UNIVERSIDAD SAN FRANCISCO DE QUITO USFQ

Colegio de Ciencias e Ingeniería

Redes Neuronales Multi-Canales de Derivadas Gaussianas para el Análisis de Multitudes.

Trabajo de Titulación

Hugo Israel Gavilima Pilataxi

Matemáticas

Trabajo de titulación presentado como requisito para la obtención del titulo de Matemático

Quito, 15 de Febrero del 2023

UNIVERSIDAD SAN FRANCISCO DE QUITO USFQ

Colegio de Ciencias e Ingeniería

HOJA DE CALIFICACIÓN DE TRABAJO DE TITULACIÓN

Redes Neuronales Multi-Canales de Derivadas Gaussianas para el Análisis de Multitudes.

Hugo Israel Gavilima Pilataxi

Nombre del profesor, Titulo académico: Julio Ibarra Fiallo, M.Sc.

Quito, 15 de Febrero de 2023

Derechos de Autor

Por medio del presente documento certifico que he leído todas las Políticas y Manuales de la Universidad San Francisco de Quito USFQ, incluyendo la Política de Propiedad Intelectual USFQ, y estoy de acuerdo con su contenido, por lo que los derechos de propiedad intelectual del presente trabajo quedan sujetos a lo dispuesto en esas Políticas.

Asimismo, autorizo a la USFQ para que realice la digitalización y publicación de este trabajo en el repositorio virtual, de conformidad a lo dispuesto en la Ley Orgánica de Educación Superior del Ecuador.

Nombres y apellidos:	Hugo Israel Gavilima Pilataxi
Código:	00138425
Cédula de Identidad:	1003993704
Lugar y fecha:	Quito, Febrero de 2023

ACLARACIÓN PARA LA PUBLICACIÓN

Nota: El presente trabajo, en su totalidad o cualquiera de sus partes, no debe ser considerado como una publicación, incluso a pesar de estar disponible sin restricciones a través de un repositorio institucional. Esta declaración se alinea con las prácticas y recomendaciones presentadas por el Committee on Publication Ethics COPE descritas por Barbour et al. (2017) Discussion document on best practice for issues around theses publishing, disponible en http://bit.ly/COPETheses

UNPUBLISHED DOCUMENT

Note: The following capstone project is available through Universidad San Francisco de Quito USFQ institutional repository. Nonetheless, this project – in wholeor in part – should not be considered a publication. This statement follows the recommendations presented by the Committee on Publication Ethics COPE de scribed by Barbour et al. (2017) Discussion document on best practice for issues around theses publishing available on http://bit.ly/COPETheses

Agradecimientos

Es mi deseo agradecer a Julio Ibarra, quien fue una gran mentoría durante la escritura de este trabajo, y quien despertó mi interés en la ciencia de datos. Así como a todos mis profesores durante la carrera, por su haberme formado profesionalmente.

Agradezco de todo corazón a mis padres Hugo y María, así como a mis hermanos Jorge y Cristina a quienes dedico esta investigación al haberme dado todo su apoyo durante mi trayectoria universitaria, en los buenos y malos momentos.

Quisiera agradecer a mis amigos: Daniel, André, Camila y Christian, quien han hecho estos años universitarios los mejores de mi vida. A Daniela T., Stephen M., Carlos V., Amanda I. y Nathy B., con quienes he compartido momentos sublimes que me ayudaron a crecer como persona.

Las palabras no bastan para agradecer todo su apoyo, siempre estaré eternamente agradecido con todos ustedes.

Por la diosa.

Resumen

En la presente investigación expone los resultados obtenidos al experimentar con una red neuronal de derivadas gaussiana para realizar análisis de multitudes para área urbana de la base de datos Shanghai Dataset. Los operadores gaussianos, basados en la Teoría Espacio-Escala, permiten procesar la información visual con mayor detalle, especialmente en conjuntos con diferentes escalas, problemas de oclusión o escenarios complejos, lo que resulta en candidatos perfectos para ser utilizados como estructura primitiva en una capa para una red neuronas de aprendizaje profundo, con lo cual la red reduciría significativamente el número de hiperparámetros en el modelo. En general, el modo propuesto logra métricas comparables a los modelos de alto nivel, utilizando solo aproximadamente el 10 % de los parámetros libres, lo que sugiere una posible solución o futura línea de investigación para el estudio de la congestión urbana. En este sentido, la red neuronal de derivadas de Gaussianas permite un procesamiento más eficiente de la información visual y reduce la cantidad de parámetros requeridos, lo que la convierte en una opción atractiva para el análisis de multitudes en áreas urbanas.

Abstract

This research exposes the results obtained by experimenting using a Gaussian derivative neural network to perform crowd analysis in urban area the ShanghaiDataset database. Gaussian operators, based in Scale-Space Theory, allows processing visual information in greater detail, especially in sets with different scales, occlusion problems, or complex scenarios, which results in perfect candidates to be used as a primitive structure in a layer in deep neural network to significantly reduce the number of hyperparameters in the model. Overall, the proposed mode achieves metrics comparable to high-level models, while using only approximately 10 % of the parameters, which suggests a possible solution or future line of research for the study of urban congestion. In this way, Gaussian derivative neural network allows for more efficient processing of visual information and reduces the number of parameters required, making it an attractive option for crowd analysis in urban areas.

Índice de figuras

Índice general

Capítulo 1

Introducción

Los recientes avances en el poder computacional de los ordenadores modernos han dado paso al procesamiento de gigantescos cálculos numéricos en menor tiempo [3]. Como resultado, el desarrollo de algoritmos que aprovechan esta ventaja ha crecido exponencialmente, así como su intervención en diversas áreas de estudio, como lo son las ciencias exactas o económicas [2]. El campo que alberga la teoría detrás estos algoritmos, se conoce como **Inteligencia Artificial**, y de esta se desprenden varios subcampos que se diferencian entre sí por el enfoque de su aprendizaje, o por la naturaleza de su dominio.

El *Aprendizaje Automático* es una rama de la inteligencia artificial, que abarca los distintos avances sobre el proceso de **aprendizaje** en los ordenadores [10]. En ciertos organismos vivos, como en los seres humanos, el conocimiento que puede abstraerse de un estimulo viene dado por una compleja conexión de células denominadas *neuronas*, que asocia esta señal de entrada con una respuesta de salida apropiada. Siendo así, las habilidades cognitivas y los mecanismos de aprendizaje orientados a responder `cómo` y `qué` aprendemos han sido de enorme interés tanto para biólogos como filósofos. Entonces, no es sorpresa que las *reglas* dentro del aprendizaje automático tomen inspiración en estos procesos biológicos [18].

Las **redes neuronales** son técnicas dentro del aprendizaje automático que emulan las conexiones neuronales de los seres vivos. De manera similar, estas se componen de unidades denominadas *perceptrón*. Las conexiones de los perceptrones son similares a las sinapsis dentro de un conjunto neuronal, donde en cada nodo existe una función que determina *el peso* que la información de entrada es relevante para la respuesta de salida. Los valores asociados a cada nodo son determinados mediante el *entrenamiento* de la red. En síntesis, una red neuronal puede ser vista como un grafo computacional que abstrae información a muy alto nivel, que junto al peso que tiene cada perceptrón en esta red determina el **error** que tendrá nuestra respuesta final [1]. Uno de los campos donde las redes neuronales son un pilar fundamental es la **Visión por Computadora**.

1.1. Contexto Visión Artificial

La Visión por artificial es un sub-campo de la IA que se enfoca en algoritmos del aprendizaje automático dedicados a abstraer información en un alto nivel de imágenes, o una secuencia de estas. Su campo de estudio se enfoca en permitir que las máquinas interpreten y comprendan la información visual del mundo de manera similar a la visión humana por medio de la extracción de característica por medio de el aprendizaje automático [10]. Los algoritmos dentro de este sub-campo, como árboles de decisión, support vector machine y redes neuronales, se pueden entrenar en un conjunto de imágenes etiquetadas para aprender a clasificar diferentes conjuntos en categorías. Estos algoritmos usan modelos matemáticos para analizar y reconocer patrones en los datos, lo que les permite hacer predicciones precisas sobre el contenido de una imagen. [18].

La teoría de **Espacio-Escala** plantea axiomas sobre `cómo` se procesa las imágenes

usando operadores afines para dominios espacio-temporales isotrópicos en términos de escalas ajustables, denominados **Kernels Gaussianos** [14]. Los avances en este campo han dado enormes resultados en la visión por computadora, ya que permite derivar estructuras receptivas en una escala determinada como base para expresar las operaciones en el procesamiento de imágenes. Al considerar las características en múltiples niveles de escala, los algoritmos de visión por computadora basados en la teoría del espacio de escala pueden capturar información más matizada y detallada sobre una imagen. Esto puede dar como resultado una mayor precisión en tareas como el reconocimiento y la clasificación de objetos. Tomando como ejemplo, [21] presenta una red neuronal que toma derivadas de kernels gaussianos como estructura primitiva para un problema de clasificación a distintas escalas, que podría ser aplicados en un dominio distinto para un rango de escala definido, como lo es el conteo de multitudes.

En las últimas décadas, el problema de conteo en multitudes ha sido de especial interés para el apropiado desarrollo urbano de las ciudades modernas, como la implementación de semáforos inteligentes, gestión de seguridad o control social [5]. Las soluciones propuestas desde el campo de la visión por computadora han sido constantes, sin embargo los problemas que enfrentan los investigadores aún no pueden ser del todo resueltas: variabilidad de escalas en los conjuntos de datos, complejos escenarios de fondo, una distribución **no** uniforme de los individuos, entre otros. [8]. Los resultados que puede traer una red neuronal enfocados a problemas en el mundo real están asociados a diversos factores, como la complejidad de sus conexiones, siendo así, no existe una única red neuronal, en cambio existen diferentes propuestas de redes que se ajustan exclusivamente a un problema específico en un dominio definido.

1.2. Redes Neuronales Artificiales en Deep Learning

Una red neuronal con una única capa de entrada y un solo nodo de salida se denomina *perceptrón*. La arquitectura de esta simple red neuronal se presenta en la Figura 1.1. Sea $(\bar{X}, \bar{y})$ un conjunto de entrenamiento, donde $\bar{X}$ es un vector de dimensión d, y $\bar{y}$ es el valor observado. Como se muestra en el gráfico, un perceptrón se compone de d-nodos de entrada, ponderados por un conjunto de pesos sinápticos en cada arista $W_i, i = \{1, 2, \ldots, d\}$ y un *umbral*. El nodo de salida se define como $\hat{y}$, que se expresa mediante:

$$\hat{y} = f(\bar{W} \cdot \bar{X} + b) \tag{1.1}$$

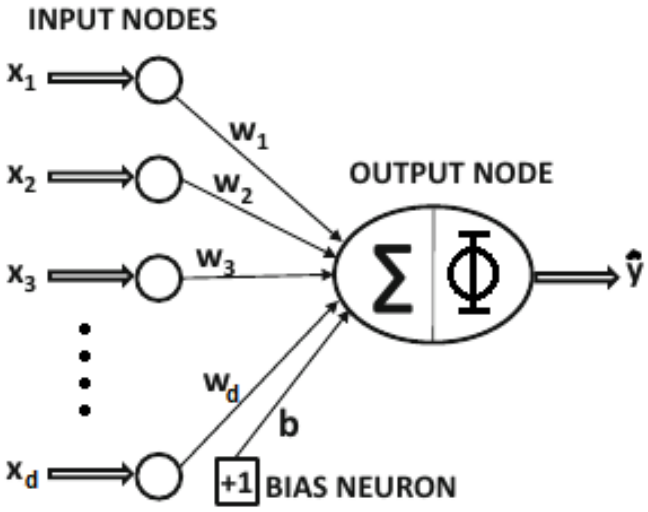


Figura 1.1: Arquitectura Red Neuronal de una Capa. Tomado de [1]

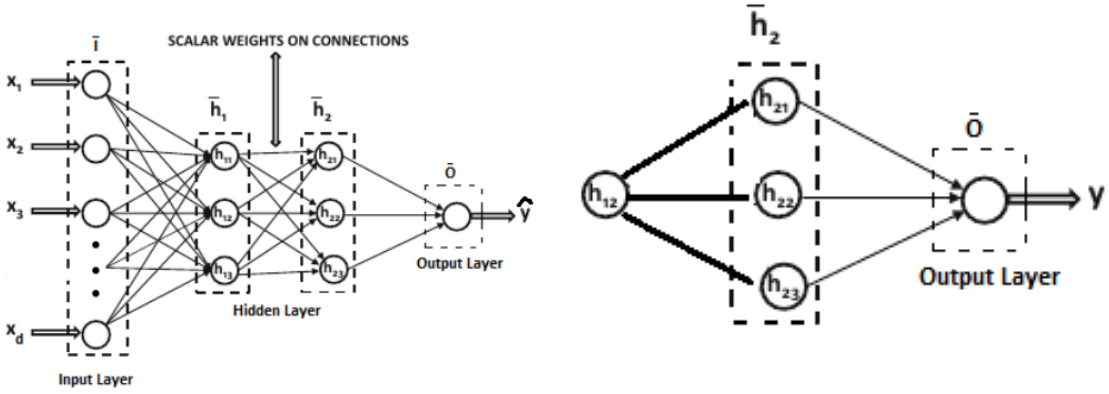
El vector de pesos se denomina $\bar{W}$, cuya dimensión es d; b es el umbral del perceptrón y $f(X)$ se denomina *función de activación*. La señal de salida es definida como *predicción*. El valor $\hat{y}$ es de la misma naturaleza que $\bar{y}$, el cual se define como *valor observado*. El error de predicción se calcula mediante $L(\hat{y}, \bar{y})$, donde L se denomina *función de pérdida*. El objetivo de una red neuronal es definir el vector de pesos $\bar{W}$ adecuado tal que el error sea mínimo. [1].

El computo del conjunto de entrada con el vector de pesos es obtenido por el producto interno $\langle \bar{X}, \bar{W} \rangle$. El *umbral* es un valor invariante propia de cada perceptrón, el cual se añade como una constante adicional que centra la predicción. Dentro de la red neuronal, este resultado se conoce como *valor pre-activación* y se representa con el signo Σ en la Figura 1.1. El *valor pos-activación* se representa por la letra Ω, y hace referencia al valor Σ evaluada en una función continua y diferenciable, denominada *función de activación*, el cual define propiamente la predicción $\hat{y}$. El resultado $f(\Sigma)$ se relaciona estrechamente con el dominio del valor observado.

1.2.1. Redes Neuronales Multicapas

Multi-layer neuronal network a Una arquitectura donde se asume que todas las respuestas de salida de los perceptrones anteriores son valores de entrada para la siguiente capa se denomina *red neuronal de retroalimentación completa* [19]. Este tipo de redes tienen la capacidad de aprender representaciones compactas y de baja dimensión para un conjunto de entrenamiento mediante el uso de una capa de codificación seguida de una capa de decodificación, que reconstruye los datos de entrada. Se identifican tres grupos de capas: *input layer*, *hidden layer* y *output layer*. La estructura individual de cada uno se representa al igual que en la Figura 1.1, y la arquitectura conjunta se representa en la Figura 1.2 (a) [1].

Al igual que en el apartado anterior, podemos definir un conjunto de entrenamiento $(\bar{X}, \bar{y})$, donde $\bar{X}$ tiene dimensión d. En la Figura 1.2, se presenta una red neuronal formada por cuatro capas: una red de entrada $\bar{I}$, dos capas ocultas, denominadas $\bar{h}_1$ y $\bar{h}_2$ de dimensión p_{h1} y p_{h2} respectivamente; y una capa de salida $\bar{O}$. Al tener capas interconectadas entre si, la idea del vector de pesos $\bar{W}$ se reemplaza por una matriz de pesos $W_{p_{hi+1} \times p_{hi}}$, donde $i = 1, 2, \ldots, k-1$ asociada a cada una de las $k-1$ capas [1].



(a) Red Neuronal Multicapa (b) Backpropagation

Figura 1.2: Tomado de [1].

La transformación de un vector $\bar{X}$ durante el entrenamiento en una red neuronal cuya arquitectura de retroalimentación completa se representa mediante:

$$\begin{aligned} \bar{h}_1 &= f(W_{p_{h1}\times d}\bar{X}) && \text{Input Layer} \\ \bar{h}_{i+1} &= f(W_{p_{hi+1}\times p_{hi}}\bar{h_i}), \quad i = 1, 2, \ldots, k-1 && \text{Hidden Layer} \\ \bar{O} &= f(W_{p_O\times p_{hk}}h_k) && \text{Output Layer} \end{aligned} \tag{1.2}$$

El comportamiento recursivo de la ecuación 1.2 nos permite interpretar el valor de la predicción $\hat{y}$ como el resultado de una composición de múltiples funciones. En la Figura 1.2, se utiliza como única función de activación a $f(\bar{X})$, que es compartida por todas las capas dentro ecuación 1.2. Para una red neuronal completamente conectada, el valor de la predicción $\hat{y}$ es el resultado de una cadena de funciones multivariables compuestas. Por consiguiente, la naturaleza *no lineal* de las funciones de activación permite que una red neuronal multicapa pueda considerarse como un estimador universal de funciones [6].

Se define como función de pérdida $L(x, y)$ como la función objetivo dentro de una red neuronal multicapa, la cual tiene un profundo impacto en el rendimiento del modelo y sus hiperparámetros. La función de pérdida sirve como un objetivo que el algoritmo de optimización se esfuerza por minimizar durante el entrenamiento. Esta función proporciona

una medida de la discrepancia entre las predicciones $\hat{y}$ y el valor observado $\bar{y}$ que guía el ajuste para $W_{p_{hi+1}}$ en cada una de las capas asociadas [1]. La elección de la función de pérdida puede afectar la velocidad de convergencia del algoritmo de optimización y la capacidad de generalización de la red, además, también puede influir en el comportamiento de otros hiperparámetros en el modelo [19].

1.2.2. Aprendizaje por retro-propagación.

En un modelo de optimización semi-heurístico, se utilizan algoritmos iterativos que actualizan los parámetros libres en cada iteración con el fin de reducir la función de pérdida. Tomando como ejemplo la Figura 1.1, en un perceptrón se utiliza el gradiente $\nabla L(\hat{y}, \bar{y})$ para conocer la *"dirección"* en la cual se maximiza nuestro error de predicción, entonces usamos este resultado como *guía* para encaminarnos a encontrar un mínimo de la función. Este método se representa en la siguiente ecuación: $\bar{W} \leftarrow \bar{W} - \alpha \nabla L$. El parámetro α se conoce como *factor de aprendizaje* y este algoritmo se denomina *método del gradiente descendiente.*

Para una red neuronal completamente conectada, el valor de la predicción es el resultado de una cadena de funciones multivariables compuestas, por lo cual el computo del gradiente se divide en dos fases: **Forward Phase**: En esta fase obtiene los valores de salida para cada nodo en las capas internas, así como el valor de la predicción, y se evalúa la función de pérdida para este resultado. **Backward Phase**: En esta fase, se calcula el gradiente de la función objetivo para cada una de las variables libres en las capas internas, comenzando desde la capa de salida. Tomando como ejemplo la arquitectura presentada en la Figura 1.2 (b), la derivada parcial de la función objetivo L con respecto al peso asociado en el nodo h_k en la capa h_{12} $w_{(h_{12},h_k)}$ se representa mediante:

$$\frac{\partial L}{\partial w_{(h_{12},h_k)}} = \frac{\partial L}{\partial O}\left[\sum_{k=[h_{21},h_{22},h_{23}]} \frac{\partial O}{\partial h_i}\Pi_{j=[h_{21},h_{22},h_{23}]}\frac{\partial h_j}{\partial h_{j-1}}\right]\frac{\partial h_{12}}{\partial w_{(h_{12},h_k)}} \tag{1.3}$$

Capítulo 2

Preprocesamiento de Imágenes con base en Kernels Gaussianos

2.1. Conteo de Multitudes

Los escenarios congestionados suelen ser un problema en muchos algoritmos de detección o clasificación. La oclusión desestima las características que estas arquitecturas pueden abstraer, así como dar paso falsos negativos [5]. Para hacer frente a este problema, se utiliza el mapa de *Densidad de Multitud*. La idea fue presentada por [11], y consiste en representar la imagen con una función $R : \mathbb{N}^2 \rightarrow \mathbb{R}$, donde la integral sobre una región B sea igual a la suma total de objetos en dicha región sobre la imagen principal. Adicionalmente, el mapa de densidad permite preservar la localización del objeto, que es de utilidad en algoritmos de seguimiento. En la Figura 2.1, se puede apreciar tanto un ejemplo de escenario congestionado **(a)**, como el mapa de densidad asociado **(c)**

(a) Imagen Real (b) anotación por puntos (c) Ground Truth

Figura 2.1: Comparativa imagen de escenario congestionado, junto al mapa de puntos y densidad generado. Tomado de [23]

2.1.1. Estimación por Mapas de Densidad

Sean $I_1, I_2, \ldots, I_N$ un conjunto de imágenes, donde cada elemento se representa como un grupo ordenado de unidades denominadas *pixeles*, agrupadas mediante una matriz *2D*. De esta forma, podemos asociar a este conjunto un tensor $T(ID, h, b, ch)$. La primera dimensión es la identificación de una imagen individual en el conjunto, mientras h y b son la altura y ancho en pixeles respectivamente. La dimensión ch asocia a cada pixel de una imagen un vector x_p de tres dimensiones, que representa su codificación RGB. Para la imagen I_k, se pueden contar n_k objetos en total, los cuales se determinan por medio de cajas limitantes o anotación por puntos [11]. En la Figura 2.1 se presenta el mapa de objetos identificados por puntos para el escenario congestionado **(b)**.

Para obtener el mapa de densidad asociado, se emplea la metodología de *estimación por kernel gaussiano*, la cual es utilizada dentro de la estadística no paramétrica para estimar la función de densidad para un conjunto de datos [22]. Supongamos una imagen matricial A_k con sólo un objeto en el pixel $\mathbf{p}$, $\mathbf{p} \in \mathbb{Z}^2$, lo cual se puede representar por medio de la función: $h(x) = \delta(\mathbf{x} - \mathbf{p})$, donde $\mathbf{x} \in A_k$. El mapa de densidad D corresponde a la convolución entre $h(x)$ y $G_\sigma(\mathbf{x})$, donde $G_\sigma(\mathbf{x}) = \frac{1}{2\pi} \exp(-\frac{x_1^2+x_2^2}{2\sigma})$ se denomina *kernel* o *filtro gaussiano*, y σ es el ancho del filtro, o también denominado *escala*. Esto se representa

en la siguiente ecuación:

$$\begin{aligned} D(\mathbf{x}) &= H(x) * G_\sigma(\mathbf{x}) \\ &= \delta(\mathbf{x}-\mathbf{p}) * G_\sigma(\mathbf{x}) \\ &= G_\sigma(\mathbf{x}) * \delta(\mathbf{x}-\mathbf{p}) \\ &= G_\sigma(\mathbf{x}-\mathbf{p}) \\ &= \frac{1}{2\pi\sigma^2}\exp\left(-\frac{1}{2\sigma^2}\|\mathbf{x}-\mathbf{p}\|^2\right) \end{aligned} \tag{2.1}$$

En este caso, la suma de total sobre D en una región B será cercano a $\mathbf{1}$ sólo si $\exists N_\sigma(p) \in B$. La precisión del resultado anterior dependerá del ancho de filtro: para un valor pequeño de σ, la suma sobre una menor región B será significativamente cercano a $\mathbf{1}$. Supongamos una imagen real I_k, para la cual se proporciona una lista con la localización (en pixeles) de cada objeto de interés en la imagen; dicha información se guarda como $P_{I_k} = \{P_1, P_2, \cdots, P_{n_k}\}$, donde P_i es un punto *2D*. De esta forma, el conteo sobre la imagen I_k se representa por medio de la función:

$$H(\mathbf{x}) = \sum_{P \in P_{I_k}} \delta(\mathbf{x}-P) \tag{2.2}$$

Sea $F_{GT}^{(k)}(\mathbf{x})$ una matriz denominada como *mapa de densidad observada* para la imagen I_k. En base al resultado del apartado anterior, este se definiría como:

$$\begin{aligned} F_{GT}^{(k)}(\mathbf{x}) &= H(\mathbf{x}) * G_\sigma(\mathbf{x}) \\ &= \sum_{P \in P_{I_k}} [\delta(\mathbf{x}-P) * G_\sigma(\mathbf{x})] \\ &= \sum_{P \in P_{I_k}} \frac{1}{2\pi\sigma^2}\exp\left(-\frac{1}{2\sigma^2}\|\mathbf{x}-P\|\right) \end{aligned} \tag{2.3}$$

Por lo tanto, la función de densidad $F_{GT}^{(k)}$ se refiere a una representación gráfica de la distribución de un conjunto de puntos I_k en dos o más dimensiones. Este mapa se construye al asignar una función gaussiana centrada en cada punto de los datos, y se suma todas estas funciones para obtener una función de densidad global, que a su vez es similar a

la suma de *funciones de densidad normal bivariada* con media en la localización de cada objeto y matriz de covarianza proporcional a la matriz identidad por un factor σ. La importancia sobre esta representación está en que la suma sobre toda la imagen I_k nos dará el valor n_k sin necesidad de recurrir a P_{I_k} preservando la localización del dichos objetos, siendo así un conjunto ideal para el entrenamiento dentro de un algoritmo de aprendizaje automático enfocado en el conteo de objetos.

Existen diversos tipos de algoritmos enfocados en el conteo de multitudes, como los métodos por conteo de regresión o detección, en el cual emplean distintas técnicas de procesamiento de imágenes para la extraer características del conjunto de entrenamiento [12]. Sin embargo, el método de conteo por estimación de mapas de densidad se ha demostrado ser una herramienta valiosa, puesto que este método es escalable y puede ser aplicado a multitudes de diferentes tamaños y densidades, lo que ha sido de utilidad para la investigación y la planificación de eventos públicos. Tanto la oclusión y los diferentes escenarios de un mismo conjunto limitan la capacidad de generalización sobre los algoritmos de conteo de multitudes, como sucede con métodos de conteo usando *detección* [17]. Por otra parte, los métodos de conteo por estimación del mapa de densidad han presentado los mejores resultados en diversos conjuntos de entrenamiento dentro del campo de visión por computadora, sobretodo en algoritmos de aprendizaje automático profundo supervisado [5].

2.1.2. Geometry Adaptive Kernels

Las noción de profundidad en imágenes bidimensionales puede verse distorsionada por la distancia entre el objeto y observador, que afecta directamente la escala que percibimos la imagen [14]. Para un conjunto de puntos P_{I_k} distribuidos **no** aleatoriamente, es factible estimar su función de densidad por medio del kernel gaussiano [22]. De esta manera, [23]

propone estimar la matriz de escala causada por la distorsión geométrica como un valor proporcional a la *distancia promedio* a los k-objetos más cercanos como un sustituto a la falta de información espacial en la cual fue tomada la imagen. Entonces, la ecuación 2.4 se escribiría como:

$$F_{GT}^{(k)}(\mathbf{x}) = \sum_{P \in P_{I_k}} \delta(\mathbf{x} - P) * G_\sigma(\mathbf{x}), \qquad \text{donde} \quad \sigma = \beta \bar{d}_P^k \tag{2.4}$$

Donde $\bar{d}_P^k$ es la distancia promedio (en pixeles) entre los k-puntos más cercanos al punto P. La metodología se denomina *filtro gaussiano con adaptación geométrica* y es utilizado en conjuntos de imágenes en aglomeraciones tal que su punto de visión capture los objetos en más de un tamaño. El resultado sobre esta metodología ha sido prometedor para diferentes algoritmos sobre distintos dominios [12] [17] [5]. Como resultado, el mapa de densidad asociado podría ser generado por filtros de diferentes escalas, por lo cual los algoritmo de conteo por estimación de densidad deberá tener estructuras receptivas locales apropiadas a estas escalas.

2.2. Filtros de Derivadas Gaussianas

En la Teoría de Espacio-Escala se plantean axiomas estructurales sobre como actúan los operadores visuales dentro de los campos receptivos. La Teoría es presentada por *Lindeberg* [13] [14], y ha formalizado estas nociones en axiomas matemáticos. La idea principal sobre el procesamiento de imágenes inicia desde su noción física como un *front-end* visual, el cual se refiere a la etapa de pre-procesamiento visual: la imagen bidimensional el resultado de la integración sobre una proyección de luz un objeto 3D sobre una área espacial B en un instante t capturado por sensor artificial (o natural) con ancho de longitud s. Por lo tanto, dicha imagen se verá afectada directamente por factores como: distancia y dirección de visualización, movimiento relativo, muestreo espacial o temporal, posición,

orientación o movimiento 3D, entre otros. En principio, toda la teoría se basa en estos cuatro axiomas:

- **Espacio de Escala lineal**: No existe una forma preferida de combinar las observaciones. En otras palabras, los operadores visuales deberán actuar como un espacio lineal sobre el espacio de escala.
- **Invariancia de Escala**: La representación de espacio de escala de una señal debe ser invariable bajo la escala espacial de la señal original. De esta forma, se plantea que para dos operadores visuales sobre una imagen observada, existirá una transformación afín que correspondiente entre ambas representaciones.
- **Naturalidad**: La representación escala-espacio de una señal debe preservar las estructuras geométricas y topológicas de la señal original. Este axioma se puede traducir en: simetría rotacional.
- **No creación de nuevas estructuras**: La representación deberá permanecer exactamente igual bajo una transformación suave, esto incluye transformaciones de escala.

Los axiomas de invariancia de escala y desplazamiento lineal conducen a la observación de que la imagen observada debe ser resultado de un operador T_s sobre una imagen real a través del operador de convolución [14]. De esta manera, el problema de definir un conjunto de operaciones visuales puede ser formulado como encontrar una familia de operadores T_s que actúan sobre una imagen f para producir una familia de nuevos representaciones de imágenes intermedias $L(:, s)$. En base a [14] [13], se puede demostrar que bajo los axiomas presentados, este espacio de escala puede ser equivalentemente construido por convolución con **kernels gaussianos affines**. De esta forma, el espacio-escala que conserva la simetría rotacional, es obtenido por:

$$L(\bar{x}; s) = \int_{\xi \in \mathbb{R}^n} f(\bar{x} - \xi) g(\xi; s) d\xi \tag{2.5}$$

Donde $g : \mathbb{R}^N \times \mathbb{R}^+ \to \mathbb{R}$ hace referencia a un kernel gaussiano isotrópico.

$$g(\bar{x}; s) = \frac{1}{(2\pi s)^{N/2}} \exp(\frac{-||\bar{x}||^2}{2N}) \tag{2.6}$$

El espacio-escala gaussiana es un método ampliamente utilizado para representar imágenes en la teoría del espacio de escala. El método se basa en la idea de convolucionar la imagen original con una serie de kernels gaussianos $g(:, s)$, cada uno de los cuales tiene una desviación estándar o **ancho de escala** diferente s, lo que da como resultado una familia de representaciones $L(:, s)$ que capturan las estructuras intrínsecas de la imagen a diferentes escalas. El espacio de escala gaussiana tiene varias propiedades importantes que lo convierten en una herramienta eficaz y robusta para el análisis de imágenes en visión artificial y procesamiento de imágenes [21].

El espacio gaussiano proporciona una representación continua y suave de la imagen original, lo que lo hace muy adecuado para analizar las propiedades del espacio de escala de la imagen. De manera similar, este espacio es **invariante** bajo la escala espacial. En otras palabras, si se agranda o se reduce el ancho de escala, la función de densidad seguirá siendo una función de distribución gaussiana, conservando las propiedades originales. El resultado anterior es importante para tareas como la detección de características, donde es necesario identificar estructuras en la imagen que persisten en diferentes escalas [14]. El espacio de escala gaussiana se basa en derivadas gaussianas, que proporcionan una representación natural del gradiente de la imagen. Esta información de gradiente es crucial para tareas como la detección de bordes y la segmentación de imágenes, donde es necesario identificar cambios en la intensidad de la imagen [21].

2.2.1. Operadores de Derivadas Gaussianas

Los operadores de derivados gaussianos son una familia de operadores lineales que se utilizan para calcular el gradiente de una señal a diferentes escalas. Estos operadores se obtienen a partir del derivadas parciales sobre un kernel con respecto a cada una de sus dimensiones espaciales, lo que da como resultado una serie de filtros derivados gaussianos [14]. En otras palabras, un operador de derivadas gaussianas, aplica un kernel gaussiano a la señal original, lo que suaviza la señal y elimina el ruido. Posteriormente, se calcula la derivada de la señal suavizada en una dirección específica. En la Figura 2.2 podemos apreciar el efecto de estos operadores en una imagen bidimensional.

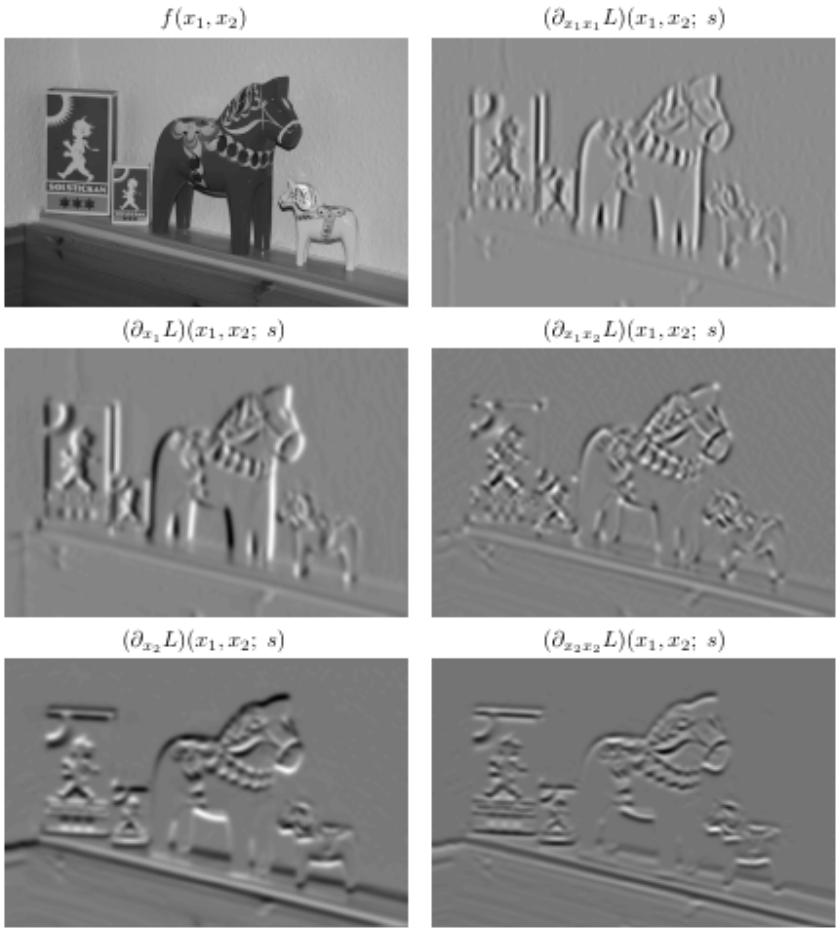


Figura 2.2: Ilustración del efecto de los operadores de derivadas gaussianas sobre una imagen bidimensional. Tomado de [14]

Debido a la linealidad del espacio-escala gaussiano, todas estas derivadas del espacio de

escala satisfacen propiedades de espacio-escala similares en términos de $L(:, s)$. Además, por la propiedad conmutativa entre convolución y diferenciación, estas derivadas de escala también se pueden calcular aplicando operadores de **derivadas gaussianas.**

$$L_x(:, s) = \partial x L(:, s) = (\partial x g(:, s)) * (f(:)) \tag{2.7}$$

A partir de combinaciones lineales de derivadas parciales, también podemos calcular derivadas en cualquier dirección, al expresar las derivadas direccionales en términos de $(cos\phi, sin\phi)$. Con respecto a las deformaciones de la imagen, las propiedades de cierre de este espacio de escala original están restringidas a traslaciones, rotaciones y reescalamiento. Por otro lado, este concepto de escala-espacio es separable correspondiente a la convolución con núcleos gaussianos unidimensionales a lo largo de cada dimensión, lo que mejora la eficiencia computacional en implementaciones computacionales. Esta idea se expresa en la ecuación, donde $\bar{x} \in \mathbb{R}$ 2.8:

$$\begin{aligned} L_{\bar{x}}(\bar{x}, s) &= (\partial \bar{x} g(\bar{x}, s)) * (f(:)) \\ &= (\partial x_1 g(x_1, s)) * (f(:)) * (\partial x_2 g(x_2, s)) * (f(:)) * \ldots * (\partial x_N g(x_N, s)) * (f(:)) \end{aligned}$$

2.3. Red Neuronal de Derivadas Gaussianas

Las redes neuronales convolucionales (CNN) se han convertido en una opción popular para las tareas de procesamiento de imágenes en los últimos años debido a su capacidad para aprender y extraer automáticamente características significativas de los datos de imágenes [1]. Una variación de las CNN que utiliza derivados gaussianos como su estructura primitiva ha mostrado resultados prometedores en varias aplicaciones de visión artificial, estas se conocen como redes neuronales de derivadas gaussianas. En estas arquitecturas, los campos receptivos se basan en este tipo de operadores, con lo cual, la señal resultante será una combinación lineal sobre esta base. Según un estudio de [4], la incorporación de derivados gaussianos como componente básico de las CNN mejoró la

precisión y la eficiencia del modelo cuando se aplicó a tareas de detección de objetos. En su investigación, los autores propusieron una arquitectura CNN basada en derivados gaussianos que utilizaba la teoría del espacio de escala para extraer características multi-escala de la imagen de entrada. El objetivo principal sobre este tipo de arquitecturas es la obtención de resultados competitivos, con una reducción significativa en la cantidad de parámetros libres del modelo sin perder las características extraías del conjunto al utilizar operadores gaussianos. Los resultados mostraron que este enfoque superó a las CNN tradicionales que usaban filtros convolucionales simples, que a su vez utilizó un 25 % del total de parámetros de la red neuronal convolucional tradicional.

2.3.1. Covarianza de Escala

Una red neuronal con propiedad de *covarianza de escala* es aquella capaz de reconocer patrones en una imagen independientemente de su escala o tamaño. Es decir, si la imagen sobre se aplicase una transformación de escala, la función resultante conmutará con dicha operación, en otras palabras, la red neuronal seguirá siendo capaz de detectar y reconocer los mismos patrones en dicha imagen. En principio, una red neuronal puede alcanzar a conmutar con estas transformaciones si al conjunto de entrenamiento lo extendemos una capa de re-escalado automático dentro en la capa de entrada, o bien si empleamos un conjunto de transformaciones sobre el conjunto de entrenamiento de la red [21]. Sin embargo, esto implica un mayor costo computacional, o se limita a un rango discreto de factores de escalas. Por lo tanto, existen distintas arquitecturas de machine learning enfocados en manejar adecuadamente la covarianza de escala como el uso de una capa de pooling en función de la escala [23], la implementación de redes neuronales multicanales piramidales [5], o capas de convolución dilatada [12].

En la investigación presentada por *Lindeberg* [21], se define una arquitectura híbrida entre la teoría del espacio-escala y una red neuronal multicapa de aprendizaje profundo.

En base a [15], una red multicapa construida a partir de capas continuas definidas por medio de operadores de derivadas parciales garantizan que la red sea covariante escalarmente. Así mismo, una arquitectura de CNN agrupada por medio de canales de escala presentan mejores resultados en la generalización de escenarios a escalas no observadas [7]. En consecuencia, el resultado presentado en [21] es el estudio de una red de aprendizaje profundo con una estructura primitiva basado en los operadores de derivadas gaussianas con múltiples canales, donde cada uno se centra en un factor de escala, que mantiene una precisión alta en un conjunto de prueba con un mayor rango continuo de factor de escala, a partir de una conjunto discreto. Este tipo de arquitectura se denomina **red neuronal multicanal**. La diferencia sobre las características posibles que puedan abstraerse de dicha red dependerá tanto del conjunto de entrenamiento, así como del problema de interés.

2.3.2. Arquitectura

La arquitectura de una red neuronal presentada en el apartado anterior, toma como referencia a una red neuronal de multiples capas, las cuales están completamente conectadas. Definimos como I, J_j y O a la capa de entrada, la *j-ésima* capa intermedia, y capa de salida respectivamente, la dimensión de cada uno depende tanto de las dimensiones del conjunto de entrenamiento, así como de los canales de entrada y salida respectivos. En principio, estas *capas* o *layers* tienen como base un conjunto de operadores de derivadas gaussianas bidimensionales isotrópicas $G_{p,q}(x_1, x_2, \sigma_x) = G_p(x_1, \sigma_x) * G_q(x_2, \sigma_x)$, donde σ_x es el ancho de escala y p, q son el orden de la derivada parcial sobre dicha coordenada. El resultado final sobre cada layer, será una combinación lineal sobre dicha base, con lo cual el conjunto de parámetros libres se reduce significativamente frente a una capa de convolución dentro de CNN [16].

En la investigación presentada por [21], se toma como base para cada layer, un conjunto

de operadores gaussianos isotrópicos formado por las derivadas parciales de primer y segundo orden. En contraste, [16] presenta una arquitectura equivalente, que emplean, como base, un conjunto de operadores gaussianos anisotrópicos, rotados y desplazados. Este nuevo conjunto de operadores cumple con los supuestos de la teoría del espacio escala, y se plantean como un conjunto adicional de parámetros libres durante el entrenamiento del modelo, que a su vez puede ser pueden tomar valores predeterminados y así recuperar el modelo de [21]. Los resultados de [16] para un problemas de segmentación y clasificación, concluyen que dichos parámetros adicionales no influyen negativamente en los resultados, y la decisión sobre si deben ser incluidos o no, dependerá de la naturaleza del problema. La representación sobre esta base se define como:

$$\begin{aligned} G_{p,q}(\mathbf{u}, \sigma, \mu, \theta) = G_p(y_1, \sigma_{x_1}) * G_q(y_2, \sigma_{x_2}), \qquad & \text{donde} \\ (y_1, y_2) = (u_1 - \mu_1, u_2 - \mu_2); \qquad & \text{con} \\ (u_1, u_2) = (x_1 \cos\theta + x_2 \sin\theta, -x_1 \sin\theta + x_2 \cos\theta) & \end{aligned}$$

Por lo tanto, la base para la construcción de cada capa se presenta como:

$$\phi_n(\mathbf{u}^n, \sigma^n, \mu^n, \theta^n) = \{\phi_1, \ldots, \phi_N\} = \{G_{p,q}(\mathbf{u}, \sigma, \mu, \theta) | p + q \leq K\}$$

Donde $K \in \mathbb{N}$ es el orden máximo sobre nuestros operadores gaussianos. Asimismo, [16] propone tanto un layer construido con más de una base, así como una técnica para reducir el costo computacional durante el entrenamiento. Sea $T(\cdot, h, b, ch)$ el tensor asociado a una imagen de nuestro conjunto de entrenamiento, entonces nuestra capa de entrada estará definida por ch canales de entrada, N_I canales de escala de salida, y se representa mediante:

$$F_I^i(:, \sigma_{\mathbf{I}}) = \sum_{k \in [1,\ldots,ch]} \left(\sum_{n=1}^{N} w_{I,n}^i \cdot \phi_n(\mathbf{u}, \sigma_{\mathbf{I}}, \mu_{\mathbf{I}}, \theta_{\mathbf{I}}) \right) * T(\cdot, h, b, k) \tag{2.8}$$

Donde i se refiere al canal de salida del layer, $i \in [1, \ldots, N_I]$; y así, $w_{I,n}^i$ es el peso asociado del layer de entrada I, del operador n, $(n \in [1, \cdot, N])$, asociado al canal de salida i. Como

resultado, tendremos un tensor $\mathbf{F_I}(\cdot, h, b, N_I)$, que a su vez pasará a ser el nuevo tensor de entrada para la capa intermedia J_1.

La transformación asociada del *layer* J_j al layer J_{j+1}, donde $j \geq 1$ es similar al que sucede en la capa de entrada. Definimos $\mathbf{F_{J_j}}(\cdot, h, b, N_{J_j})$ el tensor resultante de la capa J_j; además, $\Gamma(\cdot)$ como la función de activación asociada. Entonces, dicha transformación resultante para el canal de escala $i, i \in [1, \ldots, N_{J_{j+1}}]$,se representa como:

$$F^i_{J_{j+1}}(:, \sigma_{\mathbf{J_{j+1}}}) = \Gamma \left[\sum_{k \in [1,\ldots,N_{J_j}]} \left(\sum_{n=1}^{M_{J_{j+1}}} w^i_{I,n} \cdot \phi_n(\mathbf{u}, \sigma_{\mathbf{J_{j+1}}}, \mu_{\mathbf{J_{j+1}}}, \theta_{\mathbf{J_{j+1}}}) \right) * T(\cdot, h, b, k) \right] \tag{2.9}$$

Donde, $M_{J_{j+1}}$ hace referencia a la dimensión de la base asociada al layer J_{j+1}. Para la capa de salida O, [21] propone una capa de *max pooling* a través los múltiples canales del tensor asociado, en la cual dicha transformación tomará los valores máximos sobre cada canal de escala. *Lindeberg* demuestra que dicho layer permite obtener una representación invariante a la escala de la imagen de entrada, lo que significa que la red neuronal se enfoca en las características más importantes de la imagen, independientemente de su escala.

Capítulo 3

Experimentación

En este apartado vamos a definir el proceso de experimentación de una red neuronal de derivadas gaussianas sobre el conjunto de entrenamiento *Shanghaitech dataset*, el cual se explora los escenarios congestionados en tanto para una zona urbana dentro en distintos puntos estratégicos de Shanghai `(parte B)` y otro que cuenta con una recopilación de imágenes de libre acceso sobre eventos donde existen multitudes distribuidas aleatoriamente en diferentes escalas `(parte A)`. Este conjunto de imágenes fue creado por [23] y es un dataset muy utilizado en los algoritmos de conteo de multitudes. Los resultados obtenidos mediante este experimento se pondrán en evidencia con los presentados en la red *CSRNet*, y la red *MCNN* presentada por [23]. El código que se realizó para esta investigación se encuentra disponible en https://github.com/AquilesBailo140/Gaussian-Derivates-Network.git

3.1. Métricas

De acuerdo a [5] [8] [17], las métricas estándar para medir el éxito de un modelo en el problema de conteo de multitudes son el error absoluto **(MAE)** y el error cuadrático

medio *(MSE)*. Ambas ecuaciones se definen como:

$$MAE = \frac{1}{N}\sum_{i=1}^{N}\left|C_{I_i}^{pred} - C_{I_i}^{gt}\right|; \qquad MSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}\left|C_{I_i}^{pred} - C_{I_i}^{gt}\right|^2} \tag{3.1}$$

Donde N se refiere al total de imágenes del conjunto de test; $C_{I_i}^{pred}$ se define como el número total de objetos en el mapa de densidad estimado de la imagen I_i; $C_{I_i}^{GT}$ es el número total de objetos en el mapa de densidad real. En términos generales, **MAE** determina la precisión de la estimación, mientras que **MSE** indica la robustez de las estimaciones.

3.2. Arquitectura de la Red

La arquitectura de la red toma inspiración directa de los trabajos presentados en [5] [12] [16] [21]. En [12] presentan una arquitectura CNN basada y pre-entrenada en la red `VGG-16` [20] que es utilizada para problemas de clasificación, por lo cual se emplea dicha red como base, y se reemplaza las capas de convolución por capas de estructuras receptivas de derivadas gaussianas. El experimento se lleva acabo en el entrono de programación Python, el cual hace uso de la librería `TensorFlow`. De manera similar, [16] ofrece un layer escrito en este mismo lenguaje, el cual se emplea para la definición de los layers basado en los operadores gaussianos. La arquitectura final se presenta dentro de la Tabla 3.2.

3.3. Conjunto de Datos e Implementación de Google Colaboratory

Para el conjunto de entrenamiento, se emplea la técnica de conteo por estimación usando mapas de densidad, con lo cual se crea un código inspirado en [12] [23], por lo

tanto se emplea la técnica de *geometric adaptative kernel* presentado en la sección 2.1.2. En estas publicaciones, utilizan como factor de adaptación $\beta = 0{,}3$, además de utilizar como referencia sólo las tres distancias más cercanas a cada individuo. Con los mapas de densidad obtenidos, estos se añaden como un canal adicional al tensor $T(\cdot, h, b, ch)$ sólo con fines prácticos. Adicionalmente, tanto el conjunto de entrenamiento como validación están a diferentes tamaños, con lo cual se emplea una función adicional que deja a un mismo tamaño ambos conjuntos, utilizando como referencia la imagen más pequeña, que es $(512, 512)$ pixeles.

Shanghaitech contiene 1198 imágenes anotadas, con un total de 330.165 personas con centros de sus cabezas anotados. Este conjunto de datos contiene consta de dos partes: hay 482 imágenes en la `parte A` y 716 `parte B`. De estos conjuntos, 300 son las imágenes de la Parte A se utilizan para el entrenamiento, y las restantes, 182 imágenes, como conjunto de test; en la parte B, esta proporción cambia a 400 y 316 imágenes. Para tener una muestra más grande, sobre el tensor resultante, se divide en cuatro ventanas iguales no conectadas, y consecuentemente se toma una muestra aleatoria con una ventana del mismo tamaño sobre cada imagen. El problema de sobre entrenamiento en nuevo data set se supera la tomar una muestra aleatoria y no consecutiva durante cada época de entrenamiento. Así mismo, tanto la imagen como el mapa de densidad asociado corresponden a la imagen original, debido a la división del tensor sobre h y b. Una observación adicional sobre este nuevo conjunto corresponde al momento de la reducción de escala por el ajuste de tamaño, pues esto presenta una discrepancia entre el mapa de densidad resultante y el original al reducir sus rango en un valor de cuatro. De esta forma, se escala sobre este mismo factor para evitar problemas en el conteo final.

La función de pérdida apropiada, según [12], [23] es la distancia euclidiana, que se define como la distancia $L2$ entre el mapa de densidad real y el estimado. La ecuación se representa como:

$$L = \frac{1}{2N} \sum_{i=1}^{N} ||F_{pred}(I_i) - F_{gt}(I_i)|| \tag{3.2}$$

Donde $F_{pred}(I_i)$ se refiere al mapa de densidad estimado de la imagen I_i; $F_{gt}(I_i)$, es el mapa de densidad real, y N es el total de imágenes en el conjunto para este *mini-batch*. Se utiliza como algoritmo de optimización para ajustar los hiperparámetros al algoritmo `ADAM`, el cual es una variante de los algoritmos de gradiente descendiente estocástico **SGD** [9], tomando un nivel de aprendizaje igual a 5×10^{-5}. Todas estas funciones fueron orientadas para ser implementados en `TensorFlow`, debido a su amplia variedad de módulos orientados construir arquitecturas de machine learning. Aunque puede tener una curva de aprendizaje pronunciada, `TensorFlow` cuenta con una documentación muy completa y una gran cantidad de recursos en línea, lo que puede ayudar a los usuarios a familiarizarse con la herramienta.

Layer (type)	Output Shape	Param #
input_4 (InputLayer)	[(None, 256, 256, 4)]	0
ftgd_conv_layer (FTGDConvLayer)	(None, 256, 256, 32)	1362
tf.nn.relu_48 (TFOpLambda)	(None, 256, 256, 32)	0
ftgd_conv_layer_1 (FTGDConvLayer)	(None, 256, 256, 32)	10322
tf.nn.relu_49 (TFOpLambda)	(None, 256, 256, 32)	0
ftgd_conv_layer_2 (FTGDConvLayer)	(None, 256, 256, 64)	20594
tf.nn.relu_50 (TFOpLambda)	(None, 256, 256, 64)	0
ftgd_conv_layer_3 (FTGDConvLayer)	(None, 256, 256, 64)	41074
tf.nn.relu_51 (TFOpLambda)	(None, 256, 256, 64)	0
ftgd_conv_layer_4 (FTGDConvLayer)	(None, 256, 256, 64)	41074
tf.nn.relu_52 (TFOpLambda)	(None, 256, 256, 64)	0
ftgd_conv_layer_5 (FTGDConvLayer)	(None, 256, 256, 64)	41074
tf.nn.relu_53 (TFOpLambda)	(None, 256, 256, 64)	0
ftgd_conv_layer_6 (FTGDConvLayer)	(None, 256, 256, 128)	82098
tf.nn.relu_54 (TFOpLambda)	(None, 256, 256, 128)	0
ftgd_conv_layer_7 (FTGDConvLayer)	(None, 256, 256, 128)	164018
tf.nn.relu_55 (TFOpLambda)	(None, 256, 256, 128)	0
ftgd_conv_layer_8 (FTGDConvLayer)	(None, 256, 256, 128)	164018
tf.nn.relu_56 (TFOpLambda)	(None, 256, 256, 128)	0
ftgd_conv_layer_9 (FTGDConvLayer)	(None, 256, 256, 128)	164018
tf.nn.relu_57 (TFOpLambda)	(None, 256, 256, 128)	0
ftgd_conv_layer_10 (FTGDConvLayer)	(None, 256, 256, 128)	164018
tf.nn.relu_58 (TFOpLambda)	(None, 256, 256, 128)	0
ftgd_conv_layer_11 (FTGDConvLayer)	(None, 256, 256, 128)	164018
tf.nn.relu_59 (TFOpLambda)	(None, 256, 256, 128)	0
ftgd_conv_layer_12 (FTGDConvLayer)	(None, 256, 256, 64)	82034
tf.nn.relu_60 (TFOpLambda)	(None, 256, 256, 64)	0
ftgd_conv_layer_13 (FTGDConvLayer)	(None, 256, 256, 64)	41074
tf.nn.relu_61 (TFOpLambda)	(None, 256, 256, 64)	0
ftgd_conv_layer_14 (FTGDConvLayer)	(None, 256, 256, 32)	20562
tf.nn.relu_62 (TFOpLambda)	(None, 256, 256, 32)	0
ftgd_conv_layer_15 (FTGDConvLayer)	(None, 256, 256, 32)	10322
tf.nn.relu_63 (TFOpLambda)	(None, 256, 256, 32)	0
depth_max_pool (Depth_MaxPool)	(None, 256, 256, 1)	0

Total params: 1,211,680
Trainable params: 1,211,680
Non-trainable params: 0

Cuadro 3.1: Se presenta un resumen detallado de la arquitectura de la red en el experimento. Como primera columna tenelos los nombres de cada una de las capas. En este caso, `ftgd_conv_layer` hace referencia a la capa basada en los operadores de derivadas gaussianas, presentados en [16]; `tf_nn_relu` hace referencia a la función de activación `relu`, programada por *TensorFlow*. La capa de entrada: `input` hace referencia a la entrada de los datos, mientras la capa de salida `depth_max_pool` representa la capa de *max pooling* sobre los canales de escala, idea presentada por [21]. Como segunda columna tenemos la forma del tensor de salida para cada una de las capas, mientras que en la tercera columna tenemos la cantidad de parámetros libres en cada layer. El total se encuentra al pie de la Tabla.

Capítulo 4

Resultados y Conclusiones

Los resultados obtenidos por nuestro modelo se presentan en la Tabla 4:

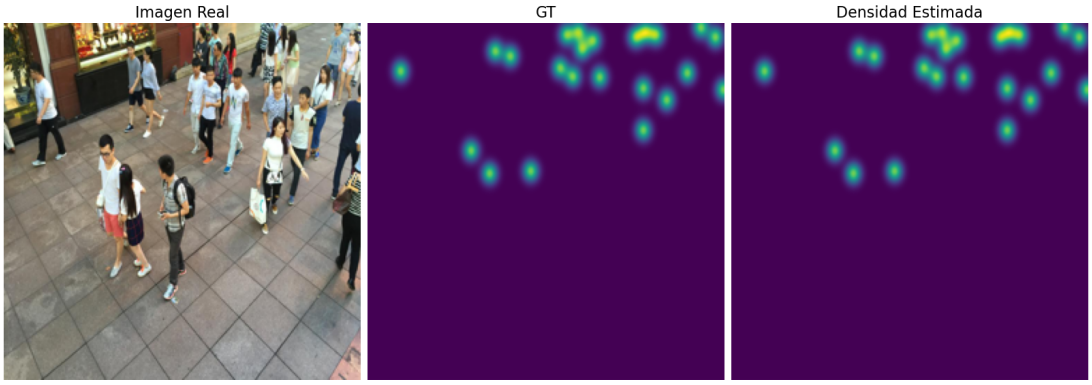
	Parte A		**ParteB**	
Método	**MAE**	**MSE**	**MAE**	**MSE**
MCNN [23]	110.2	173.2	26.4	41.3
CSRNet [12]	68.2	115.0	10.6	16.0
Betsi (nuestro método)	85.67	184.71	38.55	44.68

Una comparativa de la cantidad de parámetros en estas redes se presenta en la Tabla 4:

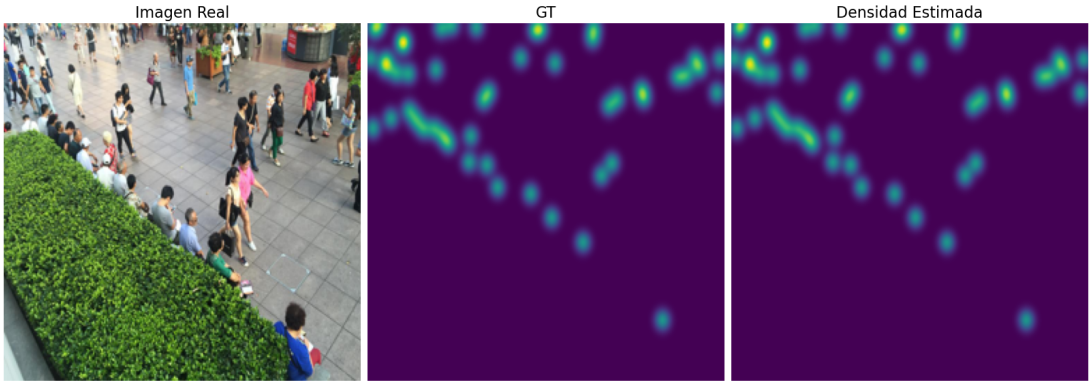
	#Parámetros
	(en millones)
Método	
MCNN [23]	0.13
CSRNet [12]	16.26
Betsi(nuestro método)	1.21

Cuadro 4.1: Cantidad de parámetros libres (en millones) en los tres métodos a comparar.

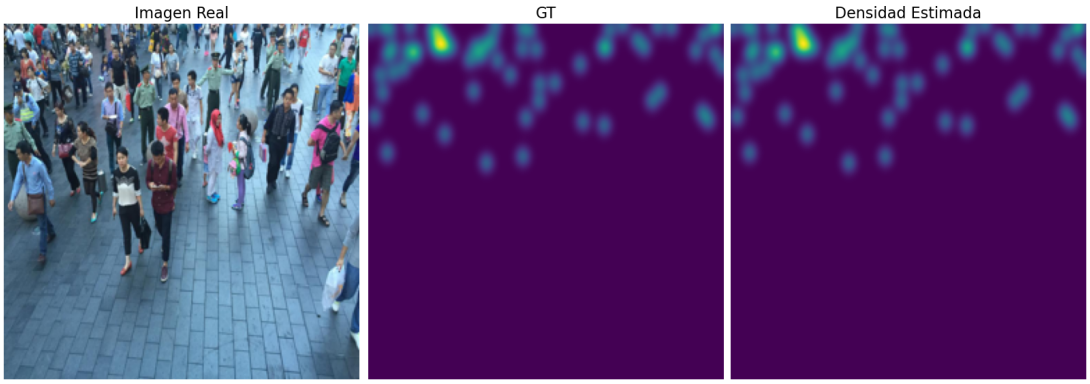
En los siguientes apartados presentamos una lista de Figuras reconstruidas sobre la parte B del conjunto de entrenamiento:



Count: 29.24; Estimate Count: 33.03; Loss: 0.00; MAE: 3.79



Count: 61.32; Estimate Count: 96.35; Loss: 0.00; MAE: 35.03



Count: 79.24; Estimate Count: 139.66; Loss: 0.00; MAE: 60.42

4.1. Conclusiones

En este estudio, se ha investigado el uso de una red neuronal de derivadas gaussianas para abordar el problema de la congestión en la base de datos ShangaiDataSet. A nivel genral, los resultados obtenidos son equiparables frente a dos arquitecturas de alto nivel que enfrentan este mismo problema, lo que demuestra el potencial sobre el uso de los operadores gaussianos como estructuras receptivas primarias, lo que a su vez podría ser útil en futuras investigaciones relacionadas al estudio del tránsito urbano. Como conclusiones específicas, tenemos:

- Las propiedades de los operadores gaussianos dentro de la teoría del espacio-escala hacen que sean ideales para ser utilizados como base para una capa de una red neuronal profunda. La capacidad de los operadores gaussianos para para obtener la invarianza y covarianza en la escala permite que los datos procesados puedan ser utilizados para recuperar información original de las imágenes del conjunto de entrenamiento, así como una correcta predicción en dominios similares. En base a los resultados obtenidos, los operadores gaussianos proporciona una poderosa herramienta para el aprendizaje profundo, en particular para el análisis de imágenes y la predicción de situaciones complejas en la vida real. Por lo tanto, la utilización de operadores gaussianos en el aprendizaje profundo es una técnica altamente efectiva que puede mejorar significativamente la capacidad de los modelos de aprendizaje profundo para procesar y analizar información visual, lo que puede tener importantes aplicaciones en diversas áreas de la ciencia y la tecnología.

- En resumen, la utilización de redes neuronales de derivadas gaussianas representa una técnica altamente efectiva para el análisis de multitudes en escenarios urbanos, al permitir la obtención de resultados precisos con un menor coste computacional. La reducción significativa de hiperparámetros que se logra con esta técnica permite el uso de modelos de aprendizaje profundo más eficientes y escalables, que son capaces de manejar situaciones de multitudes complejas en tiempo real. Además, la

técnica de redes neuronales de derivadas gaussianas permite la creación de modelos pre-entrenados, lo que amplía su dominio y aumenta la capacidad de adaptación a diferentes escenarios. En este sentido, esta técnica es una herramienta muy valiosa para el estudio y análisis de problemas de multitudes en diferentes ámbitos, tales como el control de multitudes, la planificación urbana, la seguridad pública, entre otros.

Bibliografía

[1] C. C. Aggarwal, *Neural Networks and Deep Learning*, Springer Cham, 1 ed., 2018.

[2] M. Z. Alom, T. M. Taha, C. Yakopcic, S. Westberg, P. Sidike, M. S. Nasrin, B. C. Van Esesn, A. A. S. Awwal, and V. K. Asari, *The history began from alexnet: A comprehensive survey on deep learning approaches*, 2018.

[3] Y. Bengio, *Learning deep architectures for ai*, Foundations and Trends in Machine Learning, 2 (2009), pp. 1–127.

[4] R. Burklund and A. Senger, *On the high-dimensional geography problem*, 2020.

[5] G. Gao, J. Gao, Q. Liu, Q. Wang, and Y. Wang, *Cnn-based density estimation and crowd counting: A survey*, 2020.

[6] K. Hornik, M. Stinchcombe, and H. White, *Multilayer feedforward networks are universal approximators*, Neural Networks, 2 (1989), pp. 359–366.

[7] Y. Jansson and T. Lindeberg, *Exploring the ability of CNN s to generalise to previously unseen scales over wide scale ranges*, in 2020 25th International Conference on Pattern Recognition (ICPR), IEEE, jan 2021.

[8] D. Kang, Z. Ma, and A. B. Chan, *Beyond counting: Comparisons of density maps for crowd analysis tasks - counting, detection, and tracking*, CoRR, abs/1705.10118 (2017).

[9] D. P. Kingma and J. Ba, *Adam: A method for stochastic optimization*, 2014.

[10] M. Kubat, *An Introduction to Machine Learning*, Springer Cham, 1 ed., 2015.

[11] V. Lempitsky and A. Zisserman, *Learning to count objects in images*, in Advances in Neural Information Processing Systems, J. Lafferty, C. Williams, J. Shawe-Taylor, R. Zemel, and A. Culotta, eds., vol. 23, Curran Associates, Inc., 2010.

[12] Y. Li, X. Zhang, and D. Chen, *Csrnet: Dilated convolutional neural networks for understanding the highly congested scenes*, 2018.

[13] T. Lindeberg, *Scale-Space Theory in Computer Vision*, 12 1993.

[14] T. Lindeberg, *Chapter one - generalized axiomatic scale-space theory*, vol. 178 of Advances in Imaging and Electron Physics, Elsevier, 2013, pp. 1–96.

[15] ——, *Provably scale-covariant continuous hierarchical networks based on scale-normalized differential expressions coupled in cascade*, Journal of Mathematical Imaging and Vision, 62 (2019), pp. 120–148.

[16] V. Penaud-Polge, S. Velasco-Forero, and J. Angulo, *Fully trainable gaussian derivative convolutional layer*, (2022).

[17] R. Perko, M. Klopschitz, A. Almer, and P. M. Roth, *Critical aspects of person counting and density estimation*, Journal of Imaging, 7 (2021).

[18] T. M. M. Ryszard S. Michalski, Jaime G. Carbonell, *Machine Learning: An Artificial Intelligence Approach*, Springer-Verlag, 1 ed., 1984.

[19] J. Schmidhuber, *Deep learning in neural networks: An overview*, Neural Networks, 61 (2015), pp. 85–117.

[20] K. Simonyan and A. Zisserman, *Very deep convolutional networks for large-scale image recognition*, 2014.

[21] L. Tony, *Scale-covariant and scale-invariant gaussian derivative networks*, 64 (2022), pp. 223—242.

[22] L. Wasserman, *All of Nonparametric Statistics*, Springer New York, NY, 1 ed., 2006.

[23] Y. Zhang, D. Zhou, S. Chen, S. Gao, and Y. Ma, *Single-image crowd counting via multi-column convolutional neural network*, in 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016, pp. 589–597.