como segmentación semántica de exteriores en ciudades. Estos trabajos extraen las características desde redes para clasificar imágenes. Si bien entregan un rendimiento aceptable, no se observan los resultados sobresalientes como en el caso de clasificación de imágenes. Desde aquí nace la motivación para utilizar características extraídas directamente desde una red para segmentación semántica end-to-end. Su rendimiento podría ser mayor que el de las características extraídas de redes para clasificar imágenes, aunque podrían ser mucho más específicas. Este último punto se refiere a que las características extraídas de una red para segmentar interiores domésticos solos servirían para este problema, y no por ejemplo para segmentar imágenes de exteriores en ciudades.

Otra forma de transferir el conocimiento es mediante fine-tuning, ajustando los pesos de una red ya entrenada a un problema similar. Esto se hace inicializando la red con los pesos de otra red ya entrenada y comenzar el aprendizaje desde ese punto. Las ventajas que podría tener usar un enfoque hibrido, usando algún tipo de superpixel más SVM, vs fine-tuning, son explicadas a continuación. Por un lado, donde mayor incertidumbre se presenta en las redes *deep* de segmentación end-to-end es en los contornos de los objetos. Al usar super-pixeles basados en detectores de contornos, podrían definirse mejor estas fronteras. Por otro lado, el clasificador SVM realiza una optimización global a partir de toda la data de entrenamiento, mientras que las redes *deep* solo realizan una aproximación de esto mediante entrenamiento por batch, entrenando iterativamente a partir de pequeños subconjuntos de la data de entrenamiento.

Para obtener buenos resultados al momento de transferir conocimiento, se tiene como requisitos que la red desde donde se extraiga el conocimiento tenga características con gran capacidad de generalizar y que el problema donde fueron entrenadas tenga cierta relación con el problema donde se buscan transferir. De forma intuitiva para que se cumpla esto, la base de datos desde donde se desea extraer las características debe ser bastante más grande que base donde se van a transferir. A su vez ambos problemas deben tener cierta relación, siendo el caso ideal donde el problema al que se buscan transferir las características sea un sub-problema dentro de la base de datos de la red donante de características. Este es el caso de muchas de las bases de datos de imágenes pequeñas donde la gran mayoría de las clases de estas bases de datos están ya en Imagenet. Existen casos donde no se da esta relación directa de las clases, pero gracias a la inmensa cantidad de clases de Imagenet, se logran generar características útiles para estos problemas. Para el caso del clasificador de lugares, es de interés ver que tan relevante es esta diferencia. Por ese motivo se extraen características desde una red entrenada en Imagenet y luego de esa misma arquitectura de red entrenada en Place205, comparando luego sus resultados.

Relacionado con el problema de segmentación semántica, la red desde donde se busca extraer las características debe cumplir con los mismos criterios. Dentro de las bases de datos disponibles

para segmentación semántica, SUN RGBD es la más grande por amplio margen, por lo que es lógico que sea una red entrenada en esta base de datos desde donde se extraigan las características. Con respecto a la arquitectura de la red, esta debe tener el mayor rendimiento posible, además de poseer códigos y modelos pre-entrenados disponibles. La red que cumple con todos estos requisitos en Segnet [8]. Si bien se sitúa en el tercer lugar en el ranking de SUNRGBD hasta la fecha de realización de este trabajo, tiene como ventaja adicional que su arquitectura es bastante simple, en comparación a otros trabajos existentes. Esto permite extraer características densas de sus últimas de forma directa.

El último punto corresponde al uso de contexto de lugar, o alto nivel, en el problema de segmentación semántica. De forma intuitiva el conocer cuál es el lugar que se está observando, tiene relación con la probabilidad de encontrar o no encontrar ciertos objetos. Por ejemplo si está observando un dormitorio, la probabilidad de encontrar una cama es más alta que encontrar un sillón. Esta relación también puede darse de forma inversa, es decir que a partir de los objetos que encuentro, puedo obtener información sobre el tipo de habitación en la que me encuentro.

3. Sistema propuesto

A partir de la discusión del estado del arte se propone un sistema que permita corroborar las hipótesis planteadas. Para esto se utiliza como base el sistema que obtiene el mejor resultado en la base de datos NYU Depth v1 [43], que utiliza como base características KDE RGB-D y clasificadores SVM. Las características KDE RGB son reemplazadas por las características densas extraídas de las capas finales e intermedias de la red *deep* Segnet [10] previamente entrenada en otra base de datos distinta (SUN-RGBD). Dado que Segnet solo extrae características RGB, se siguen utilizando las características KDE para la imagen de profundidad. La clasificación se realiza a nivel de superpixeles y se utiliza un clasificador SVM multiclase. Los superpixeles son generados mediante un método para detectar bordes en imágenes y segmentarlas de forma jerárquica, gPb-ucm[21]. En paralelo, se ejecuta un clasificador de lugares y se utiliza información de esta etapa para mejorar el resultado de segmentación semántica. Esto corresponde, a grandes rasgos, a la descripción del sistema propuesto. La Figura 3-1 muestra un esquema completo del sistema propuesto.

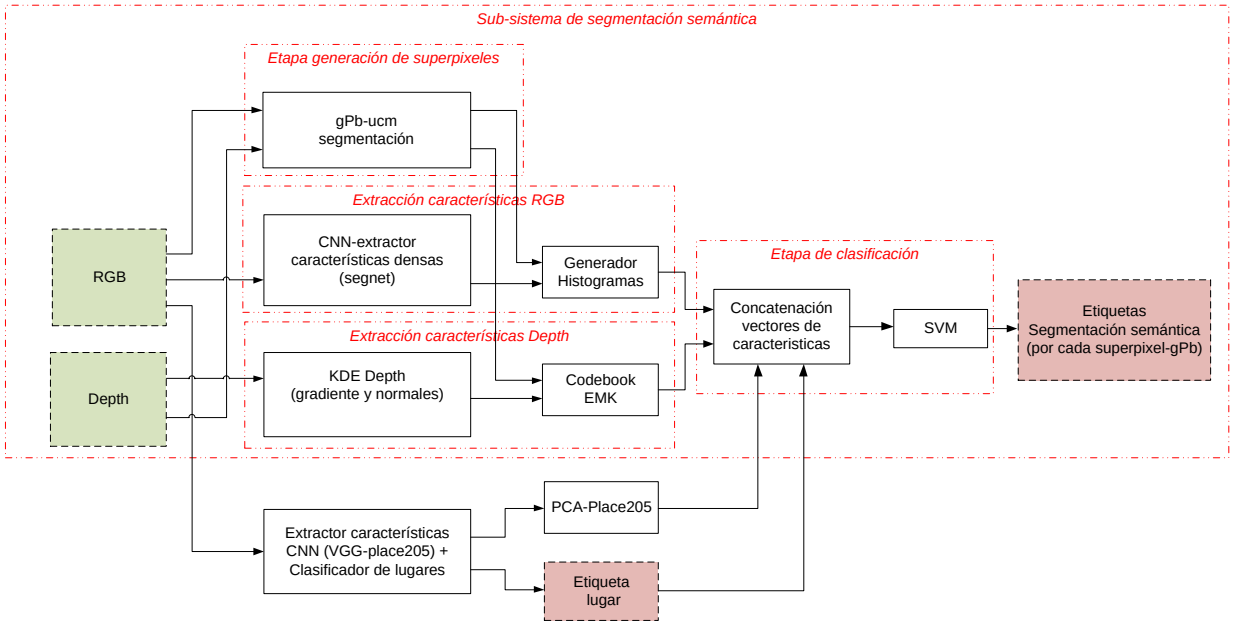


Figura 3-1 : Esquema general del sistema propuesto

El Capítulo 3 se divide en dos secciones principales. La 3.1, donde se describen en detalle el sub-sistema de segmentación semántica y la 3.2, donde se describe el sub-sistema para clasificar lugares y generar la información de contexto para el sub-sistema de segmentación semántica.

3.1 Sub-sistema de segmentación semántica

En esta sección se describe el subsistema encargado de la segmentación semántica. Este sub-sistema recibe como entradas las imágenes RGB-D y entrega a la salida las etiquetas correspondientes a la segmentación semántica. Está compuesto por cuatro etapas principales. A

continuación, se describe brevemente cada una de estas etapas, para luego explicarlas en detalle en las siguientes secciones de este documento.

- *Etapa de generación de superpixeles* - En esta etapa se generan los distintos segmentos o zonas a clasificar. Recibe como entrada las imágenes RGB-D y entrega a la salida superpixeles, que corresponden a segmentos de la imagen. El método utilizado corresponde a gPb-ucm [21]. Este método se divide en dos etapas, la primera gPb[56] que corresponde a un algoritmo entrenado de forma supervisada para detectar los contornos de los objetos en una imagen, y la segunda, ucm, que se encarga de generar una segmentación jerárquica a partir de estos bordes detectados.

- *Etapa de extracción característica RGB* - Este módulo recibe como entradas la imagen RGB y los segmentos de la *etapa de generación de superpixeles* y entrega a la salida un vector de características por cada una de estas regiones. Para lograr esto lo primero que se realiza es la extracción de características densas desde las últimas capas de la red *deep* convolucional Segnet (previamente entrenada en SUN-RGBD). Luego para cada superpixel se calcula un histograma de las características densas en esa región.

- *Etapa de extracción característica Depth* – Las características de la imagen de profundidad se extraen de la misma forma que en el trabajo de Ren *et al.* [43]. Esto corresponde a la extracción densa de las características KDE de gradiente y KDE spin/normales de superficie. Una vez obtenidas estas características densas KDE Depth, se calcula un vector de características en cada superpixel obtenido en la primera etapa. El último paso se realiza mediante EMK, que corresponde a un método alternativo a *Bag-of-words* (BOW). Al igual que BOW, al momento de entrenar se genera un diccionario o *codebook*, y al momento de evaluar se recurre a este *codebook* para generar el vector de características.

- *Etapa de clasificación* – La última etapa corresponde a la clasificación. En esta etapa se concatenan todos los vectores de características de las etapas anteriores y se clasifican utilizando SVMs. La salida de esta etapa corresponde a la etiqueta con la clase para cada superpixel en la imagen.

3.1.1 Etapa generación de superpixeles

En esta etapa se generan las regiones en las que se segmenta la imagen, a partir de un sistema basado en el detector de bordes gPb/UCM (Global Probability Boundary - Ultrametric Contour Map), propuesto por Arbelaez et al. [21]. Este método también posee la posibilidad de trabajar con una segmentación jerárquica, para poder definir distintos grados de segmentación en la salida. En esta tesis se utiliza el método utilizado en el trabajo de Ren et al. [43], que corresponde a gPb/UCM implementado en imágenes rgb-D.

La etapa gPb (Global Probability Boundary) corresponde a un detector de contornos, que entrega la función de probabilidad de $P_b(y, \theta)$. Esta función predice la probabilidad de que un punto (x, y) corresponda al contorno de una imagen, dada también una orientación θ. Esto a partir de características generadas con información basada en las diferencias de brillo, color y textura en la imagen. Se utiliza la misma implementación del trabajo de Ren et al. [43], cuyo código se encuentra disponible. El detalle de esta etapa está en la sección 3 de este documento.

3.1.2 Extracción de características RGB

Las características RGB se extraen de la red *deep* pre-entrenada tipo encoder-decoder Segnet. Esta red se encuentra previamente entrenada, en la base de datos SUN RGB-D [55]. La base de datos SUN RGB-D también corresponde a una base de datos de segmentación semántica en interiores domésticos. Posee 10355 imágenes y 38 clases etiquetadas, mientras que NYUv1 posee 2284 imágenes y 13 clases. Dado que las características a extraer fueron entrenadas en una base de datos más grande y en un problema más complejo (38 clases vs 13 clases), se espera que entreguen un buen rendimiento.

Inspirado en los trabajos basados en la extracción de características de redes pre-entrenadas para clasificar imágenes [6] [4] [5], se busca extraer características de las últimas capas de la red *deep* Segnet. Se espera que estas últimas capas generen las características que podrían ser más útiles para ser transferidas a otro problema de clasificación. Se seleccionan como opciones todas las capas finales, anteriores a la salida, que posean la misma resolución que la imagen de entrada (480x360). Este criterio deja las últimas 3 capas de Segnet como alternativas para extraer características. En la Figura 3-2 se pueden observar las últimas 3 capas elegidas, que corresponden a una de upsampling, una convolucional y, finalmente, una *softmax*. En la Tabla 3-1 se muestra las dimensiones de cada una de estas capas. A su vez se les asocia un índice a cada una de ellas, tomando como referencia la capa de salida, índice 0, y luego enumerando las anteriores de forma negativa (-1,-2 y -3).

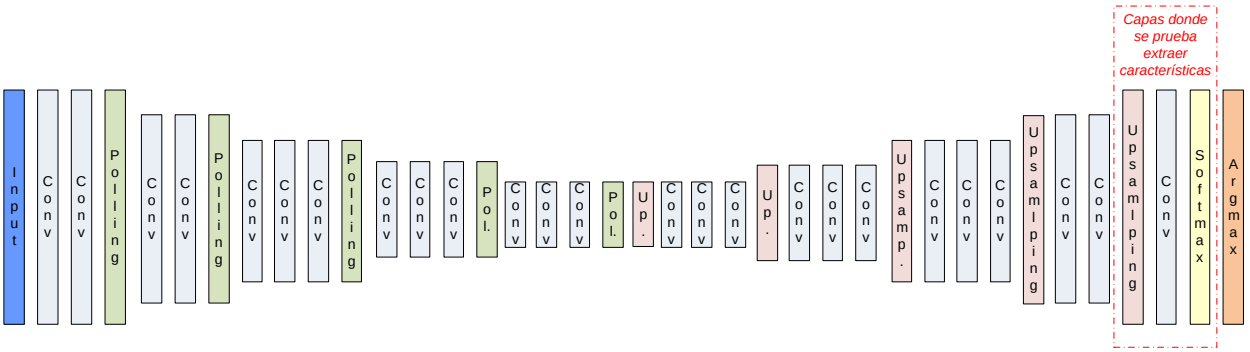


Figura 3-2: Capas de Segnet donde se extraen características densas

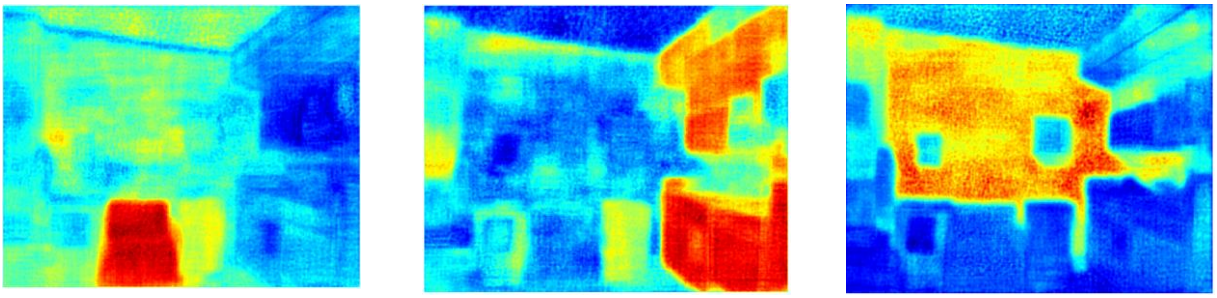
Tabla 3-1. Dimensiones de las 4 últimas capas

Capa	Dimensión de salida	Índice
Upsampling	[480,360,64]	-3
Conv-relu	[480,360,64]	-2
Softmax	[480,360,38]	-1
Argmax (salida)	[480,360]	0

Con respecto a estas capas, la primera (más profunda) corresponde a upsampling, que se encarga de aumentar la resolución desde la capa anterior. Esta capa contiene 64 matrices de dimensión 480×360. Estas matrices suelen llamarse canales o *feature-maps*. La información de estas capas de la red es abstracta, pero se pueden interpretar como las características que la red va aprendiendo. La segunda que le sigue es una relu-convolucional, que corresponde, como lo dice su nombre, a una convolución seguida de la no linealidad relu. Esta capa también posee la misma dimensión de la capa anterior, y corresponde a un nivel más arriba en la generación de las características. La penúltima capa es un clasificador *softmax*, que corresponde a *n* matrices de 480x460, siendo *n* el número de clases semánticas del problema donde fue entrenada la red. Como esta red fue pre-entrenada en la base de datos SUN-RGBD, esta capa posee un valor de *n* igual a 38 (Correspondientes a las clases en SUN-RGBD). El clasificador *softmax* entrega un grado de certidumbre para cada una de las clases por cada pixel en la imagen. Para obtener las etiquetas finales, a cada pixel se le asigna la etiqueta con la clase que posee el mayor grado de certidumbre correspondiente a este pixel. Esto es, el máximo para esa coordenada entre las 38 clases. Esta es la función que realiza la capa final *argmax*. A continuación se grafican cada una las capas mencionadas, probándola con una imagen de NYU Depth v1.

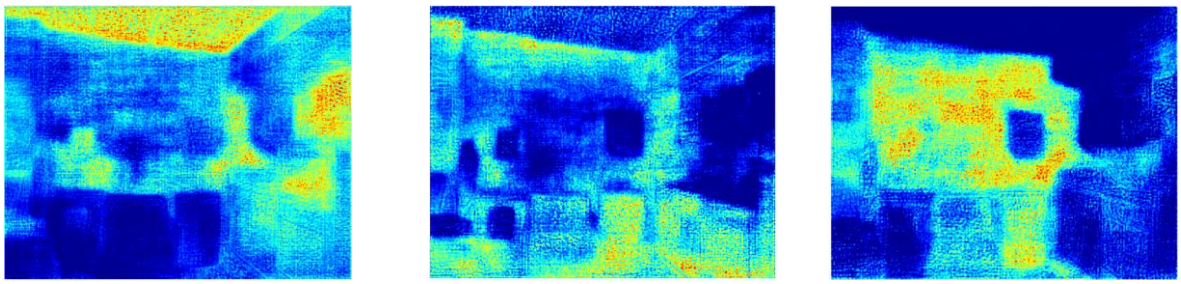
a) **b)**

Figura 3-3 : **a)** Imagen de entrada **b)** Imagen salida Segnet (*argmax*)



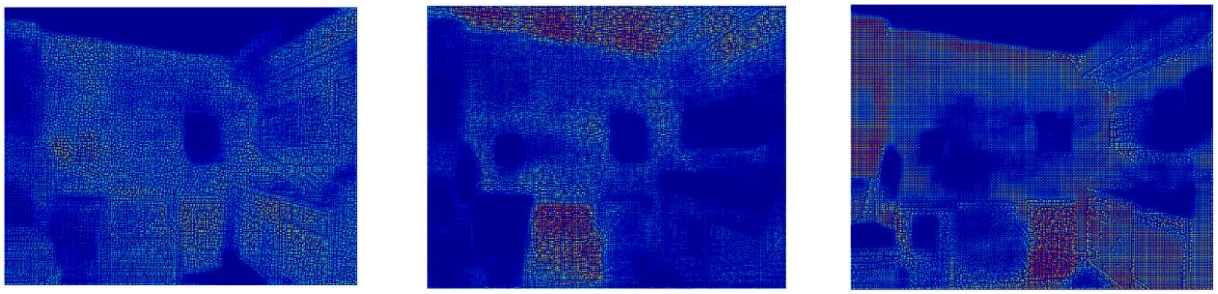
a) **b)** **c)**

Figura 3-4 : **a) b)** y **c)** *feature-maps* de la capa *Softmax*. Se muestran 3 al azar, de las 38 existentes.



a) **b)** **c)**

Figura 3-5 : **a) b)** y **c)** *feature-maps* de la capa *Conv-relu*. Se muestran 3 al azar, de las 64 existentes.



a) **b)** **c)**

Figura 3-6 : **a) b)** y **c)** *feature-maps* de la capa *Upsampling*. Se muestran 3 al azar, de las 64 existentes.

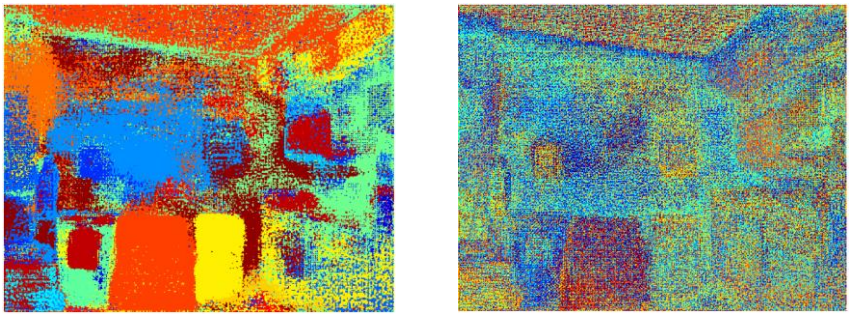
En la Figura 3-4 **a) b)** y **c)** se pueden ver 3 *feature-maps* de los 38 que se generan en la capa *softmax*. Cada uno de estos *feature-maps* corresponde a la probabilidad de una clase de la base SUN-RGBD. Se puede ver, por ejemplo que la Figura 3-4 **a)** corresponde a la probabilidad de un sillón, la Figura 3-4 **a)** a la probabilidad de ser un mueble y Figura 3-4 **c)** a la probabilidad de ser pared. La última capa, Figura 3-3 **b)** *arg-max*, entrega la clase más probable para cada pixel, que corresponde al máximo valor para ese pixel entre los 38 *feature-maps*. La Figura 3-5 muestra 3

feature-maps de los 64 de la capa convolucional antes del clasificador *soft-max*. Estas imágenes representan 64 características abstractas que genera la red Segnet en esta capa. Se puede ver que existe una relación entre algunos objetos de la imagen de entrada y estos *feature-maps*. La Figura 3-6 corresponde a 64 *feature-maps* que se generan en la capa de *upsampling*. Este número, 64, es el tamaño estándar de la capas de características de Segnet antes de la etapa de clasificación. En esta capa también se puede ver cierta relación entre los *feature-maps* que genera y los objetos en la imagen de entrada, aunque no es tan marcada como en las capas siguientes.

Antes de extraer las características, se realiza la siguiente prueba preliminar con la capa de salida (*arg-max*). Dado que esta red Segnet, entrenada en SUN-RGBD, ya entrega una segmentación semántica de interiores, la alternativa más simple es reasignar cada una de las 38 clases de la salida, a una de las 11 clases de NYU Depth v1. El resultado de esta prueba es 67.19%, que es mayor al que entregan las características KDE RGB solas, 65.79%, aunque muy por debajo del máximo que se obtiene con el sistema KDE completo, 76.35%. La prueba anterior sólo usa información correspondiente a la capa de salida o *argmax*. Esta información corresponde al valor de la mayor activación entre las 38 capas *soft-max*. El resultado de esta prueba, 67.19%, se considera como *baseline* para el resto de las pruebas.

Para generar las características que se utilizan en la etapa de segmentación, el enfoque utilizado es generar un vector de características para cada *superpixel*. Este vector de características se genera a partir de la información de las características densas que se encuentran dentro de la región del *superpixel*. Las estrategias propuestas para generar el vector de características a partir de las características densas extraídas de Segnet son las siguientes.

Valor máximo de activaciones por feature-map: Dado que la última capa de Segnet, *Argmax*, clasifica a partir del valor de la mayor activación entre los *feature-maps* de la capa *Soft-max*, una alternativa sería utilizar información de este tipo para generar las características a nivel de *superpixeles*. Si bien la información de la máxima activación es discriminante para la capa *Soft-max*, no es claro que esto se cumpla en capas anteriores de la red. A modo de prueba cualitativa se aplica la función *argmax* a las capas -2 y -3, la Figura 3-7 muestra el resultado. En esta figura puede verse de forma cualitativa que en las capas anteriores de la red ya existe cierta relación entre el valor máximo de las activaciones y los objetos presentes en la imagen de entrada. También se aprecia que, a medida que la capa está más alejada de la salida de la red, esta relación es menos clara, ver la Figura 3-7.



a) **b)**

Figura 3-7 : **a)** Resultado aplicar capa de salida (*argmax*) a capa -2, *conv-relu*.
b) Resultado aplicar capa de salida (*argmax*) a capa -3, *upsampling*.

Para generar las características, por cada *superpixel* generado, se crea un vector donde se almacenan los valores de la máxima activación de cada *feature-map.* La dimensión del vector de características obtenido es igual a la cantidad de *feature-maps* existente en la capa. Estas se pueden ver en la Tabla 3-6.

Tabla 3-2 Dimensiones vector caracterìstica, caracterìstica màxima activación

Capa	**Dimensión vector característica**
Softmax (-1)	38
Conv-relu (-2)	64
Upsampling(-3)	64

Histograma de las activaciones por característica map: De forma de almacenar más información que solo el valor de la mayor activación, se genera un histograma de las activaciones de los *feature-maps*. Se calcula un histograma por cada *feature-map,* de una capa determinada, y luego estos histogramas son concatenadados. Estos histogramas se calculan dentro de las regiones de cada *superpixel*. La dimensión del vector de características resultante es igual al número de *feature-maps* multiplicado por el largo del histograma. Al igual que en la caso anterior, se obtiene un vector de características por cada superpixel. Se prueban las tres capas como alternativas para extraer las características. Como primera prueba se establece el largo del histograma en 10 para cada capa. Luego para la capa que entregue mejores resultados, se prueba además con largos de 5 y 15.

Tabla 3-3 Dimensiones vector característica. Histogramas de activaciones, primera prueba

Capa	**Dimensión vector característica**
Softmax (-1)	380
Conv-relu (-2)	640
Upsampling(-3)	640

Con respecto a los rangos de los histogramas, el primer enfoque a probar es asegurar que todos los valores queden al interior del histograma. Para esto se analizan todas las muestras del conjunto de entrenamiento, por canal (o *feature-map*) y se establecen los límites del histograma con el valor máximo y mínimo encontrado. Estos límites se estable en forma individual para cada *feature-map*. Un inconveniente de este enfoque, es que estos valores extremos encontrados pueden estar alejados de los valores más usuales dentro de los *feature-maps,* por lo que algunos

de los intervalos del histograma podrían quedar muchas veces vacíos. Estos "bines" vacíos se generarían a los extremos del histograma, y una buena opción es eliminarlos. Esto además entregaría más resolución en la zona del histograma que contiene mayor información. Se debe tener en cuenta que si este intervalo se acorta demasiado, llega un punto donde se comienza a perder información útil.

Teniendo en cuenta que en cada *feature-map* la data distribuye de una forma distinta, podría resultar muy arbitrario recortar solo a partir de un porcentaje del valor máximo/mínimo y utilizar ese porcentaje para todos los *feature-maps*. Por ejemplo, recortar en un 10% el límite máximo y mínimo podría resultar bien para el *feature-map* número 1, pero para el *feature-map* número 2 podría dar un mejor resultado recortarlo en un 15%, por ejemplo. Por este motivo se utiliza un umbral distinto según cada *feature-map.*

Los límites de los histogramas quedan definidos a partir de la forma en que distribuye los valores máximos y mínimos, en cada *feature-map*. El proceso para definirlos es el siguiente, con $x \in [0,100]$.

1. Se almacenan los valores mínimos de cada *feature-map*.
2. Se almacenan los valores máximos de cada *feature-map*.
3. Se estable como el límite inferior del histograma el percentil x del conjunto de valores mínimos.
4. Se estable como el límite superior del histograma el percentil 100- x del conjunto de valores máximos.

De esta forma, se fija el intervalo de los histogramas de cada mapa de *feature-map* de manera específica, en función de cómo distribuyen sus extremos. Los valores en los que se prueba el valor de x son los siguientes:

Tabla 3-4 Valores x a probar

Valor x	0	25	50	75

3.1.3 Extracción de características Depth

Las características de profundidad utilizadas son las mismas del trabajo que obtiene el mejor resultado hasta la fecha en la base de datos NYU Depth v1, de ren et al [43]. Estas características corresponden a los Kernels Descriptors, KDE. Para este trabajo sólo se utilizan los KDE de profundidad, pues para extraer las características RGB se utiliza la red Segnet.

Los KDES permiten calcular descriptores densos sobre un segmento determinado de una imagen, a partir de una función similitud a nivel de pixeles. Como ejemplo, se describe brevemente el kernel descriptor de gradiente sobre un segmento de una imagen de profundidad. Los detalle de cómo se definen estas características se encuentra en el Capítulo 2 de este documento.

Las imágenes de profundidad se tratan igual que imágenes en escala de grises, calculando su gradiente a nivel de pixel. El kernel descriptor de gradiente F_{grad} se construye a partir de la función de similitud de gradiente k_o, ver ecuación (3-1).

$$F_{grad}^{t}(\mathbf{Z}) \sum_{i=1}^{d_o} \sum_{j=1}^{d_s} \alpha_{ij}^{t} \left\{ \sum_{z \in \mathbf{Z}} \tilde{m}_z k_o\left(\tilde{\theta}_z, p_i\right) k_s(z, q_i) \right\} \tag{3-1}$$

$$k_o\left(\tilde{\theta}_z, \tilde{\theta}_x\right) = \exp(-\gamma_o \left\| \tilde{\theta}_z - \tilde{\theta}_x \right\|^2) \tag{3-2}$$

$$k_s(z, x) = \exp(-\gamma_s \|z - x\|^2) \tag{3-3}$$

Donde **Z** es un segmento de la imagen de profundidad y *z* la posición relativa de pixel dentro del segmento **Z** (normalizada entre [0,1]). $\tilde{\theta}_z$ y $\tilde{m}_z$ son la orientación y magnitud del gradiente de profundidad normalizados, en un pixel *z*. El kernel de orientación (ec. (3-2) calcula la similitud de la orientación de los gradientes. El kernel gaussiano de posición (3-3) mide que tan cerca son dos pixeles espacialmente. Los conjuntos de pixeles $\{p_i\}_{i=1}^{d_o}$ y $\{q_j\}_{j=1}^{d_s}$ son muestreados de forma uniforme en segmento **Z**. d_o y d_s son la cantidad de vectores bases muestreados para los kernels de orientación y posición. α_{ij}^{t} son los coeficientes proyectados usando KPCA (kernel principal component analysis). Otros kernel descriptors se construyen de forma análoga, utilizando distintas funciones para medir similitud a nivel de pixeles, como lo pueden ser LBP o SIFT. En este trabajo se utilizan los mismos kernel descriptor utilizados en el trabajo de Ren et al [43] .

3.1.4 Clasificación

El clasificador usado corresponde a un SVM de la librería liblinear 2.01. Esta librería está optimizada para problemas de alta dimensionalidad y gran cantidad de muestras, además de poder trabajar con varias clases (SVM all-vs-one). El problema abordado en esta tesis cumple con todos estas características, es por eso que esta librería también es usada en el trabajo de Ren et al. [43]. La cantidad total de muestras (segmentos de imagen) para entrenar/clasificar es de 351612, con un vector de características de dimensión 640, el cual puede llegar hasta dimensión 1067, al agregar las características de profundidad y contexto, en el sistema final.
Dado que cada superpixel a clasificar es de distinto tamaño y además existe una asimetría en los tamaños de las clases, una pregunta válida es como considerar estos dos factores al momento de entrenar. Respondiendo a esta interrogante, se considera un factor de peso a cada muestra en el conjunto de entrenamiento. Este factor es directamente proporcional al área del *superpixel* e inversamente proporcional a la cantidad total de pixeles en el conjunto de entrenamiento de la clase a la que pertenece el superpixel. De esta forma los *superpixeles* muy pequeños se ven penalizados en importancia al momento de entrenar, cobrando mayor relevancia los de mayor área. Por otro lado, las clases que poseen una menor cantidad de muestras y donde el clasificador podría tender a ignorarlas, cobran mayor relevancia gracias al segundo factor. Escrito más formalmente, sea s un superpixel de clase c, A_s el área de s, P_c el conjunto de todos los superpixeles p de clase c y cada uno con área A_q, entonces la función de peso $w(s)$ queda expresada como:

$$w(s) = A_s \Big/ \sum_{q \in P_c} A_q$$

Los conjuntos de entrenamiento y test, así como las métricas de rendimiento, son las mismas utilizadas por los trabajos a compararse. Esto corresponde a conjuntos de entrenamiento y test separados en proporción 60% y 40% respectivamente, generados de forma aleatoria bajo una distribución uniforme. La evaluación del rendimiento se mide a nivel de pixeles, considerando cada pixel una muestra a evaluar. Se calcula la matriz de confusión para las 13 clases existentes, dejando de lado los pixeles sin etiquetar (zonas en negros en imágenes del *ground-truth*). Se utilizan como métricas de rendimiento la diagonal principal de la matriz de confusión (en forma de porcentaje), que corresponde al porcentaje de pixeles correctamente clasificados bajo una clase c; y además la media de la diagonal de la matriz de confusión (como indicador global del rendimiento). Estas mismas métricas y enfoque para entrenar el clasificador, en el problema de segmentación semántica, son usados a lo largo de todo este trabajo de tesis.

3.1.5 Resultados

En esta etapa se hace necesario presentar ciertos resultados intermedios, correspondientes a las distintas alternativas de características CNN extraídas de Segnet mostradas en este capítulo. Esto con el objetivo de definir la mejor opción para obtener las características RGB, la cual será utilizada durante la continuación este documento, a menos que se especifique lo contrario. En esta sección solo se prueban las características RGB extraídas de Segnet, dejando de lado las pruebas con características de profundidad y de contexto. Las pruebas y resultados correspondientes al sistema completo se mostrarán en el Capítulo 4 de este documento.

Las pruebas se realizan con cada una de las opciones propuestas en la sección 3.1.2 para generar las características RGB, a partir de las características densas extraídas de Segnet. Los primeros resultados se observan en la Tabla 3-5, que corresponden a las características generadas a partir de los valores máximos de activación, para las capas -1,-2 y -3.

Tabla 3-5 Resultados Segmentación semántica utilizando características extraídas valores máxima activación

	Cama	**Persiana**	**Librero**	**Armario**	**Techo**	**Piso**	**cuadro**	**Sofá**	**Mesa**	**TV**	**Pared**	**Ventana**	**Fondo**	**Media**
MaxCap-1	73.8	64.9	87.3	68.3	82.7	92.8	71.3	70.3	54.8	75.0	87.4	75.3	16.6	**70.81%**
MaxCap-2	83.9	68.9	83.3	67.7	84.5	86.4	70.2	75.4	50.9	76.4	75.8	65.5	7.5	68.97%
MaxCap-3	77.9	64.8	77.5	63.6	81.3	82.6	71.5	73.1	52.4	74.1	71.3	66.5	9.1	66.58%

En la Tabla 3-5 se observa es una disminución del rendimiento medio a medida que se acerca a las capas más internas de la red. Adicionalmente, llegando a la capa -3 el rendimiento es más bajo que el *baseline* establecido para estas características, de 67.19%. Este *baseline* corresponde al resultado de reasignar cada una de las 38 etiquetas de SUN-RGBD a una de las 13 de NYU Depth v1 y evaluar Segnet pre-entrenada.

Luego se realizan las pruebas con características generadas a partir de histogramas, de largo $n = 10$, de los *feature-maps*. Para cada capa, se prueban las 4 opciones posibles para establecer

los límites de los histogramas (Tabla 3-4). Se prueban las capas -1, -2 y -3, y los resultados se pueden ver en la Tabla 3-6, Tabla 3-7 y Tabla 3-8 respectivamente.

Tabla 3-6 Resultados Segmentación semántica utilizando características extraídas capa -1 *Soft-max*

Capa -1 Soft-max	**Cama**	**Persiana**	**Librero**	**Armario**	**Techo**	**Piso**	**Cuadro**	**Sofá**	**Mesa**	**TV**	**Pared**	**Ventana**	**Fondo**	**Media**
His10_0	85.3	73.6	88.3	67.8	87.6	90.6	77.0	78.4	55.9	85.9	83.6	66.5	21.7	74.01%
His10_25	85.9	73.5	87.8	68.3	87.6	90.4	77.4	77.5	57.5	87.7	84.0	66.9	23.9	**74.50%**
His10_50	84.2	75.4	87.6	68.7	86.9	90.7	75.8	78.6	57.6	86.4	83.8	69.0	23.6	74.49%
His10_75	85.8	73.9	87.4	68.3	87.1	91.0	76.2	76.5	56.7	86.4	83.8	69.9	23.9	74.37%

Tabla 3-7 Resultados Segmentación semántica utilizando características extraídas capa -2 *Conv-Relu*

Capa -2 Conv-Relu	**Cama**	**Persiana**	**Librero**	**Armario**	**Techo**	**Piso**	**Cuadro**	**Sofá**	**Mesa**	**TV**	**Pared**	**Ventana**	**Fondo**	**Media**
His10_0	85.4	73.7	87.7	69.1	87.3	89.8	75.9	81.3	57.4	82.3	82.9	69.9	23	74.27%
His10_25	86.7	74.6	88.2	70.5	88	90.8	76.4	80.8	57.7	84.6	83.6	69.1	23.6	**74.98%**
His10_50	86.4	74	88	70.9	87.7	90.8	76.3	80.7	57.8	84.7	83.9	69.4	24	74.97%
His10_75	86.2	74.5	88	71.2	87.6	90.9	76.4	81.2	57.5	84.3	84	68.6	24.1	74.96%

Tabla 3-8 Resultados Segmentación semántica utilizando características extraídas capa -3 *Up-sampling*

Capa -3 Upsam.	**Cama**	**Persiana**	**Librero**	**Armario**	**Techo**	**Piso**	**Cuadro**	**Sofá**	**Mesa**	**TV**	**Pared**	**Ventana**	**Fondo**	**Media**
His10_0	82	71.2	88.4	66.7	83.5	89.6	70.4	77.6	47.9	82.2	79.9	71.4	9.55	70.79%
His10_25	84.5	71.1	88.9	68.5	86	90.3	78	76.4	55.4	88.7	83	69.1	18.3	73.71**%**
His10_50	84.9	69.7	89	68.1	86.7	90	76.7	78.2	56	87.6	82.7	68.6	19.7	73.69%
His10_75	86.5	68.2	89	67.9	87.5	90.1	77.3	77.5	56.8	87.8	82.5	72.2	20.1	**74.11%**

Los resultados anteriores, con histogramas de las activaciones en vez de solo sus valores máximos, muestran un aumento de rendimiento de 4% aproximadamente, en comparación con los resultados obtenidos en la tabla Tabla 3-5. El mejor resultado se observa para la capa -2, *Conv-Relu.*

Finalmente, para el mejor resultado obtenido en los experimentos anteriores (que corresponde a las características extraídas de la capa -2 con histogramas recortado al 25%), se prueba nuevamente, pero agregando distintos largos de histograma. Los largos de histograma probados son de 5, 10 y 15 bines, con los extremos recortados al percentil 25. Esto genera vectores de características de largo 320, 640 y 960, respectivamente. Los resultados se observan en la Tabla 3-9.

Tabla 3-9 Resultados Segmentación semántica características Capa -2 Conv-Relu 25% distintos largos histograma

	Cama	**Persiana**	**Librero**	**Armario**	**Techo**	**Piso**	**Cuadro**	**Sofá**	**Mesa**	**TV**	**Pared**	**Ventana**	**Fondo**	**Media**
His5_25	86.6	73	88.5	69.2	87.8	90.5	76.8	79.7	57.1	87.8	83.5	68.4	22.2	74.70%
His10_25	86.7	74.6	88.2	70.5	88	90.8	76.4	80.8	57.7	84.6	83.6	69.1	23.6	**74.98%**
His15_25	85.6	74.2	87.9	71.9	87.3	90.8	76.2	80.9	57.4	84.7	84	69.4	24.1	74.95%

3.1.6 Análisis de resultados

De los resultados anteriores, es de interés que para todos los casos se logra superar el rendimiento de las características KDE-RGB, 65.79%, aunque algunas de los enfoques utilizados no logran superar el *baseline* de 67.19%.

El enfoque de generar histogramas entrega los mejores resultados, alcanzado un 74% en todas las capas probadas (muy por encima del *baseline*), dependiendo de los límites del histograma establecidos. Con respecto al histograma, se observa también que recortar su intervalo entrega un mejor rendimiento, en comparación a utilizarlo en su rango completo. Lo anterior se cumple para todas las capas.

El rendimiento del clasificador, al comparar el rendimiento del mejor caso para cada una de las capas, varía entre un 74.11% y un 74.98%. A su vez, la Tabla 3-9 muestra que dentro de los largos de histogramas probados, con un largo de 10 por cada *feature-map*, se obtiene el mejor rendimiento. El mejor caso se obtiene con un histograma de largo 10 por cada *feature-map* y con una reducción de los límites del histograma de un 25%, ver Tabla 3-9. Estas características, las correspondientes al mejor caso, son las usadas a lo largo de este trabajo (en lo referente a característica CNN extraídas para segmentación semántica).

3.2 Sub-sistema de clasificación de lugares y generación de contexto

Del estado del arte se puede concluir que la mejor alternativa actual para clasificar imágenes corresponde a las redes *deep* tipo CNN. En casos en que las bases de datos son muy pequeñas y no es posible entrenar una red deep, se ha demostrado el éxito del uso de características extraídas de redes previamente entrenadas. Este último corresponde al caso de la base de datos NYUv1, que solo posee 2284 imágenes. Por este motivo se utiliza un sistema de clasificación de lugares en base a este tipo de características. A su vez se prueba integrar la información del clasificador de lugares al sistema de segmentación semántica, para mejorar los resultados de la segmentación Esto último corresponde a la generación de información de contexto. En la Figura 3-8 se muestran los bloques que corresponden al sistema de clasificador de lugares y generador de contexto, dentro del diagrama general del sistema.

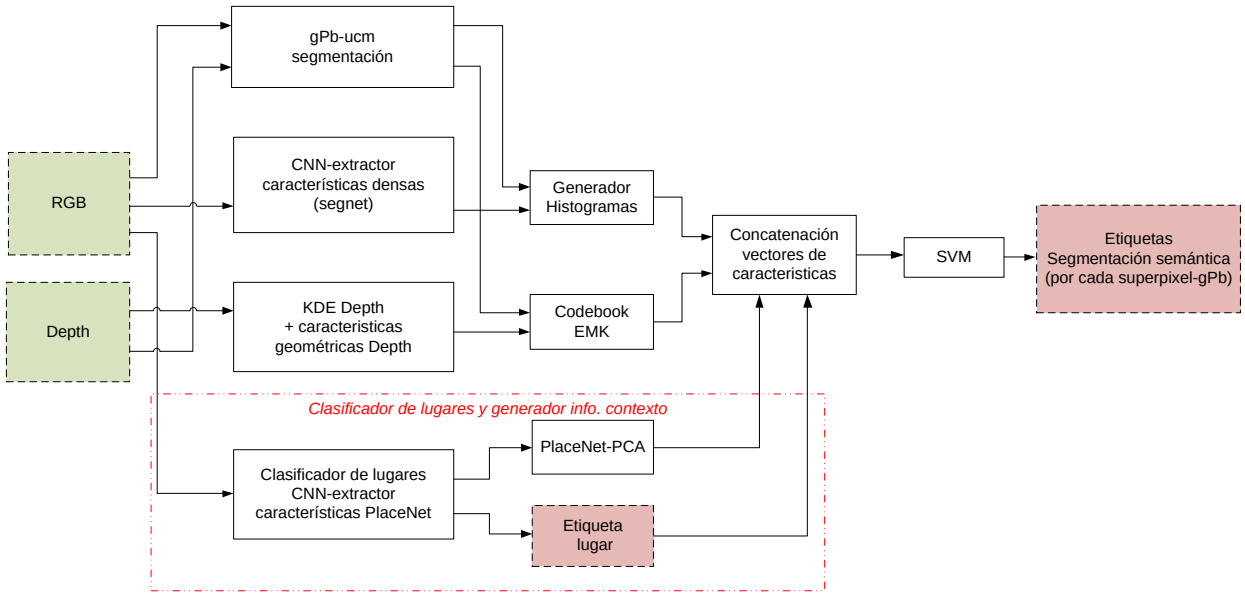


Figura 3-8 : Esquema general del sistema, señalando las etapas correspondientes al clasificador de lugares y generación de contexto.

La Figura 3-9 muestra en mayor detalle los módulos que se utilizan para generar la información de contexto y del clasificador de lugares. Se puede ver que una parte de todas las etapas es una red CNN previamente entrenada, para clasificar imágenes. Ambos módulos, el de clasificación de lugares y el de generación de contexto, son detallados en este las secciones 3.2.1 y 3.2.2, respectivamente.

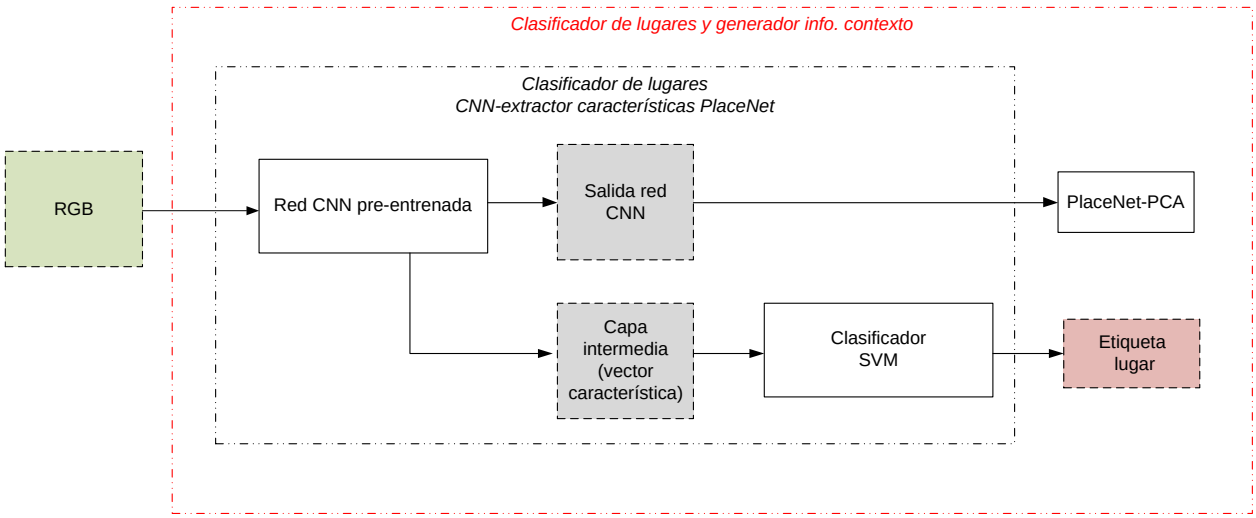


Figura 3-9 : Detalle sistema clasificador de lugares y generación de contexto.

3.2.1 Clasificador de lugares

El sistema para clasificar lugares es el mismo usado en varios trabajos para clasificar imágenes [6][4][36] y que ha demostrado entregar muy buenos resultados. Este está compuesto por una red pre-entrenada desde donde se extraen las características y un clasificador SVM. Para el problema de clasificación de imágenes, existen varias arquitecturas de redes disponibles y más de una base de datos de entrenamiento aplicable para este caso.

La Figura 3-10 muestra una esquema de esta etapa.

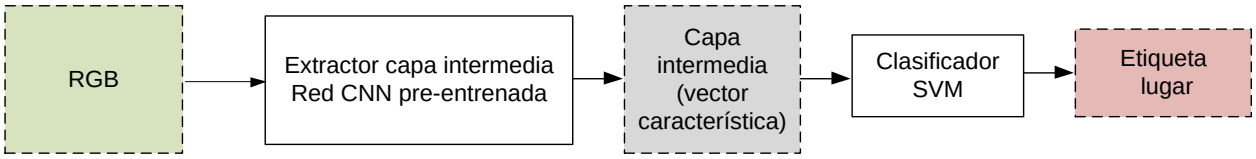


Figura 3-10 : Esquema clasificador de lugares.

La parte más importante de este esquema de clasificación son las características que se extraen. Existen 3 variables principales al momento de extraerlas:

- *Arquitectura de la red:* Corresponde al diseño de la red pre-entrenada desde donde se extraen las características. La arquitectura de la red determina las dimensiones y cantidad de capas en la red, lo que define las opciones al momento de extraer las características. A su vez, la arquitectura está directamente relacionada con el rendimiento de la red en el problema original donde fue entrenada y el rendimiento de sus características extraídas.

- *Conjunto de entrenamiento de la red*: Esta variable corresponde al conjunto donde es entrenada la red, desde donde luego se van a extraer las características. El más común para el caso de clasificación de imágenes es *Imagenet*, pues es la base de datos etiquetada de imágenes más grande existente.

- *Capa de extracción de características*: Este punto corresponde específicamente a la capa desde donde se van a extraer las características y de qué forma. Lo más común es extraer un vector de características desde alguna de las ultimas capas *fully-conected*, aunque existen trabajos que buscan formas de extraerlas desde las capas convolucionales.

Para este trabajo en particular, se prueban 3 arquitecturas CNN que son bastantes populares en la comunidad de *deep* learning. Estas corresponden a Alexnet, a VGG16 y a GoogLeNet. A su vez, cada una de estas arquitecturas es pre-entrenada en dos bases de datos distintas. Una de ellas es Imagenet, que posee 14 millones de ejemplos y 1000 categorías distintas. Otra es Place205, una base de datos compuesta solo por imágenes de lugares, tanto interiores como exteriores, con 2.5 millones de ejemplos y 205 categorías solo de lugares. Se espera contrastar las características genéricas obtenidas a partir de Imagenet versus las especializadas que genera la misma red pero entrenada para reconocer lugares en Place205. Finalmente, se extraen los vectores de características de forma directa desde las capas *fully-conected* y la capa *soft-max* de cada red. La capa *soft-max*, la última capa de cada red, corresponde propiamente al resultado de la clasificación en la base de datos donde se entrenó (vector con las clases). Sin embargo, se puede interpretar como una etapa más de codificación de la red CNN, al momento de ser transferida como características para otro problema. Para el caso particular de GoogLeNet se aprovecha de

extraer características de la capa *mean-pooling*, que también es vectorial. Desde la Tabla 3-10 hasta la Tabla 3-12, se muestra la estructura de las últimas capas de Alexnet, VGG16 y GoogLeNet, respectivamente. En negrita se destacan las dimensiones de las capas que son vectoriales, y que pueden utilizarse de forma directa para extraer un vector de características.

Tabla 3-10 Estructura de las últimas capas de Alexnet

Capa	Tipo de capa	Dimensión	Abreviación
Capa 5	conv_3_256_relu	[13,13,256]	Conv5
Capa 5	Max pooling	[6,6,256]	Pool5
Capa 6	fully connected	**[4096,1]**	FC6
Capa 7	fully connected	**[4096,1]**	FC7
Capa 8	fully connected	**[n°clases,1]**	FC8
Salida	Soft-max prob.	**[n°clases,1]**	Prob

Tabla 3-11 Estructura de las últimas capas de VGG16

Capa	Tipo de capa	Dimensión	Abreviación
Capa 5	conv_1_512-relu	[14,14,512]	Conv5
Capa 5	Max pooling	[7,7,512]	Pool5
Capa 6	fully connected	**[4096,1]**	FC6
Capa 7	fully connected	**[4096,1]**	FC7
Capa 8	fully connected	**[n°clases,1]**	FC8
Salida	Soft-max prob.	**[n°clases,1]**	Prob

Tabla 3-12 Estructura de las últimas capas de GoogleNet

Capa	Tipo de capa	Dimensión	Abreviación
Capa 5	inception 5ª	[7,7,832]	Ince5a
Capa 5	inception 5b	[7,7,1024]	Ince5b
Capa 5	Average pooling	**[1024,1]**	Pool5
Capa 6	Fully conected	**[n°clases,1]**	FC6
Salida	Soft-max prob.	**[n°clases,1]**	Prob

Se puede ver que tanto Alexnet (Tabla 3-10) como VGG16 (Tabla 3-11) tienen la misma estructura en las ultimas 4 capas, que corresponden a las capas que pueden ser utilizadas como posibles vectores de características. Estas capas corresponden a tres *fully-conected* y una *soft-max*. Las dos primeras capas *fully-conected* de ambas redes corresponden a un vector de dimensión 4096 (FC6 y FC7) respectivamente. La tercera capa *fully-conected* (FC8) y la de salida (*soft-max*) corresponden a un vector de dimensión igual al número de clases de la base de datos donde fueron entrenadas. Para el caso de las redes entrenadas en Imagenet, la dimensión de estos vectores es de 1000, mientras que para las entrenadas en Place205, esta dimensión es 205. La arquitectura de GoogLeNet es bastante diferente a la de AlexNet y VGG16. GooLeNet posee solamente 3 capas vectoriales, correspondientes a las 3 últimas. La primera capa es una *average-pooling*, un vector de dimensión 1024, la siguiente es una fully-conected de dimensión igual al número de clases y la última corresponde una *soft-max*, de la misma dimensión que la anterior. Esto deja como alternativas para la extracción de características las últimas 4 capas de Alexnet y VGG16 y las 3 últimas de GoogLeNet. A su vez, para cada una de estas opciones, existe la

alternativa de pre-entrenar la red en Imagenet o en Place205. Se prueban extrayendo características para cada una de estas alternativas.

3.2.2 Clasificación

Para el clasificador se usan SVM multiclase *one-vs-all* de la librería Libsvm 3.21. Los conjuntos de entrenamiento y pruebas se separan de forma aleatoria, pero por habitaciones, de forma que las imágenes de una misma habitación estén todas en el conjunto de prueba o en el conjunto de entrenamiento (la clase café es omitida, pues solo existe una habitación, ver Tabla 2-9,). Se separan los conjuntos de entrenamiento y prueba, en proporción 70% y 30%, respectivamente. Las métricas de rendimiento son la diagonal principal de la matriz de confusión, en forma de porcentaje y la media de esta diagonal principal.

La forma en que se crean los conjuntos para este caso es distinta a como se generaron para el problema de segmentación semántica. En el caso de segmentación semántica, se separaban los conjuntos de entrenamiento y prueba, de forma aleatoria, a nivel de imágenes (pudiendo quedar imágenes distintas, pero de una misma pieza, en el conjunto de entrenamiento y prueba a la vez). Para el problema de clasificación de lugares, se separan los conjuntos de imágenes a nivel de habitaciones.

Los objetivos de generar los conjuntos por grupos de piezas son los siguientes. Primero, dado que el problema de clasificar imágenes es más simple que el de segmentación semántica, la idea es probarlo bajo una situación más desafiante. A su vez, esta prueba entregaría un valor cercano del rendimiento de este sistema bajo condiciones más reales. Al momento de ocupar esta información para generar el contexto de lugar, el clasificador es reentrenado bajo los conjuntos definidos para el problema de segmentación semántica.

3.2.3 Resultados

En esta sección se prueban todas las opciones posibles para extraer las características. Las arquitecturas a probar son las 3 siguientes:

- Alexnet (4 capas como opciones donde extraer características).
- VGG16 (4 capas como opciones donde extraer características).
- GoogLeNet (3 capas como opciones donde extraer características).

Además para cada arquitectura, se prueba con una red entrenada en Imagenet y otra versión de la red entrenada en Place205. Esta da un total de 22 alternativas desde donde extraer las características. A continuación, las tablas 3-13 a 3-17 muestran los resultados obtenidos.

Alexnet

Tabla 3-13. Resultados pruebas clasificador de lugares NYUv1, Alexnet pre-entrenada en Imagenet

Alexnet-Imagenet	**Cocina**	**Oficina**	**Baño**	**Living**	**Pieza**	**Librería**	**Media**
Capa Prob	61.8%	55.4%	67.6%	43.9%	70.7%	56.6%	58.27%
Capa FC6	77.5%	67.3%	73.5%	51.0%	67.2%	86.9%	73.14%
Capa FC7	76.4%	**73.3%**	**85.3%**	50.0%	**69.0%**	**89.6%**	**75.57%**
Capa FC8	**78.7%**	72.3%	64.7%	**57.1%**	68.1%	84.6%	73.90%

Tabla 3-14. Resultados pruebas clasificador de lugares NYUv1, Alexnet pre-entrenada en Place205

Alexnet-Place205	**Cocina**	**Oficina**	**Baño**	**Living**	**Pieza**	**Librería**	**Media**
Capa Prob	76.4%	58.4%	91.2%	41.8%	72.4%	89.1%	72.84%
Capa FC6	78.7%	77.2%	88.2%	36.7%	63.8%	92.8%	74.81%
Capa FC7	**84.3%**	**80.2%**	**91.2%**	46.9%	65.5%	**91.4%**	77.54%
Capa FC8	83.1%	**80.2%**	82.4%	**49.0%**	**75.9%**	**91.4%**	**79.06%**

VGG16

Tabla 3-15. Resultados pruebas clasificador de lugares NYUv1, VGG16 pre-entrenada en Imagenet

VGG16-Imagenet	**Cocina**	**Oficina**	**Baño**	**Living**	**Pieza**	**Librería**	**Media**
Capa Prob	88.8%	53.5%	82.4%	55.1%	80.2%	70.6%	70.41%
Capa FC6	83.1%	76.2%	44.1%	44.9%	65.5%	90.5%	73.75%
Capa FC7	82.0%	80.2%	64.7%	46.9%	64.7%	91.4%	75.72%
Capa FC8	**86.5%**	**81.2%**	**88.2%**	**51.0%**	**74.1%**	**93.2%**	**80.58%**

Tabla 3-16. Resultados pruebas clasificador de lugares NYUv1, VGG16 pre-entrenada en Place205

VGG16-Place205	Cocina	Oficina	Baño	Living	Pieza	Librería	Media
Capa Prob	86.5%	82.2%	94.1%	42.9%	79.3%	88.7%	79.21%
Capa FC6	83.1%	88.1%	73.5%	55.1%	78.4%	94.1%	82.09%
Capa FC7	88.8%	**89.1%**	58.8%	**52.0%**	**81.0%**	**98.6%**	**83.76%**
Capa FC8	**89.9%**	85.1%	**97.1%**	48.0%	**81.0%**	89.6%	81.64%

GoogLeNet

Tabla 3-17. Resultados pruebas clasificador de lugares NYUv1, GoogLeNet pre-entrenada en Imagenet

GoogLeNet-Imagenet	Cocina	Oficina	Baño	Living	Pieza	Librería	Total
Capa Prob	77.5%	58.4%	76.5%	38.8%	68.1%	51.1%	58.27%
Capa FC6	**71.9%**	**88.1%**	**88.2%**	**51.0%**	79.3%	88.2%	**78.91%**
Capa Avr-Pool	**71.9%**	81.2%	**88.2%**	50.0%	**81.9%**	**90.5%**	**78.91%**

Tabla 3-18. Resultados pruebas clasificador de lugares NYUv1, GoogLeNe pre-entrenada en Place205

GoogLeNet-Place205	Cocina	Oficina	Baño	Living	Pieza	Librería	Media
Capa Prob	92.1%	77.2%	97.1%	46.9%	77.6%	86.9%	79.06%
Capa FC6	77.5%	83.2%	**70.6%**	**57.1%**	81.0%	**95.5%**	81.64%
Capa Avr-Pool	**82.0%**	**85.1%**	**70.6%**	**57.1%**	**81.9%**	**95.5%**	**82.70%**

En general, se observa un rendimiento alrededor de un 80% para la mayoría de los mejores casos por tabla. El mejor resultado obtenido es al utilizar la red VGG 16 entrenada en Place205, extrayendo características desde la capa FC7. Esta es la forma en que se extraen las características para el clasificar de lugares finalmente utilizado.

3.2.4 Análisis de resultados

En esta sección se analizan los resultados anteriores. Se añaden las Tabla 3-19 y Tabla 3-20 a modo de resumen, para apoyar los análisis. La Tabla 3-19 muestra el resultado más alto obtenido en cada arquitectura de red y base de datos de entrenamientos probadas. El resultado más alto se obtiene siempre en las características de la capa más alejada de la salida de la red o en la posterior a esta (ver tablas sección 3.2.3). Por su parte, la Tabla 3-20 muestra el resultado más bajo obtenido en cada arquitectura de red y base de datos de entrenamientos probadas. El resultado más bajo se obtiene siempre al utilizar la salida de la red como vector de características.

Tabla 3-19 Resultado más alto obtenido en cada prueba de arquitectura y base de datos de entrenamiento

Resultado más alto de cada red	Media
Alexnet Imagenet	75.57%
Alexnet Place205	79.06%
VGG16 Imagenet	80.58%
VGG16 Place205	**83.76%**
GoogleNet Imagenet	78.91%
GoogleNet Place205	82.70%

Tabla 3-20 Resultado más bajo obtenido en cada prueba de arquitectura y base de datos de entrenamiento

Resultado más bajo de cada red	Media
Alexnet Imagenet	58.27%
Alexnet Place205	72.84%
VGG16 Imagenet	70.41%
VGG16 Place205	79.21%
GoogleNet Imagenet	58.27%
GoogleNet Place205	79.06%

De las tablas Tabla 3-19 y Tabla 3-20 se desprende que, para todas las arquitecturas de redes probadas, la versión entrenada en la base de datos Place205 entrega un rendimiento mayor a la misma red entrenada en Imagenet. Por su parte, esta diferencia es más alta al comparar los peores resultados (Tabla 3-20), donde es alrededor de un 10%, que al comparar los mejores resultados (Tabla 3-19), donde es alrededor de un 4%. Otra tendencia observada es el rendimiento decreciente desde la capa más alejada de la salida de la red, hasta llegar a la salida propiamente tal (ver tablas sección 3.2.3).

De los resultados observados se puede inferir lo siguiente:

- Mientras más similar sea el problema donde fue entrenada la red a extraer características al problema donde se buscan trasferir, mayor es su rendimiento. Esta ventaja es mayor al comprar en las capas más cercanas a la salida, aunque en las capas más cercanas al interior de la red es menos significativa.

- Con respecto a la diferencias de rendimiento por capa (manteniendo todo el resto constante), se puede atribuir el mayor rendimiento al hecho de que las capas más alejadas de la salida de la red, son más genéricas y pueden generalizar mejor. Por su parte las más cercanas a la salida, hasta llegar a la salida propiamente tal, son más específicas al problema donde fue entrenada la red CNN. Lo anterior también explicaría por qué las diferencias son más grandes al utilizar la capa de salida como característica que utilizar las capas de más al interior. Dado que las características de capas más interiores son más genéricas, la diferencia entre usar la red entrenada en Imagenet o Place205 es menor. Por su parte, dado que las características más cercanas a la salida son más específicas al problema donde fueron entrenadas, la diferencia, en ventaja a favor de las entrenadas en place205, es mayor.

3.2.5 Uso de contexto de lugar en segmentación semántica

Una idea poco explorada en los trabajos actuales es la utilización del contexto de lugar para mejorar los resultados de la segmentación semántica. De forma intuitiva, si se sabe a priori que la imagen a segmentar corresponde a un lugar determinado, se puede usar estar información para tener una idea de la probabilidad de encontrar ciertos objetos. Por ejemplo, si se sabe que la imagen a segmentar corresponde a un dormitorio, la probabilidad de encontrar una cama es mucho más alta que la probabilidad de encontrar un sofá.

El enfoque que se utiliza es bastante simple. Se añade la información de lugar al vector de características utilizado en la etapa de segmentación semántica. La información de lugar se obtiene del clasificador de lugares previamente entrenado. Esta información de contexto queda representada por un vector de características de largo igual a la cantidad de clases de lugares, donde cada característica representa la probabilidad de encontrarse en un tipo determinado lugar. Este vector es de dimensión 7 y se obtiene luego de aplicar el clasificador de lugares a la imagen de entrada.

Es necesario recordar, como se mencionó en la sección 3.2.2, que el clasificador de lugares debe ser reentrenado según el conjunto de entrenamiento utilizado en el problema de segmentación semántica (para ser agregado como información a este problema). En caso de no hacer esto, podrían terminar evaluándose imágenes del conjunto de entrenamiento en el de test (lo que es metodológicamente incorrecto). Al reentrenarse el clasificador de lugares según este nuevo conjunto, el rendimiento global aumenta 98.7%, bastante más que el mejor caso anterior, correspondiente solo a 83.8%. Esto se debe a que en este caso el clasificador debe identificar lugares entre habitaciones que ya ha visto, lo que es un problema más fácil que identificar lugares en habitaciones que está viendo por primera vez.

La segunda forma de generar contexto es a partir de la información de la misma red entrenada en PlaceNet. Esta información de contexto, a diferencia de la anterior, es generada de forma no supervisada, pues no se realiza ningún tipo de entrenamiento en la base datos NYU Depth v1 para obtenerla. El objetivo de este segundo tipo de información de contexto es generar información más variada. La información de lugar solo tiene 7 tipos de clases distintas consideradas, sin embargo, podrían existir más clases o sub-clases codificadas. Por ejemplo, para la clase living, podrían existir distintos tipos de living, dependiendo del diseño, estilo, o incluso del mismo encuadre de la imagen. Por este motivo se aprovecha el vector de salida de Place205, que posee 205 clases de lugares. De estas 205 clases, 150 se activan con el conjunto de imágenes de NYU Depth v1. Esto es útil para generar la información de lugar más específica.

El vector de características de cada superpixel es de dimensión igual al largo vector de características RGB + largo vector de características Depth + largo vector de contexto clase de lugar, lo que es igual a 640+400+7 = 1047. Dado que esta dimensión ya es alta, y Place205 entrega un vector de salida de tamaño 205, se utiliza PCA como método para reducir la dimensionalidad del vector de Place205. Se prueba reduciendo la dimensionalidad hasta el mínimo valor antes de comenzar a perder rendimiento en el problema de segmentación semántica, que corresponde a utilizar 20 componentes principales. Esto genera un vector de características de Place205 de dimensión 20, lo que sumada a las 7 características de lugar, genera un vector de dimensión 27 para las características de contexto.

La Figura 3-11 muestra la segunda etapa donde se genera el contexto de lugar. Se puede ver en el diagrama la información correspondiente a la salida de Place205 y como luego es reducida por el bloque “Place205-PCA”.

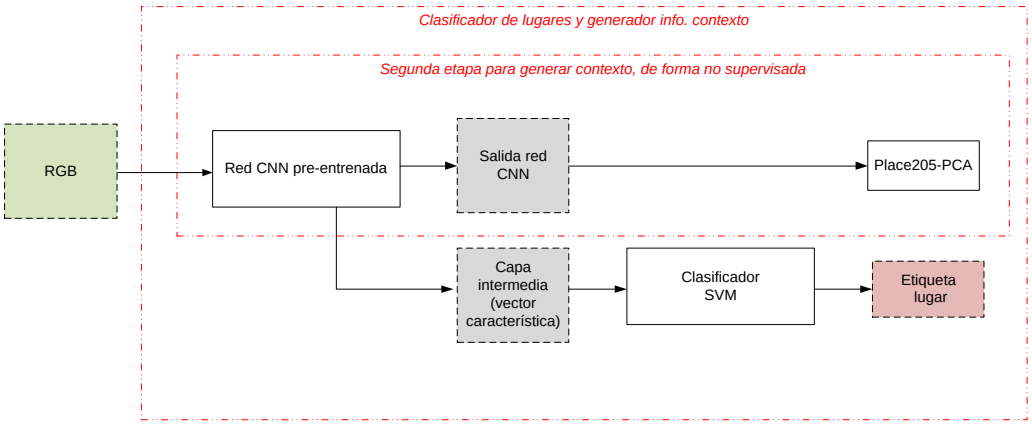


Figura 3-11 : Esquema etapa de clasificación de lugar y generación de contexto.

4. Pruebas, resultados y análisis

4.1 Metodología para pruebas

Para verificar las hipótesis planteadas en el primer capítulo de este documento, así como también para cuantificar las variaciones en rendimiento, es necesario realizar las pruebas correspondientes. Esto significa contrastar las distintas alternativas y mejoras propuestas versus algunos de los trabajos o métodos existentes.

Para evaluar el rendimiento en el problema de segmentación semántica, se generan los conjuntos de entrenamiento y test de la misma forma explicada en la sección 3.1.4, así como también sus mismas métricas de rendimiento. Esto corresponde al rendimiento a nivel de pixeles, considerando cada pixel una muestra a evaluar. Se muestra la diagonal principal de la matriz de confusión, en forma de porcentaje y la media de la diagonal principal como indicador global del rendimiento.

Para contrastar los resultados, se entrena la red Segnet en NYU Depth v1 de dos formas. La primera es entrenándola de forma normal, con el conjunto de entrenamiento de NYU Depth v1. Para esto se sigue el protocolo de entrenamiento sugerido en el repositorio oficial de Segnet. La segunda forma, es probar transferir el conocimiento de una red Segnet entrenada en SUN-RGBD a otra red Segnet, para ser aplicada en NYU Depth v1. Para esto se reentrena solo la capa de clasificación de Segnet, *softmax*, manteniendo el resto constante.

4.2 Resultados

En esta sección se muestran los resultados para los problemas de segmentación semántica y clasificación de lugares.

4.2.1 Segmentación semántica y uso de contexto

A continuación se muestran los resultados obtenidos en este trabajo de tesis en el área de segmentación semántica. Las características RGB para segmentación semántica son las que dieron mejor resultado en la sección 3.1.2. En la Tabla 4-1 se muestra el resultado de añadir el contexto solo con la clase de lugar (context 1) y el resultado al agregar además el contexto generado con las etiquetas de Place205 (context 1+2).

La Tabla 4-2 muestra el resultado de la implementación propuesta en este trabajo de tesis, y el resultado de cada una de sus distintas partes. Se incluye también para contrastar el mejor resultado hasta la fecha [43], y el resultado de sus distintas etapas. También se muestran las pruebas realizadas únicamente con la arquitectura de Segnet.

Tabla 4-1. Pruebas al añadir información de contexto sobre características RGB

	Cama	Persiana	Librero	Armario	Techo	Piso	Cuadro	Sofá	Mesa	TV	Pared	Ventana	Fondo	Media
Extrac CNN	86.7	74.6	88.2	70.5	88.0	90.8	76.4	80.8	57.7	84.6	83.6	69.1	23.6	74.98%
Extrac CNN + contex 1	91.0	73.4	87.7	71.9	87.7	89.5	80.8	81.8	63.4	84.2	82.0	65.3	25.5	75.71%
Extrac CNN + contex 1+2	92.5	73.9	89.0	72.2	88.2	90.9	79.0	82.4	62.0	85.3	83.2	67.1	26.8	**76.35%**

En la Tabla 4-1 puede apreciarse que el efecto de agregar el contexto de las dos etapas propuestas. Se observa que el contexto que más contribuye es el con la etiqueta de los lugares.

Tabla 4-2. Pruebas sistema completo, comparación etapas intermedias y otras implementaciones

	Cama	Persiana	Librero	Armario	Techo	Piso	Cuadro	Sofá	Mesa	TV	Pared	Ventana	Fondo	Media
KDE rgb	73.2	71.7	75.3	56.8	87.9	82.4	71.8	69.7	39.0	80.0	77.0	61.2	9.5	65.79%
KDE rgb-d	82.0	74.6	79.3	62.1	90.3	89.8	80.0	74.1	56.4	82.2	80.6	66.2	19.8	72.10%
KDE rgb-d+Multiesca.[43]	-	-	-	-	-	-	-	-	-	-	-	-	-	74.60%
KDE rgb-d+Multiesca.+CRF[43]	85	80	89	66	93	93	82	81	60	86	82	59	35	76.10%
Segnet	66.2	77.4	48.6	58.8	87.0	92.8	64.0	71.4	65.7	88.3	91.3	44.3	74.6	71.57%
Segnet Fine tuning	68.3	78.7	49.4	60.8	88.5	94.4	64.1	72.6	66.8	89.8	92.9	44.1	75.9	72.80%
Extrac CNN	86.7	74.6	88.2	70.5	88.0	90.8	76.4	80.8	57.7	84.6	83.6	69.1	23.6	74.98%
Extrac CNN + contex1+2	92.5	73.9	89.0	72.2	88.2	90.9	79.0	82.4	62.0	85.3	83.2	67.1	26.8	76.35%
Extrac CNN + depth+ context1+2	92.7	77.0	90.6	72.3	89.7	92.8	81.9	81.7	65.0	89.4	84.0	69.6	27.5	**78.00%**

Lo primero que se observa en la tabla 4.2 es que la implementación de este trabajo de tesis es la que entrega el mejor resultado. Esta tabla entrega bastante más información relevante, la cual es revisada en detalle en la sección 4.3

4.2.2 Clasificación de lugares

Con respecto a la clasificación de lugares, se compara el método utilizado en este trabajo de tesis vs los métodos que usan características *hand-crafted* clásicas. Los métodos basados en características *hand-crafted* utilizan SIFT, LBP y una variante de LBP para profundidad, llamada 3DLPB. Se prueban múltiples combinaciones de estas características y se muestran sus resultados más relevantes en la tabla 4-3.

Tabla 4-3. Pruebas distintos métodos para clasificar lugares

Comparación distintos métodos	**Media**
SIFT RGB-D	55%
LBP RGB(CENTRIST) + 3DLBP-D	64.42%
SIFT RGB+ 3DLBP-D	62.71%
VGG16 Placenet (capa3)	**83.76%**

En la Tabla 4-3, puede apreciarse cómo el enfoque basado en características extraídas CNN RGB supera ampliamente a las implementaciones basadas en características *hand-crafted*.

4.2.3 Resultados cualitativos

A continuación se muestran los resultados cualitativos de este trabajo, correspondientes a las imágenes de interés de los distintos resultados en segmentación semántica. Se comparan los mejores resultados obtenidos con cada método, basado en características *hand-crafted* (KDE-RGBD), red *deep* end-to-end (Segnet fine tuning) y el sistema propuesto en este trabajo (con todas sus etapas).

Los resultados mostrados con características *hand-crafted* corresponden a KDE-RGB y de profundidad, más las características geométricas. Los correspondientes a Segnet son del caso *fine-tuning* a partir de las características extraídas de SUN-RGBD. Para el sistema propuesto en este trabajo, corresponde al clasificador SVM con las características extraídas de Segnet SUN-RGBD, KDE de profundidad y características de contexto. A continuación pueden verse los resultados obtenidos, además de la imagen de entrada y del *ground truth*.

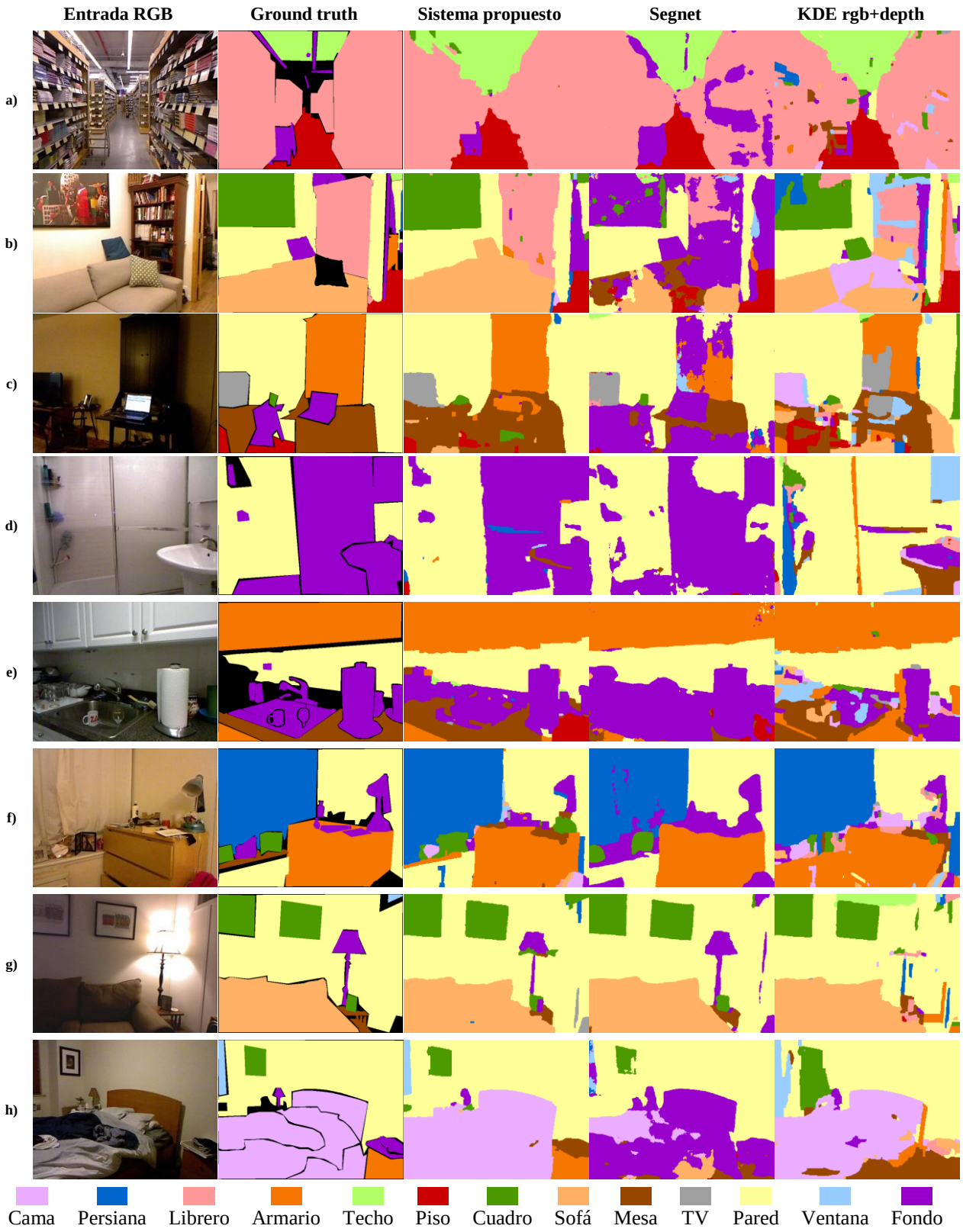


Figura 4-1 : Imágenes resultado segmentación semántica. Se muestra imagen de entrada, ground truth y el mejor resultado obtenido con cada uno de los enfoques.

En la Figura 4-1 se muestran algunos de los resultados obtenidos. La primera columna corresponde a la imagen de entrada en formato RGB, la segunda al *ground truth*, la tercera al resultado obtenido con el mejor sistema propuesto (SVM+feat CNN+KDE Depth+full contex.), la cuarta al mejor resultado obtenido con Segnet (Segnet Fine-tuning) y la última columna a KDE RGBD (sin CRF). En las imágenes de *ground truth* se ven sectores en negro, los que corresponde a las zonas no etiquetadas. Las zonas no etiquetadas no se utilizan ni para entrenar ni para evaluar el rendimiento, por eso solo están presentes en las imágenes del *ground truth*.

En la mayoría de los resultados de la Figura 4-1, se pueden ver casos donde el resultado obtenido con el método propuesto en este trabajo es cualitativamente mejor que los otros casos, tal como también se ve en la Tabla 4-2. En particular, para la Figura 4-1 a) b) c) y h) se observan como grandes zonas la imagen son confundidas con otras clases, tanto por Segnet como KDE. De igual forma se ve que el método propuesto es mucho más similar al ground truth, aunque existen zonas muy pequeñas donde se confunden las clases.

En las Figura 4-1 b) d) y e) se puede apreciar las ventajas de la etapa de segmentación gPb-ucm, que es la genera los superpixeles. Se puede ver como las fronteras de las clases quedan mejor definidas. Por ejemplo en la Figura 4-1 d) se puede ver como gran parte de la clase definida como fondo se confunde en los bordes con la clase pared. De igual forma en la Figura 4-1e) prácticamente toda la mesa es etiquetada como fondo por Segnet, debido a la complicada forma de los objetos. En esta misma imagen se puede ver que el método propuesto logra identificar la frontera entre la mesa y los objetos sobre ella.

Por su parte, el método basado en características KDE RGBD tiende a cometer errores muy notorios a la vista, como en las Figura 4-1 e) y c), donde confunde muchas etiquetas con clases muy dispares, o como en la Figura 4-1 g), donde no detecta los contornos de la lámpara (etiquetada como fondo). Cabe recordar que esta implementación de KDE no posee la parte de análisis multi-escala ni CRF, pues sus respectivas implementaciones no están disponibles. Si bien CRF podría corregir varios de estos errores, también es cierto que ningúno de los otros dos métodos que se están comparando usa CRF, por lo que estos también mejorarían sus resultados en caso de añadírseles esa etapa. En particular para el método propuesto por este trabajo, en algunas imágenes se observan pequeños superpixeles donde se comenten errores, como en el armario de la Figura 4-1 b). Este tipo de errores podría verse corregido en gran medida con el uso posterior de CRF sobre los resultados actuales.

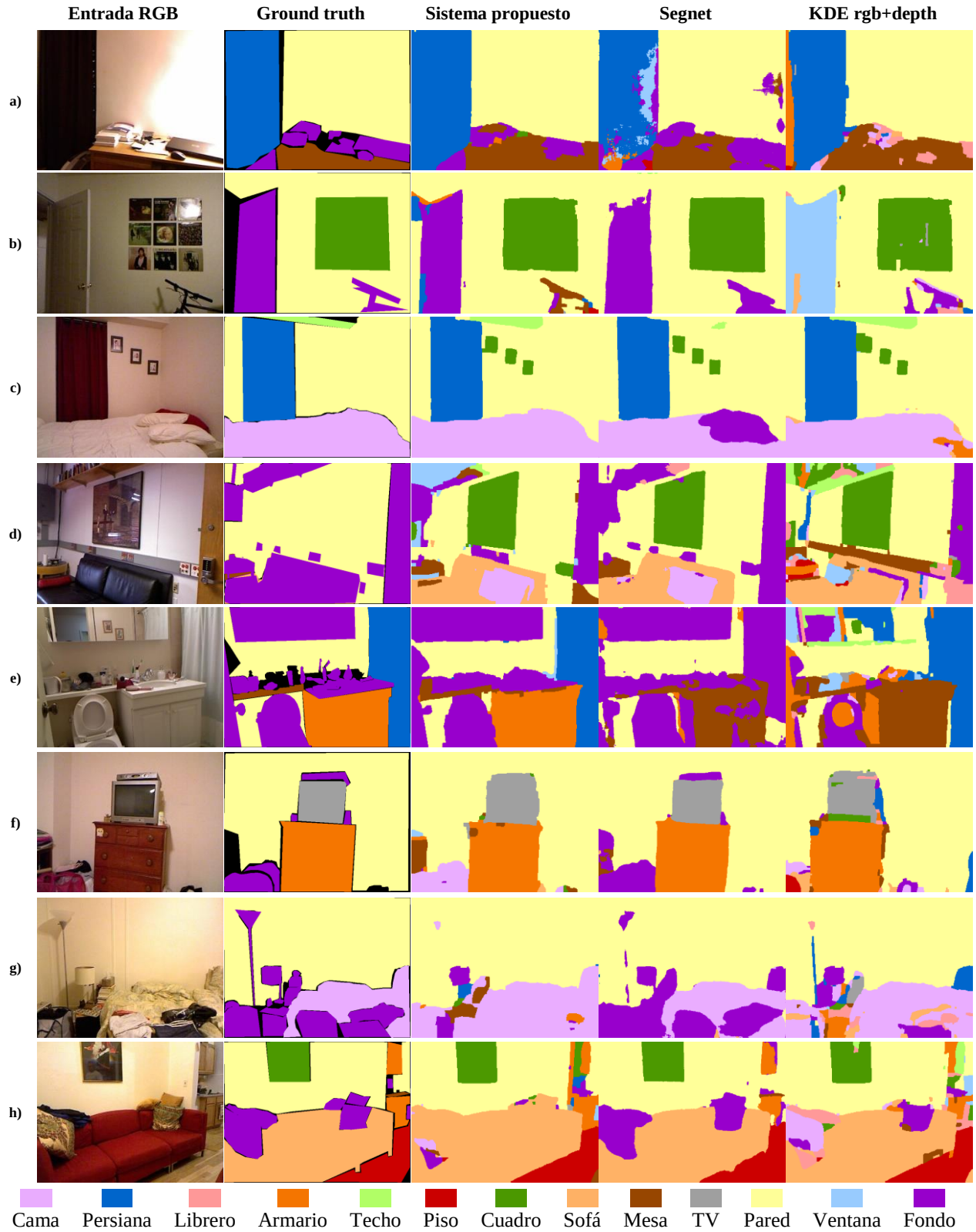


Figura 4-2: Imágenes resultado segmentación semántica. Se muestra imagen de entrada, ground truth y el mejor resultado obtenido con cada uno de los enfoques.

La Figura 4-2 muestra más imágenes de los resultados obtenidos. En particular las Figura 4-2 c) y d) son de interés, pues se ve que algunos métodos logran clasificar zonas que no están

correctamente etiquetadas en el *ground truth*. Por ejemplo en la Figura 4-2 c) puede verse que todos los métodos logran segmentar y clasificar de manera acertada los tres cuadros que están en la pared y que no están en *ground truth*. En ninguna de las imágenes de esta habitación que se encuentran en el conjunto de entrenamiento están etiquetados estos cuadros, lo que muestra en parte las capacidades de estos métodos. La Figura 4-2 d) es un caso más extremo, pues todo está etiquetado solo con dos clases, pared o fondo. Para este caso, todos los métodos son capaces de distinguir el cuadro del fondo y el sillón, con distintos grados en la calidad de los resultados, dependiendo del método. Como se comentaba en el Capítulo 2 de este documento, los errores de etiquetado existentes son uno de los problemas con esta base de datos. Son problemáticos no tan solo porque entorpecen el entrenamiento, sino también porque distorsionan los resultados de rendimiento. Por ejemplo, para estos casos, todos los pixeles que efectivamente corresponden a un cuadro o a un sillón, son considerados como incorrectos, pues no corresponde a la etiqueta, errónea, del *ground truth*. Si bien al comparar entre distintos métodos en esta misma base de datos, el problema afecta a todos ellos por igual, al buscar el rendimiento absoluto del método, este verá mermado su resultado.

En particular para las Figura 4-2 f) g) y h) pueden verse casos donde Segnet entrega un mejor resultado cualitativo que el método propuesto en este trabajo. Por un lado, en la Figura 4-2 f) puede verse que Segnet alcanza a segmentar objetos pequeños, como los dos objetos a los costados del televisor o el reproductor de video encima del televisor. Por otro lado, el método implementado en este trabajo fusiona estos objetos con el televisor. Este problema puede deberse a los superpixeles generados, que pueden fusionar segmentos muy pequeños con otros de mayor tamaño. La solución a este problema podría ser mejorar el método para generar los superpixeles o utilizar varias escalas de segmentación. En el método implementado solo se usa una escala fija de segmentación, aunque la idea de usar varias escalas es utilizada en el trabajo de los KDE[43], con buenos resultados

4.3 Análisis de resultados

En esta sección se comparan y analizan los resultados obtenidos, para cuantificar el aporte de cada una de las partes implementadas en este trabajo. Para esto se añade Tabla 4-4 a modo de resumen, con el rendimiento creciente de cada una de las distintas partes que componen el sistema y el rendimiento de otros sistemas o trabajos pertinentes en la comparación. Esta tabla también muestra el tipo de entrada que recibe cada método y las etapas adicionales que utiliza. Estos resultados también son representados de forma gráfica en la Figura 4-3 para poder apreciar de mejor forma las diferencias.

Tabla 4-4. Resumen resultados distintos métodos y distintos tipos de información y/o etapas utilizados

	Método	Info. deph	Detec. bordes	Análisis multi-escala	Contexto	CFR	media
1)	KDE-RGB		x				65.79%
2)	KDE-RGBD	x	x				71.40%
3)	Segnet entrenada NYUv1						71.57%
4)	Segnet finetuning						72.80%
5)	KDE-RGBD + multiescala [43]	x	x	x			74.60%
6)	Extrac. Feat.		x				74.98%
7)	KDE-RGBD + multiescala+CRF [43]	x	x	x		x	76.10%
8)	Extrac. Feat.+ contexto		x		x		76.35%
9)	Extrac. Feat. +KDE-D + contexto	x	x		x		**78.00%**

La Tabla 4-4 muestra un resumen de los resultados y métodos obtenidos en este trabajo y otros pertinentes en la comparación. Los métodos que comienzan con las siglas "KDE" se refieren a los trabajos que utilizan estas *características*, clasificando los superpixeles generados mediante gPb, con SVMs. Los que comienzan con "Segnet", como su nombre lo indica, son pruebas realizadas directamente con esta red. Los ítems que comienzan con "Extrac. Feat." se refieren al método propuesto en este trabajo. Este corresponde a la extracción de características densas desde la capa *Conv-relu* de Segnet, previamente entrenada en SUN-RGBD. Estas características luego son re-calculadas a nivel de superpixeles-gPb y clasificadas mediante SVMs. Con respecto a las columnas con los detalles, la primera se refiere a si usa o no la información de profundidad, la segunda a si usa información adicional de algún detector de bordes. La tercera columna, "Análisis multi-escala", indica si se utilizan varias escalas de la imagen de entrada en el análisis. La columna contexto, a si se utiliza información de contexto de lugar, planteada en este trabajo y la última, CRF, a si se utiliza un post-procesado mediante *conditional random fields*.

Lo primero que puede apreciarse en la Tabla 4-4 es que las dos últimas implementaciones de este trabajo 8) y 9), logran superar el mejor resultado actual obtenido en esta base de datos [43]. En particular la implementación 8) solo usa las *características* extraídas de Segnet más la información de contexto y con esto logra superar en un 0.25% al mejor resultado a la fecha [43]. Si bien el aumento de rendimiento es pequeño, es importante recalcar que se logra esto sin utilizar la información de profundidad, ni análisis en varias escalas, ni CRF, utilizados en el trabajo [43]. Al agregar la información de profundidad, con los KDE-Depth, se logra incrementar este margen a uno más significativo, de 1.65%. Con respecto a las pruebas con la red Segnet 3) y 4) estas lograron superar en ambos casos a los resultados obtenidos con KDE-RGBD 2). En este punto, es importante recalcar que Segnet solo usa como entrada la imagen RGB, a diferencia de 2) que también usa la información de la imagen de profundidad. En particular si se comparan Segnet 3) y KDE 1), solo usando la información RGB, observa una gran diferencia bastante amplia, de 5.78%, a favor de Segnet. Sin embargo, al utilizar KDE-RGBD más análisis multi-escala 5), ya se logra mejorar a Segnet. Por su parte, en el método propuesto en este trabajo, y usando como entrada solo la información RGB 6), se logra superar Segnet fine tuning 4) y a KDE-RGBD más análisis multi-escala 5).

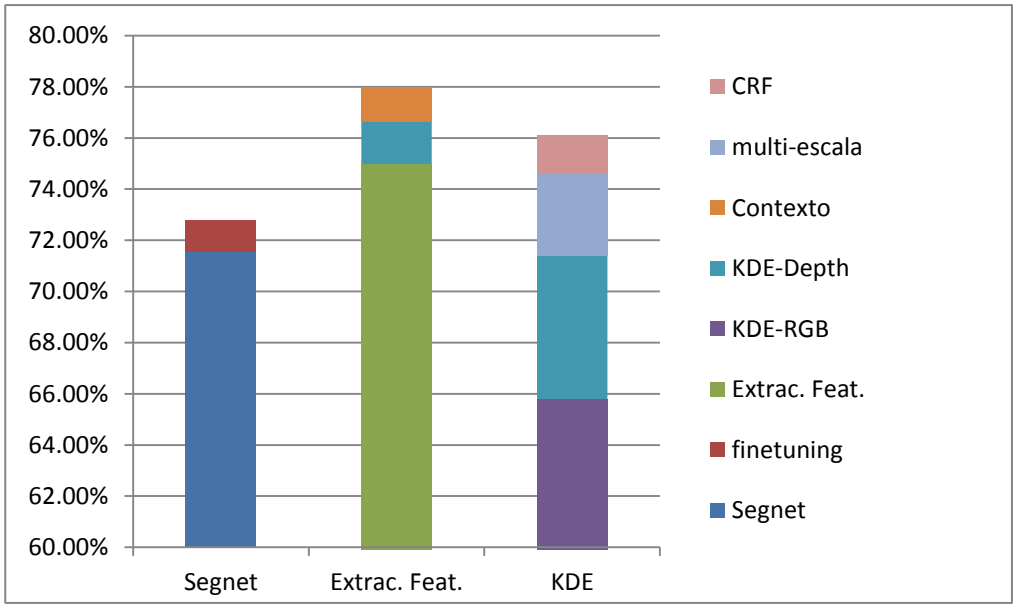


Figura 4-3 : Aporte de cada etapa a los distintos métodos probados.

Figura 4-3 muestra los mismos resultados ya comentados, pero en forma de grafico de barras, para facilitar su visualización. Por ejemplo, resulta sencillo apreciar que el fine-tuning, el añadir contexto de lugar, o utilizar CRF, son bastantes similares en cuanto a la magnitud de la mejora obtenida. También se puede visualizar que agregar las características KDE-Depth mejora en una medida bastante menor cuando se agregan al método propuesto en este trabajo, que cuando son agregadas al método basado solo en KDE-RGB.

5. Conclusiones

Se implementó un sistema de segmentación semántica, logrando superar el mejor resultado publicado hasta la fecha [43] en la base de datos NYU Depth v1. A su vez se implementó un sistema para clasificar lugares y se logró usar esta información para mejorar el resultado de la segmentación semántica. Se evaluó el rendimiento y aporte de los métodos propuestos en este trabajo.

Se comprobó que es posible extraer y transferir características densas de Segnet y utilizarlas de forma exitosa bajo el esquema clásico de superpixeles más clasificadores SVMs. Para esta base de datos en particular, estas características RGB logran desempeñarse 9.2% mejor que las características KDE RGB e incluso, un 2.1% mejor que el Segnet con fine-tunning. El motivo de esta mejora es, por un lado, el claro mayor poder de separabilidad de las características generadas mediante deep-learning, con Segnet, en comparación con las características *hand-crafted* KDE. Pero también se debe al enfoque superpixeles-gPb más clasificador SVM. La ventaja de este enfoque es que se añade información respecto a los contornos de los objetos y aparte es posible entrenar los SVM optimizando de forma global el problema a partir de toda la data, a diferencia del entrenamiento *batch* realizado por las redes deep.

Con respecto al reconocimiento de lugar, se verificó lo que indicaba el estado del arte, respecto a la eficacia de las características extraídas de las últimas capas de CNN. El desempeño de estas logrò superar a las características SIFT, LBP y 3D-LBP RGBD, siendo las características extraídas CNN solo RGB, a diferencias de las *hand-crafted* donde también se utilizó información de profundidad. Respecto al rendimiento de estas características, se observó de forma empírica que las extraídas desde capas más al interior de una red entregaban un mayor rendimiento que las extraídas desde capas más cercanas a la salida de la red. Lo anterior se constató en las tres arquitecturas de redes probadas, Alexnet, VGG16 y GoogLeNet. Se atribuye como explicación a este fenómeno que las características extraídas de capas más cercanas a la salida de la red son más específicas al problema donde la red fue entrenada, mientras que las más alejadas de la salida de la red, son más genéricas.

El uso de la información de contexto de lugar demostró ser efectiva. Gracias a esta información, se logró aumentar el rendimiento en 1.37%, comparable en magnitud a utilizar CRF, que aumenta en un 1,5% el rendimiento del trabajo basado en KDE [43]. El uso de información de contexto de lugar ha sido mínimamente explotado en los trabajos de segmentación semántica. El resultado obtenido en este trabajo, que utiliza un enfoque simple, da pie para el uso de enfoques más sofisticados.

Un punto importante al momento de comparar distintos trabajos de segmentación semántica, es tener en cuenta las entradas y etapas de las que están compuestos los métodos. Generalmente esta información no es añadida en las tablas de resumen y puede llevar a conclusiones erróneas. Por ejemplo, si se compara el rendimiento neto en NYU Depth v1 Segnet, 72.80% y el trabajo basado en KDE [43], 76.35%, se puede llegar a la conclusión que Segnet es un mal método. Pero no se tiene en cuenta que el trabajo [43] utiliza la información de la imagen de profundidad, una etapa de detección de bordes de objetos, varias escalas de análisis y CRF. Si se compara ambos en condiciones más parejas, solo usando como entrada la imagen RGB y sin las etapas de análisis multi-escala ni CRF, Segnet entrega un rendimiento 7.01% mayor. De igual forma la etapa RGB

del enfoque propuesto en este trabajo de tesis por sí sola no supera el trabajo[43], aunque al agregar las características de profundidad KDE y la información de contexto, sí. Lo anterior se logra sin utilizar análisis en varias escalas ni CRF, donde aún queda un margen por mejorar estos resultados.

6. Bibliografía

[1] A. Torralba and A. A. Efros, "Unbiased look at dataset bias," CVPR 2011, Colorado Springs, CO, USA, 2011, pp. 1521-1528

[2] A. Khosla, T. Zhou, T. Malisiewicz, A. A. Efros, and A. Torralba, "Undoing the damage of dataset bias," *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 7572 LNCS, no. PART 1, pp. 158–171, 2012.

[3] U. Neisser, "A Deeper Look at Dataset Bias," *Cogn. Psychol.*, vol. 1, pp. 46–85, 1967.

[4] A. S. Razavian, H. Azizpour, J. Sullivan, and S. Carlsson, "CNN features off-the-shelf: An astounding baseline for recognition," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, 2014, pp. 512–519.

[5] J. Yosinski, J. Clune, Y. Bengio, and H. Lipson, "How transferable are features in deep neural networks ?" Advances in Neural Information Processing Systems 27 (NIPS 2014) , pp. 3320-3328.

[6] J. Donahue, Y. Jia, O. Vinyals, J. Hoffman, N. Zhang, E. Tzeng, and T. Darrell, "DeCAF: A Deep Convolutional Activation Feature for Generic Visual Recognition," *Icml*, vol. 32, pp. 647–655, 2014.

[7] J. Long, E. Shelhamer, and T. Darrell, "Fully Convolutional Networks for Semantic Segmentation," *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, pp. 3431–3440, 2015.

[8] V. Badrinarayanan, A. Handa, and R. Cipolla, "SegNet: A Deep Convolutional Encoder-Decoder Architecture for Robust Semantic Pixel-Wise Labelling," *arXiv Prepr.*, p. 5, 2015.

[9] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, "Semantic Image Segmentation with Deep Convolutional Nets and Fully Connected CRFs," *Iclr*, pp. 1–14, 2014.

[10] V. Badrinarayanan, A. Kendall, and R. Cipolla, "SegNet: A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation," *Cvpr 2015*, p. 5, 2015.

[11] X. H. X. He, R. S. Zemel, and M. a. Carreira-Perpinan, "Multiscale conditional random fields for image labeling," *Proc. 2004 IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognition, 2004. CVPR 2004.*, vol. 2, 2004.

[12] J. Shotton, J. Winn, C. Rother, and A. Criminisi, "{TextonBoost} for Image Understanding: Multi-Class Object Recognition and Segmentation by Jointly Modeling Appearance, Shape and Context," vol. 81, no. 1, pp. 2–23, 2007.

[13] J. Shotton, M. Johnson, and R. Cipolla, "Semantic Texton Forest for Image Categorization and Segmentation," *Proc. Conf. Comput. Vis. Pattern Recognit.*, pp. 1–8, 2008.

[14] P. Kontschieder, S. R. Bulò, H. Bischof, and M. Pelillo, "Structured class-labels in random forests for semantic image labelling," *Proc. IEEE Int. Conf. Comput. Vis.*, pp. 2190–2197, 2011.

[15] J. Tighe and S. Lazebnik, "SuperParsing: Scalable Nonparametric Image Parsing with Superpixels\nComputer Vision – ECCV 2010," vol. 6315, pp. 352–365, 2010.

[16] A. Kendall, V. Badrinarayanan, and R. Cipolla, "Bayesian SegNet: model uncertainty in deep convolutional encoder-decoder architectures for scene understanding," *arXiv:1511.02680v1 [cs.CV]*, 2015.

[17] DD. Eigen and R. Fergus, "Predicting Depth, Surface Normals and Semantic Labels with a Common Multi-scale Convolutional Architecture," 2015 IEEE International Conference on Computer Vision (ICCV), Santiago, 2015, pp. 2650-2658

[18] C. Couprie, L. Najman, and Y. Lecun, "leargning hierchachical features for Scene Labeling (deep learning)," *Pattern Anal. Mach. Intell. IEEE Trans.*, vol. 35, no. 8, pp.

1915–1929, 2013.
[19] C. Couprie, "Indoor Semantic Segmentation using depth information (deep learning)," *Iclr*, pp. 1–8, 2013.
[20] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet Classification with Deep Convolutional Neural Networks," *Adv. Neural Inf. Process. Syst.*, pp. 1–9, 2012.
[21] P. Arbelaez, M. Maire, C. Fowlkes, J. Malik, P. Arbeláez, M. Maire, C. Fowlkes, and J. Malik, "Contour detection and hierarchical image segmentation.," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 33, no. 5, pp. 898–916, 2011.
[22] S. Gupta, R. Girshick, P. Arbeláez, and J. Malik, "Learning rich features from RGB-D images for object detection and segmentation," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2014, vol. 8695 LNCS, no. PART 7, pp. 345–360.
[23] H. Noh, S. Hong, and B. Han, "Learning Deconvolution Network for Semantic Segmentation" IEEE International Conference on Computer Vision (ICCV), 2015, pp. 1520-1528
[24] L.-C. Chen, G. Papandreou, F. Schroff, and H. Adam, "Rethinking Atrous Convolution for Semantic Image Segmentation," 2017.
[25] M. Ranzato, F. J. Huang, Y. L. Boureau, and Y. LeCun, "Unsupervised learning of invariant feature hierarchies with applications to object recognition," *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, 2007.
[26] M. D. Zeiler and R. Fergus, "Visualizing and Understanding Convolutional Networks arXiv:1311.2901v3 [cs.CV] 28 Nov 2013," *Comput. Vision–ECCV 2014*, vol. 8689, pp. 818–833, 2014.
[27] a. Pronobis, O. Martinez Mozos, B. Caputo, and P. Jensfelt, "Multi-modal Semantic Place Classification" *International Journal of Robotics Research* vol. 29, pp. 298–320, 2010.
[28] J. Wu, H. I. Christensen, and J. M. Rehg, "Visual place categorization: Problem, dataset, and algorithm," *2009 IEEE/RSJ Int. Conf. Intell. Robot. Syst. IROS 2009*, pp. 4763–4770, 2009.
[29] J. Xiao, J. Hays, K. A. Ehinger, A. Oliva, and A. Torralba, "SUN database: Large-scale scene recognition from abbey to zoo," *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, pp. 3485–3492, 2010.
[30] A. Bosch, A. Zisserman, and X. Muñoz, "Scene classification using a hybrid generative/discriminative approach.," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 30, no. 4, pp. 712–727, 2008.
[31] S. Lazebnik, C. Schmid, and J. Ponce, "Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories," *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, vol. 2, pp. 2169–2178, 2006.
[32] O. M. Mozos, H. Mizutani, R. Kurazume, and T. Hasegawa, "Categorization of indoor places using the Kinect sensor," *Sensors (Switzerland)*, vol. 12, no. 5, pp. 6695–6711, 2012.
[33] J. Wu and J. M. Rehg, "CENTRIST: A visual descriptor for scene categorization," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 33, no. 8, pp. 1489–1501, 2011.
[34] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel, "Backpropagation Applied to Handwritten Zip Code Recognition," *Neural Computation*, vol. 1, no. 4. pp. 541–551, 1989.
[35] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, vol. 07–12–June, pp. 1–9, 2015.

[36] K. Simonyan and A. Zisserman, “Very Deep Convolutional Networks for Large-Scale Image Recognition,” arXiv:1409.1556, 2014
[37] B. Zhou, A. Khosla, A. Lapedriza, A. Torralba, and A. Oliva, “Places: An Image Database for Deep Scene Understanding,” pp. 1–12, 2016.
[38] F. Hu, G.-S. Xia, J. Hu, and L. Zhang, “Transferring Deep Convolutional Neural Networks for the Scene Classification of High-Resolution Remote Sensing Imagery,” *Remote Sens.*, vol. 7, no. 11, pp. 14680–14707, 2015.
[39] A. Wang, J. Lu, G. Wang, J. Cai, and T. Cham, “Multi-modal Unsupervised Feature Learning for RGB-D Scene Labeling,” ECCV 2014: Computer Vision – ECCV 2014 pp 453-467.
[40] N. Silberman and R. Fergus, “Indoor scene segmentation using a structured light sensor,” *2011 IEEE Int. Conf. Comput. Vis. Work. (ICCV Work.*, pp. 601–608, 2011.
[41] A. Hermans, G. Floros, and B. Leibe, “(ICRA)Dense 3D Semantic Mapping of Indoor Scenes from RGB-D Images,” pp. 2631–2638, 2014.
[42] D. Wolf, J. Prankl, and M. Vincze, “Fast semantic segmentation of 3D point clouds using a dense CRF with learned parameters,” *2015 IEEE Int. Conf. Robot. Autom.*, pp. 4867–4873, 2015.
[43] X. Ren, L. Bo and D. Fox, "RGB-(D) scene labeling: Features and algorithms," 2012 IEEE Conference on Computer Vision and Pattern Recognition, Providence, RI, 2012, pp. 2759-2766.
[44] C. Couprie, L. Najman, L. Najman, and Y. Lecun, “Toward Real-time Indoor Semantic Segmentation Using Depth Information,” *J. Mach. Learn. Res.*, vol. 1, pp. 1–48, 2014.
[45] S. H. Khan, M. Bennamoun, F. Sohel, and R. Togneri, “Geometry Driven Semantic Labeling of Indoor Scenes,” ECCV 2014: Computer Vision – ECCV 2014 pp 679-694.
[46] A. Handa, V. Patraucean, V. Badrinarayanan, S. Stent, and R. Cipolla, “SynthCam3D: Semantic Understanding With Synthetic Indoor Scenes,” *arXiv Prepr. arXiv1505.00171*, pp. 1–5, 2015.
[47] R. F. N Silberman, D Hoiem, P Kohli, “Indoor Segmentation and Support Inference from RGBD Images,” *Eur. Conf. Comput. Vis. - ECCV*, pp. 746–760, 2012.
[48] A. C. Muller and S. Behnke, “Learning depth-sensitive conditional random fields for semantic segmentation of RGB-D images,” *Proc. - IEEE Int. Conf. Robot. Autom.*, no. May, pp. 6232–6237, 2014.
[49] S. Gupta, P. Arbelaez, and J. Malik, “Perceptual organization and recognition of indoor scenes from RGB-D images,” *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, pp. 564–571, 2013.
[50] A. Paszke, A. Chaurasia, S. Kim, and E. Culurciello, “ENet: A Deep Neural Network Architecture for Real-Time Semantic Segmentation,” *arXiv*, pp. 1–10, 2016.
[51] L. Bo and C. Sminchisescu, “Efficient Match Kernels between Sets of Features for Visual Recognition,” *Comput. Complex.*, vol. 2, no. 1, pp. 1–9, 2009.
[52] L. Bo, X. Ren, and D. Fox, “Kernel descriptors for visual recognition,” *Advances in Neural Information Processing Systems 23 (NIPS 2010)* pp. 1–9, 2010.
[53] J. Yang, S. Member, Y. Tian, S. Member, and L. Duan, “Group-Sensitive Multiple Kernel Learning for Object Recognition,” vol. 21, no. 5, pp. 2838–2852, 2012.
[54] A. K. Campbell, J. Povey, K. J. Hancock, F. Mitrou, and M. Haynes, “Parents’ interest in their child’s education and children’s outcomes in adolescence and adulthood: Does gender matter?,” *Int. J. Educ. Res.*, vol. 85, pp. 131–147, 2017.
[55] S. Song, S. P. Lichtenberg, and J. Xiao, “SUN RGB-D: A RGB-D scene understanding benchmark suite,” *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, vol.

07–12–June, pp. 567–576, 2015.
[56] D. R. Martin, C. C. Fowlkes, and J. Malik, “Learning to Detect Natural Image Boundaries Using Local Brightness and Texture Cues,” *Pami*, vol. 26, no. 1, pp. 1–20, 2004.