Redes convolucionales avanzadas I

Visión por Computador, curso 2024-2025

Silvia Martín Suazo, silvia.martin@u-tad.com

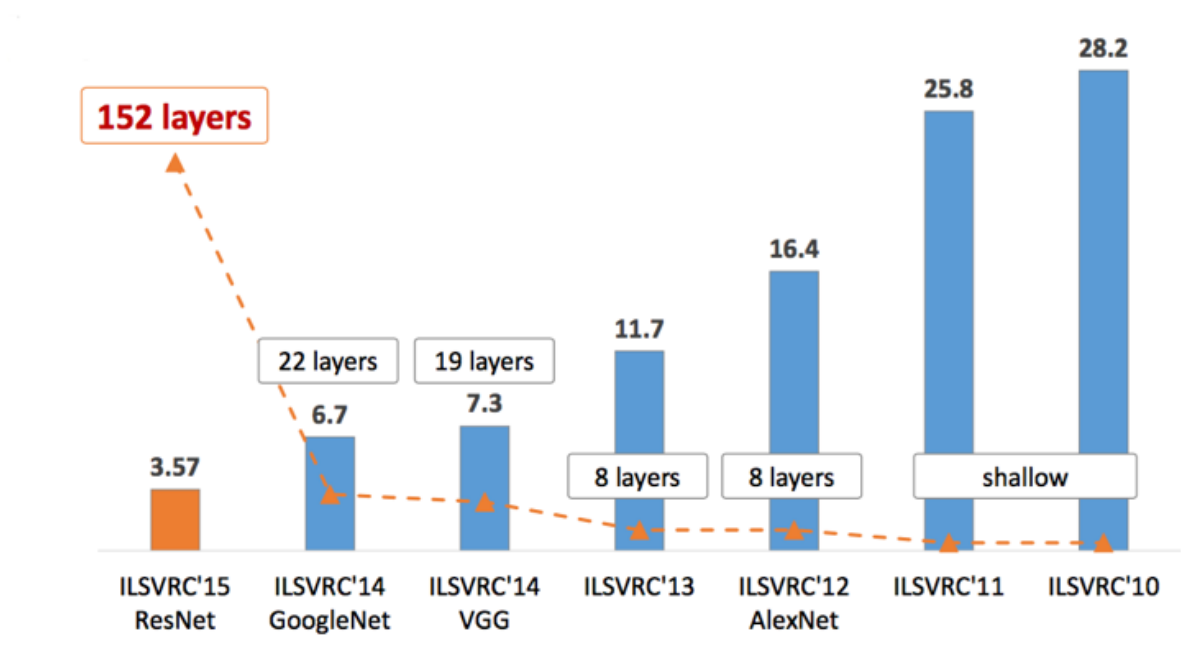
21 de noviembre de 2024

U-tad | Centro Universitario de Tecnología y Arte Digital

Introducción

Hasta ahora, se han estudiado como diseñar Convolutional Neural Networks (CNNs) siguiendo una estructura tradicional. Sin embargo, esto no es siempre así.

Los avances de los últimos años sugieren el cambio en ciertos aspectos de las CNNs para obtener mejores resultados en ciertos aspectos.

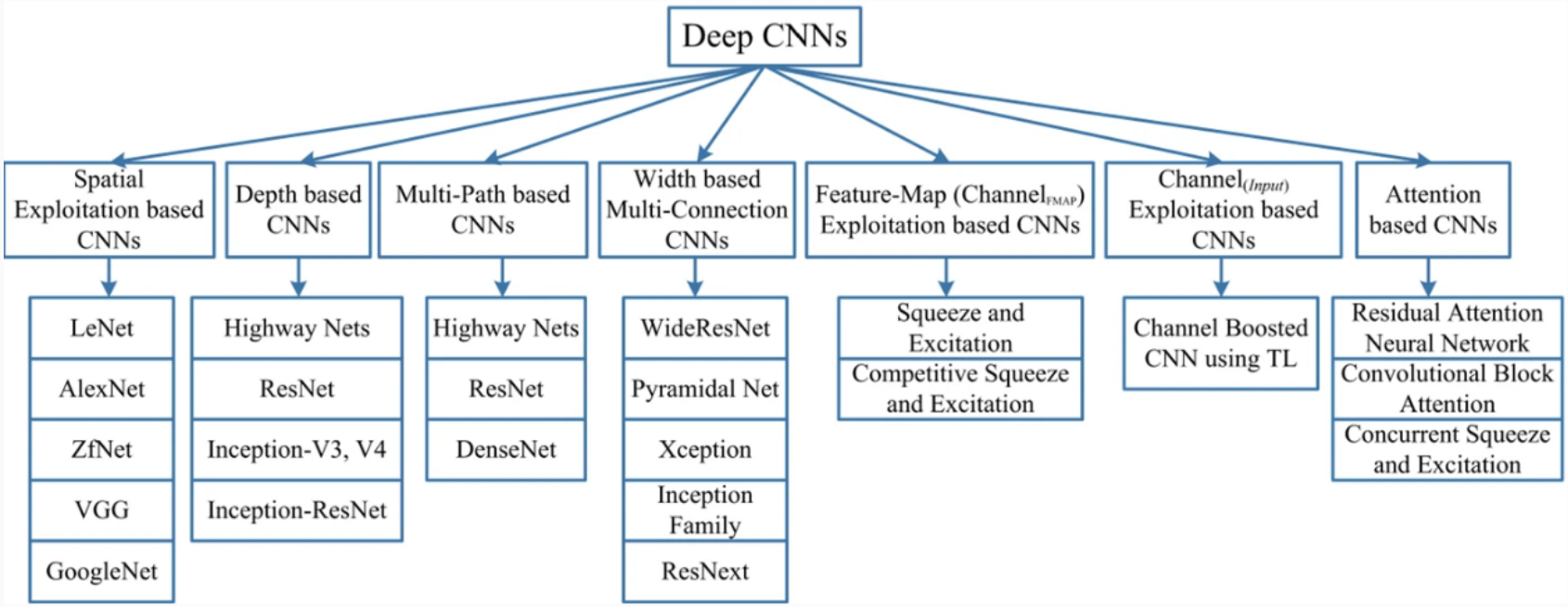


[1]

Evolución de las CNNs

Durante los últimos años la manera de diseñar, construir y entrenar CNNs ha ido evolucionando. Durante este tema se estudiarán las distintas líneas de investigación más relevantes seguidas durante los últimos años.

Cabe destacar que el paradigma investigador es extremadamente amplio y no se podrán estudiar todos los avances.



[2]

LeNet

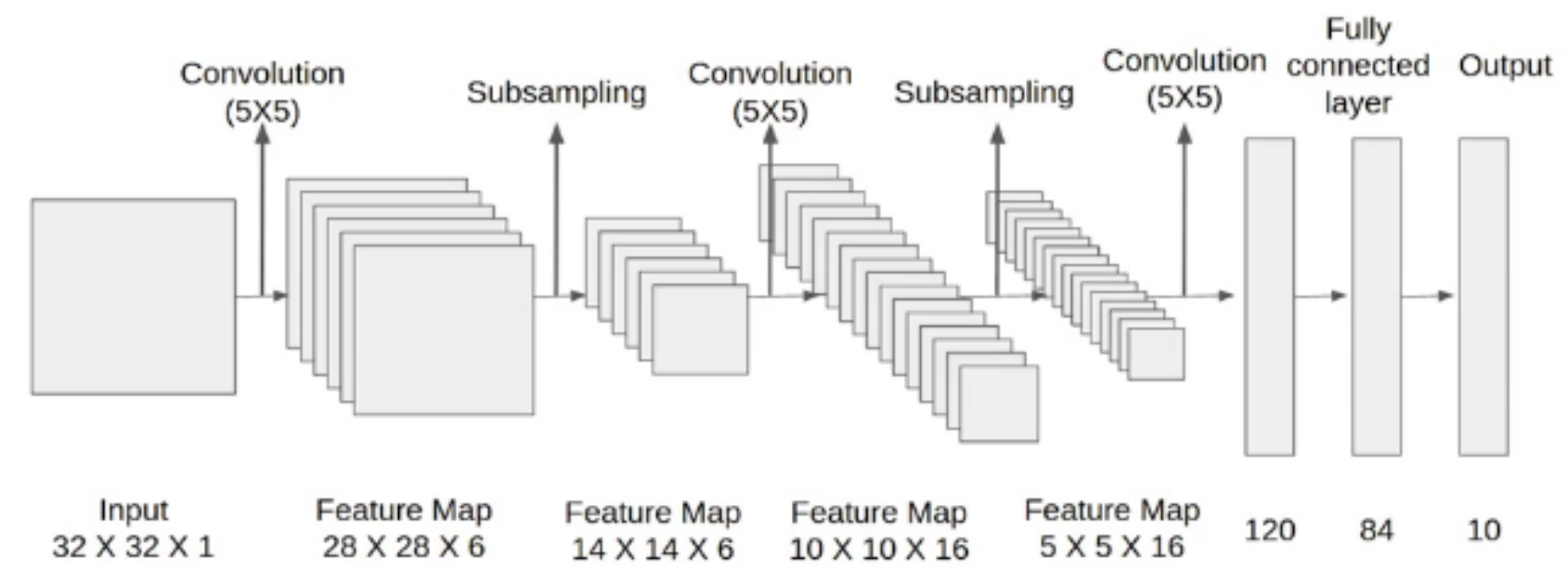
La arquitectura LeNet fue la primera arquitectura que utilizaba capas convolucionales. Introducida en el año 1988 por LeCun et al. en el artículo [3].

La arquitectura es muy simple y directa, ideal para aprender los básicos de las CNNs. Su sencillez hace que pueda ser entrenada incluso haciendo uso de CPU.

** como curiosidad su uso primario era el reconocimiento de caracteres (Optical Character Recognition (OCR)).*

LeNet

La arquitectura LeNet consiste en el uso de las siguientes capas:
INPUT -> CONV(Tanh) -> POOL -> FLATTEN -> FC(RELU) -> FC(softmax)



[4]

Originalmente la activación usada en las capas ocultas era la tangencial.

AlexNet

La arquitectura AlexNet fue publicada en el año 2012[5] y su principal aportación es la mejora de los modelos anteriores.

Para ello la principal aportación de AlexNet es la profundización en el número de capas utilizadas.

La profundidad de la red es un aspecto esencial a la hora de usar CNNs. Es por ello que a partir de este momento se busca aumentar el número de capas para mejorar el rendimiento de la red.

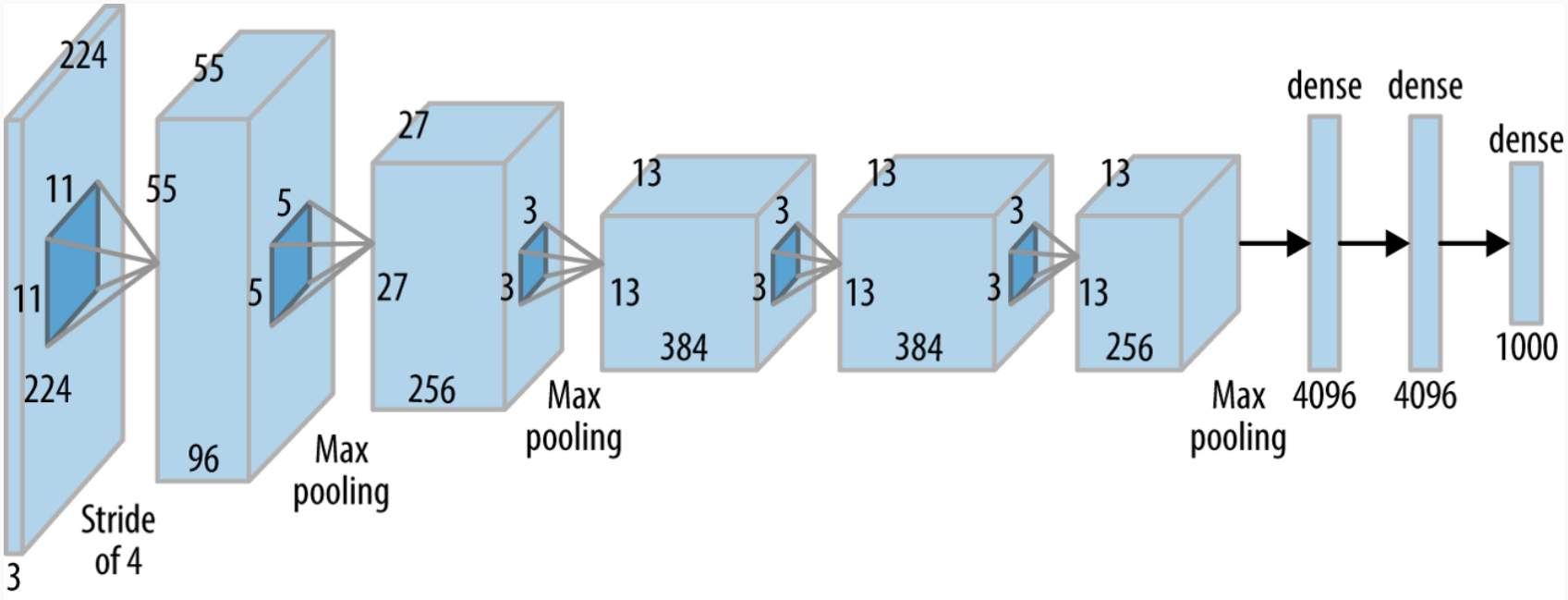
AlexNet

Los modelos anteriores tenían limitaciones a la hora de escalar su tamaño. Sin embargo hay ciertos avances que hacen posible esta arquitectura:

- Entreno haciendo uso de GPU.
- Uso de activación ReLU por primera vez.
- Regularización con Dropout.
- Data augmentation.

AlexNet

El principal hito de AlexNet fue mejorar los resultados de la competición ImageNet Large Scale Visual Recognition Challenge (ILSVRC) en el año 2012 de una puntuación de 25.8 a 16.4. En gran medida esto se debe al aumento en profundidad de la red.



[6]

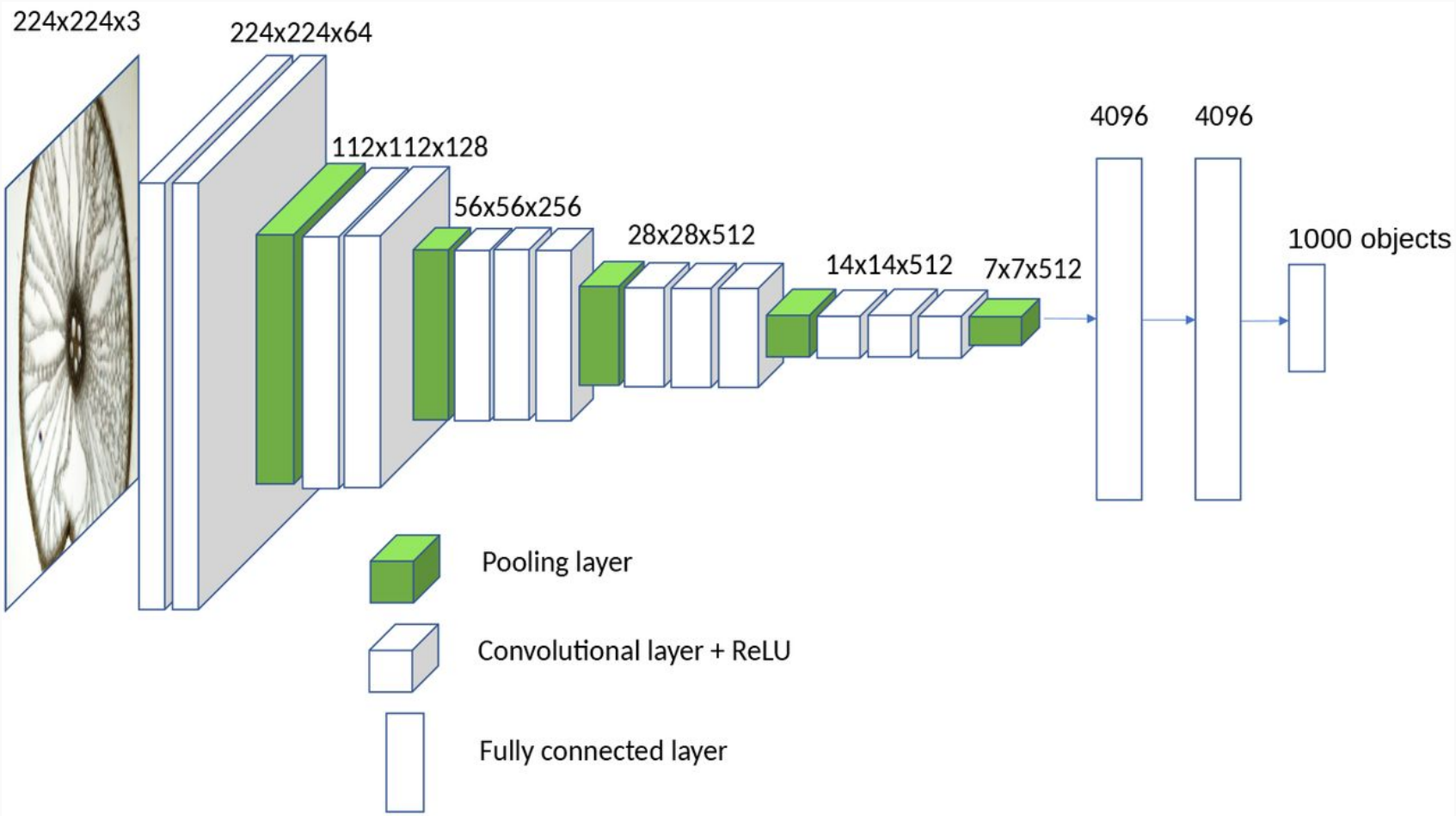
VGG

La red Visual Geometry Group (VGG)[7] supone más avances en la arquitectura de CNN para clasificación de imágenes. De la misma manera que la red AlexNet, la VGG fue la ganadora del concurso ILSVRC en el año 2014.

Respecto a AlexNet la profundidad de la red pasa de 8 a 19 capas.

VGG

La principal innovación propuesta en este trabajo son las convoluciones apiladas (stacked convolutions).



[8]

Convoluciones apiladas

La red VGG usa convoluciones consecutivas de 3x3 en vez de las anteriores convoluciones de 5x5 o 7x7.

- Dos convoluciones de 3x3: Campo receptivo de 5x5.
- Tres convoluciones de 3x3: Campo receptivo de 7x7.

Las ventajas de esta aproximación son:

- Menos parámetros.
- Más no linealidad.

Número de filtros

Otra de las innovaciones propuestas en esta red, es doblar el número de filtros a medida que el tamaño de la imagen se divide entre dos.

Data augmentation

Al mismo tiempo, se introduce aumento de datos haciendo uso de recorte, escalado de imágenes, permutación de canales y volteo horizontal.

Inception

La arquitectura Inception[9], tambien conocida como GoogleLeNet, busca mejorar el rendimiento de los clasificadores modificando la arquitectura.

Las anteriores soluciones buscaban introducir más capas de manera directa, esperando un mejor rendimiento. Sin embargo con Inception se consigue una mejora en la velocidad de entrenamiento y rendimiento al mismo tiempo.

Las versiones más populares de Inception son las siguientes:

- Inception v1.
- Inception v2
- Inception v3.
- Inception v4 and Inception-ResNet.

Inception: La premisa

Dentro de una imagen las estructuras relevantes tienen gran variabilidad en su tamaño.

Esta gran variación hace que elegir un kernel de tamaño adecuado sea una tarea complicada.

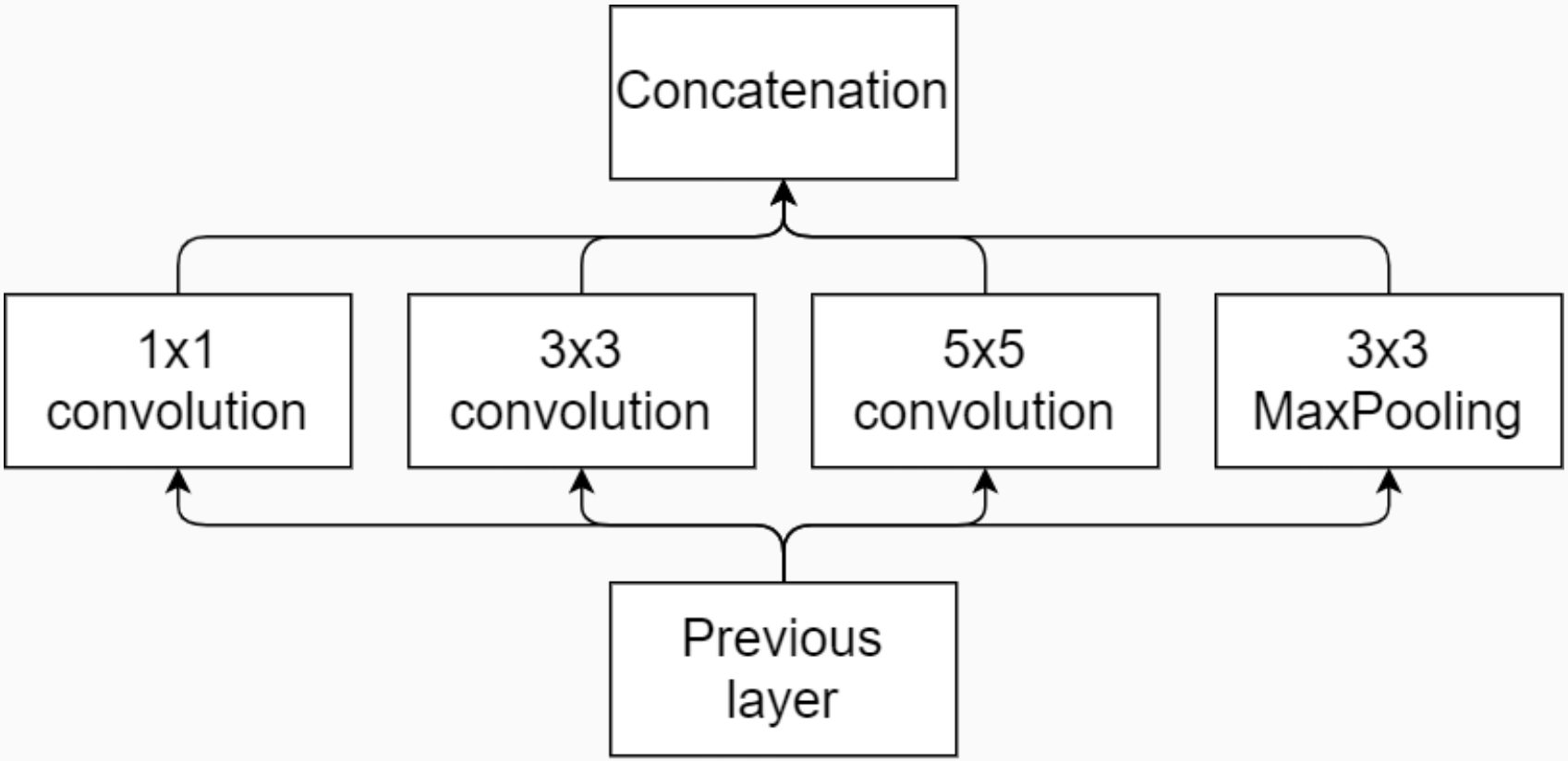
- Un kernel grande funcionaría mejor para información distribuida de manera global.
- Un kernel pequeño funcionaría mejor para información distribuida de manera local.

Además las redes muy profundas tiene tendencia al overfiting y problemas para actualizar los gradientes.

Inception

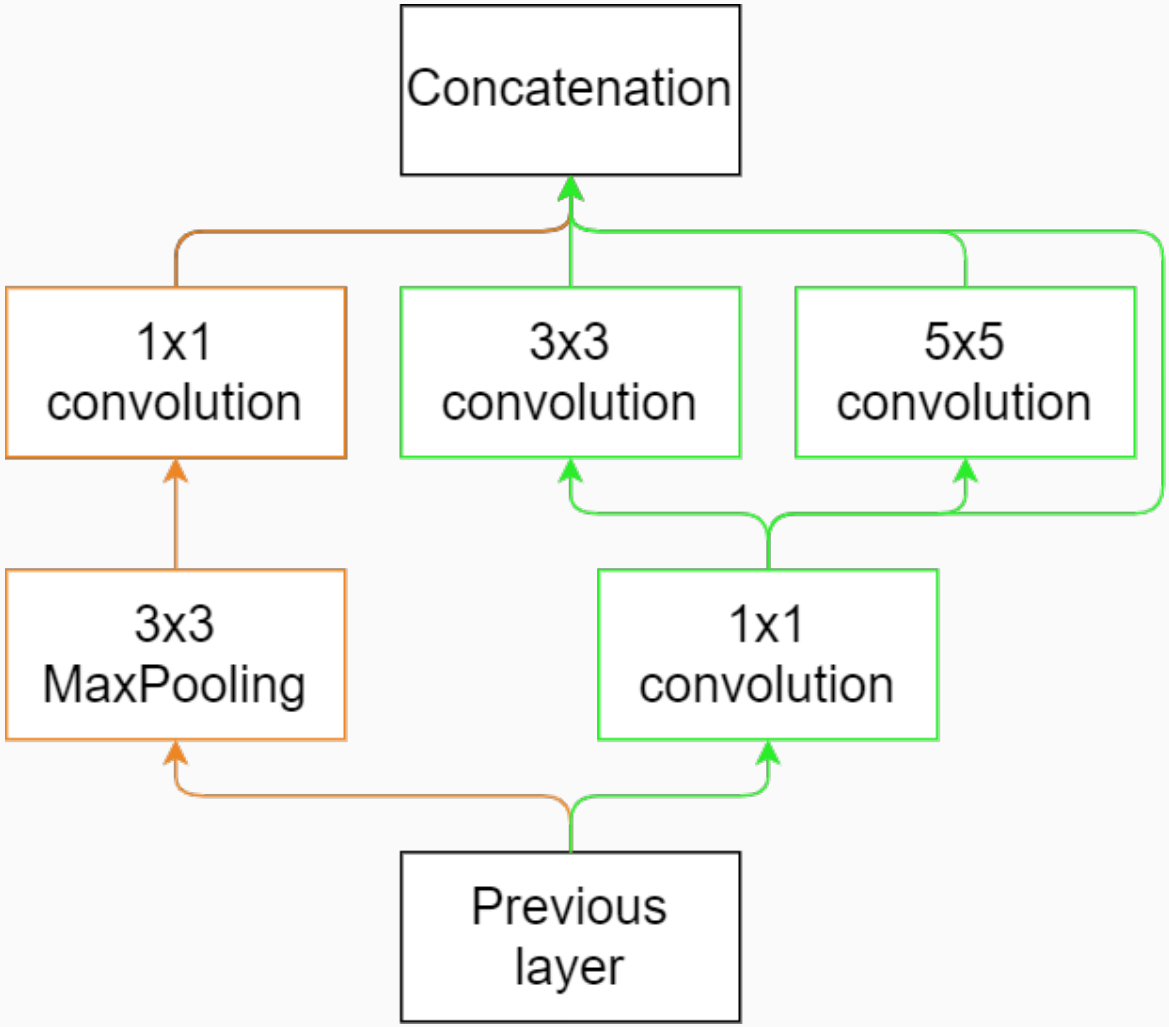
¿Por qué no tener distintos tamaño de filtro al mismo nivel?

El modulo naïve de inception realiza 3 convoluciones de distinto tamaño de filtro. El resultado de este filtrado se combina concatenando sus salidas.



Inception

El principal problema de esta solución es su excesivo coste computacional. Para evitar esto se realiza una reducción de canales haciendo uso de convoluciones de 1x1.



Inception

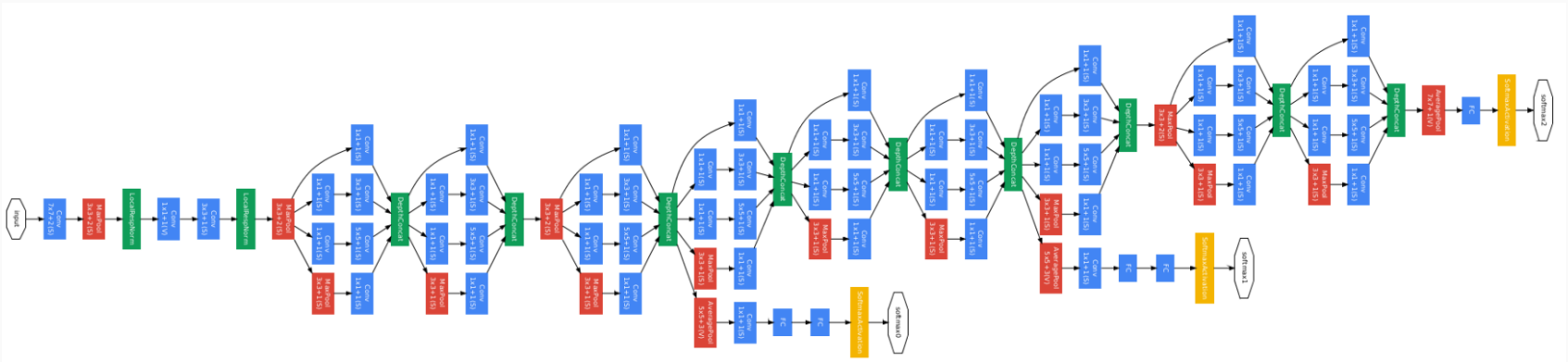
Para prevenir que las capas intermedias del modelo "mueran" por el desvanecimiento de gradiente la arquitectura Inception v1 introduce clasificadores intermedios.

Estos clasificadores auxiliares tienen sus propias arquitecturas (con convolución, pooling y capas completamente conectadas) y funciones de pérdida, las cuales son combinadas a la función de pérdida de la red de la siguiente manera:

$$Loss_{total} = 0.7 * Loss_{end} + 0.3 * Loss_{auxiliary} \tag{1}$$

De esta manera conseguimos estabilidad y una mayor eficiencia del entrenamiento al facilitar la propagación de la información. Finalmente, estos clasificadores auxiliares se desactivan.

Inception



Arquitectura de la Inception v1

Inception v2 o Inception-BN

La segunda versión de Inception propone la introducción de Batch Normalization en la red y los clasificadores auxiliares. Se consigue acelerar y estabilizar el entrenamiento.

También ayuda a solucionar el problema de la covariable interna, que es el cambio en la distribución de las activaciones de la red debido a la variación en los parámetros de la red durante el entrenamiento.

Inception v3 (antes conocida como Inception v2)

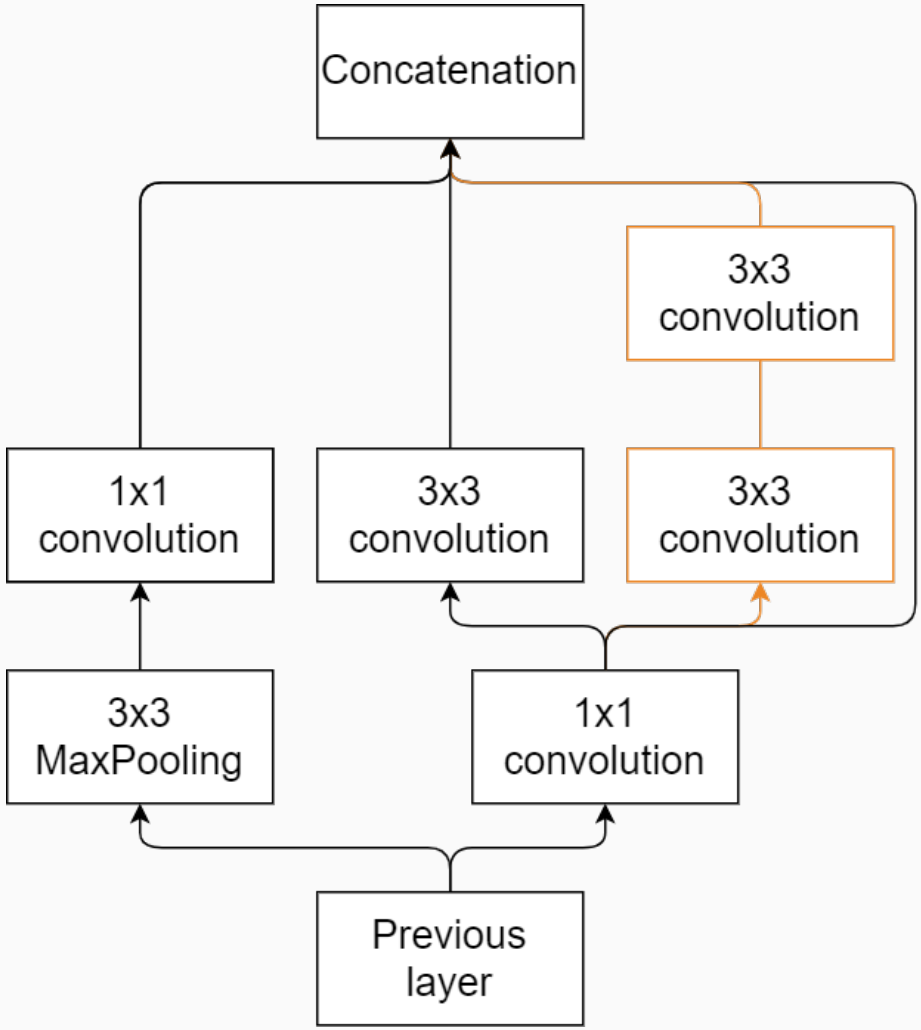
Las CNNs tienen un mejor rendimiento cuando no modifican drásticamente las dimensiones de entrada.

Una reducción muy radical puede dar lugar a pérdida de información, lo que se conoce como "**cuello de botella de representación**" (representational bottleneck).

Para ello, a partir de Inception v2 se recurre a diversas modificaciones para amplificar las convoluciones apiladas, con el fin de evitar lo máximo posible este problema.

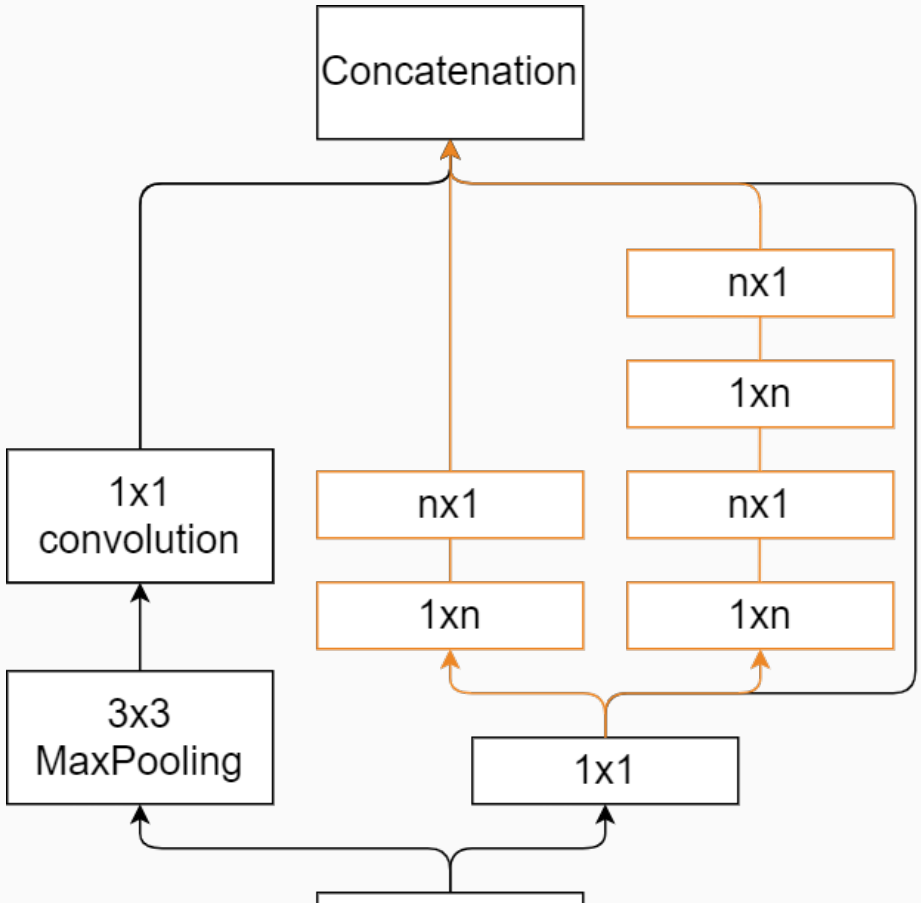
Inception v3 (antes conocida como Inception v2)

Las convoluciones de 5x5 son factorizadas a dos convoluciones de 3x3. De esta manera se mejora la velocidad de convergencia y rendimiento.



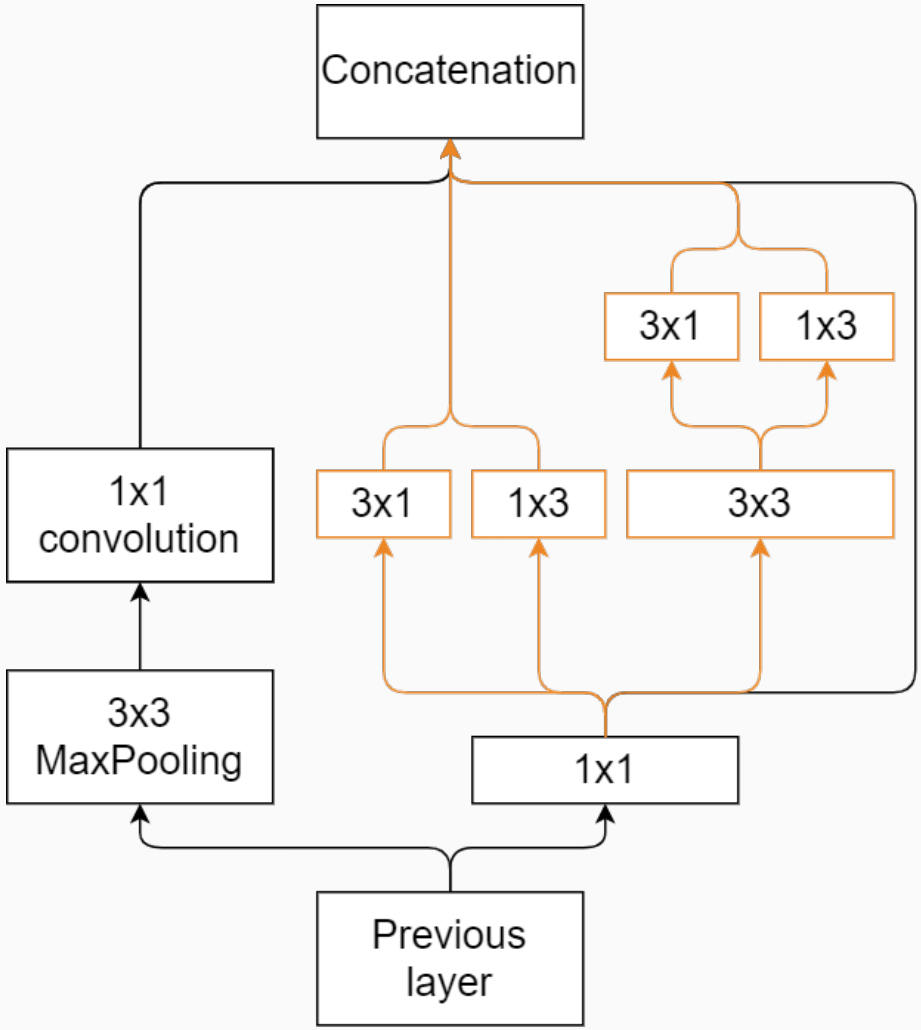
Inception v3 (antes conocida como Inception v2)

Además, cada una de estas convoluciones de *nxn* se factorizan en una convolución de 1x*n* y otra de *n*x1. Por ejemplo, en vez de realizar una convolución de 3x3 esta es factorizada en dos convoluciones, una de 1x3 y otra de 3x1. Este proceso es un 33% más “barato” que la convolución normal.



Inception v3 (antes conocida como Inception v2)

Al mismo tiempo, se expanden los conjuntos de filtros. Al expandirse en amplitud y no en profundidad se evita pérdida de información



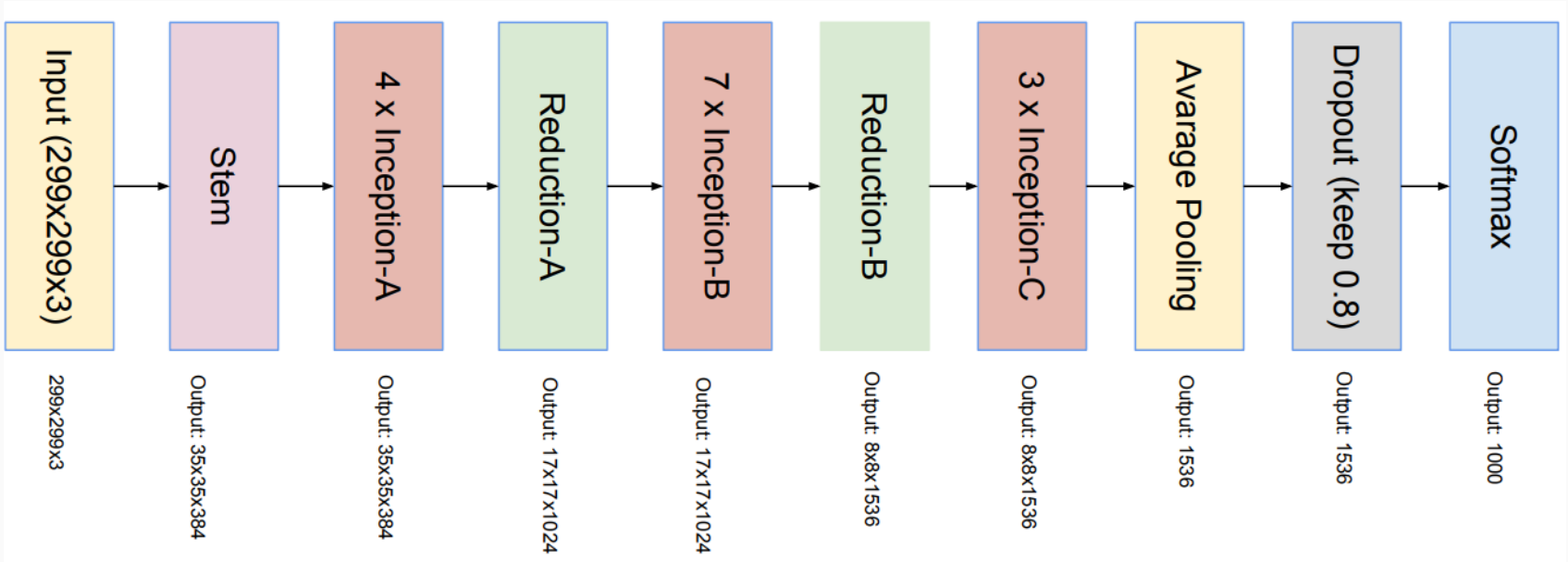
Inception v3 (antes conocida como Inception v2)

También se llevan a cabo las siguientes modificaciones:

- Reducciones en las últimas capas mediante convoluciones con stride 2.
- Se reduce el stride en las primeras dos capas o se elimina la primera capa de pooling para lidiar con imágenes de baja resolución.
- Label smoothing. Es decir, se modifican las etiquetas para que no sean 0 o 1, sino que se utilicen valores en un rango $[\epsilon, 1 - \epsilon]$. De esta forma se previene el *overfitting* al bajar la confianza del modelo.

Inception v4

La arquitectura Inception v4 es una variante sin conexiones residuales y con más módulos inception. El principal objetivo fue la reducción de complejidad respecto al Inception v3.



Arquitectura general de Inception v4

Inception v4

Además, junto a esta se propusieron redes Inception con conexiones residuales en lugar de las operaciones pooling de los bloques inception, debido al éxito de estas en la ResNet:

- Inception ResNet v1
- Inception ResNet v2

Estas redes difieren entre ellas principalmente en el *stem* de la red, manteniendo el bloque de reducción igual.

A pesar de la sustitución de las capas pooling en los bloques inception, se continuan utilizando en los bloques de reducción. Otra diferencia es que cada bloque inception tiene una convolución 1x1 sin activación a continuación, con el fin de aumentar el número de filtros para que coincida con la profundidad de la siguiente capa.

Uso de clasificadores avanzados en Keras

La librería de Keras está preparada con modelos preentrenados de las arquitecturas vistas.

Los siguientes enlaces llevan a la documentación de cada modelo:

- Inception v3
- InceptionResNet v2
- VGG

La siguiente entrada de keras contiene el listado entero de modelos preentrenados así como una breve guía de uso.

Referencias i

[1] Opengenus.
Ilsvrc results image.
[Online; accessed September, 2022].

[2] Asifullah Khan, Anabia Sohail, Umme Zahoora, and Aqsa Saeed Qureshi.
A survey of the recent architectures of deep convolutional neural networks.
Artificial intelligence review, 53(8):5455–5516, 2020.

[3] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner.
Gradient-based learning applied to document recognition.
Proceedings of the IEEE, 86(11):2278–2324, 1998.

Referencias ii

[4] Shipra Saxena (Analytics Vidhya).
Lenet image.
[Online; accessed September, 2022].

[5] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton.
Imagenet classification with deep convolutional neural networks.
Communications of the ACM, 60(6):84–90, 2017.

[6] Aqeel Anwar (Towards Data Science).
Alexnet image.
[Online; accessed September, 2022].

[7] Karen Simonyan and Andrew Zisserman.
Very deep convolutional networks for large-scale image recognition.
arXiv preprint arXiv:1409.1556, 2014.

Referencias iii

[8] user40847 (StackExchange).
Vgg16 image.
[Online; accessed September, 2022].

[9] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich.
Going deeper with convolutions.
In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9, 2015.