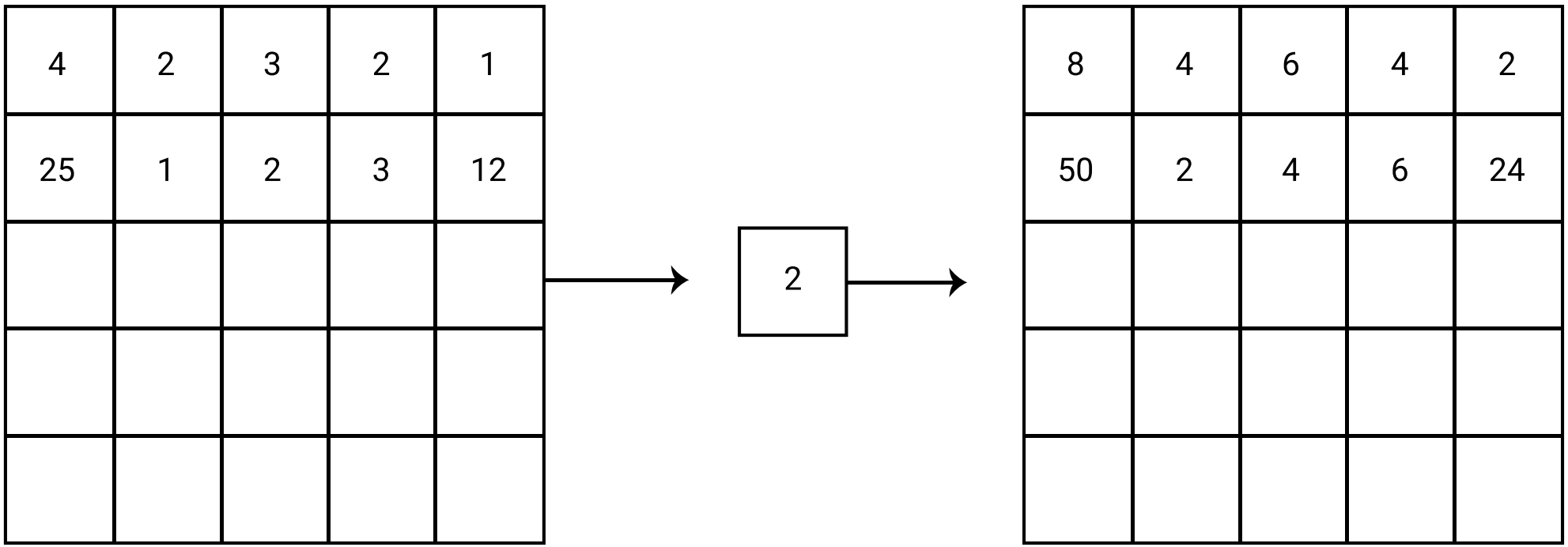
1x1 Convolution

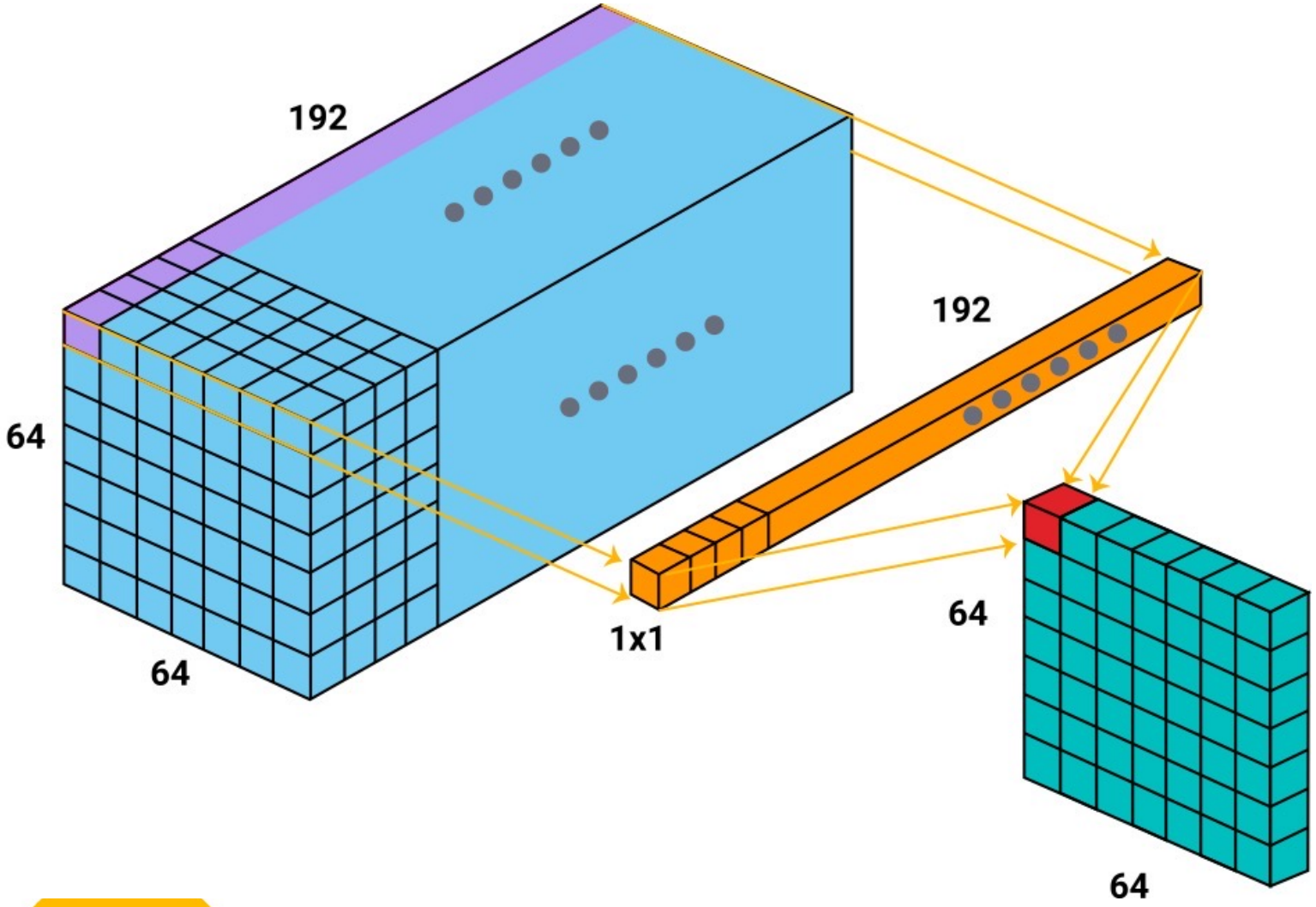
- Consiste in una convoluzione in cui la dimensione del kernel è pari a 1
- Equivale alla moltiplicazione per uno scalare dei dati di input



1x1 Convolution

- Usi principali:
  - Dimensionality reduction
  - Aggiunta di non linearità
  - Feature combination
- Uso comune: nell’Inception Block di GoogleNet

Dimensionality reduction



L'utilità delle 1x1 è legata al **numero di filtri** che vengono utilizzati durante il processo di convoluzione

Se l'input è WxHxD il volume di profondità di uscita verrà dato unicamente dal numero di filtri del kernel

[redacted] 1x1 CONVOLUTION

1x1 Convolution

- Le 1x1 quindi vengono usate per modificare il numero di canali di profondità in uscita
- Effetti:
  - Riduzione della complessità computazionale
  - Efficienza dei parametri
  - Abbassamento rischio overfit
  - Livelli «bottleneck»
  - Feature Extraction

1x1 Convolution vs. Pooling

	1x1	Pooling
Scopo	Feature combination e dimensionality reduction	Downsampling e quindi dim. reduction
Operazione	Convoluzione con il più piccolo receptive field	Max o media su receptive field più ampi
Output	Potenzialmente possono conservare più informazioni	«Riassumono» le info potenzialmente con perdita
Impatto sui parametri	Numero minimo di parametri	Non hanno parametri

Non Linearità

- Le 1x1 quindi vengono usate per **aggiungere non linearità ai filtri**
- Le conv 1x1 aggiungono non linearità perché sono **dei livelli convolutivi veri e propri**
  - Conv 1x1 di per sé è lineare
  - Dopo la conv 1x1 si applica la **funzione di attivazione** che aggiunge non linearità
- La non linearità in generale:
  - Aumenta il potere di rappresentazione di una rete
  - Aiuta contro l'overfit
  - Aiuta con la cattura di informazioni complesse

Feature combination

- Una conv 1x1 può essere vista come un **aggregatore** di canali
  - È sostanzialmente una **somma ponderata** degli output di un canale
- La feature combination è l'unione o fusione di informazioni provenienti da diverse feature map in un'unica feature map
- Di norma le feature composite sono più ricche e più informative delle feature di partenza

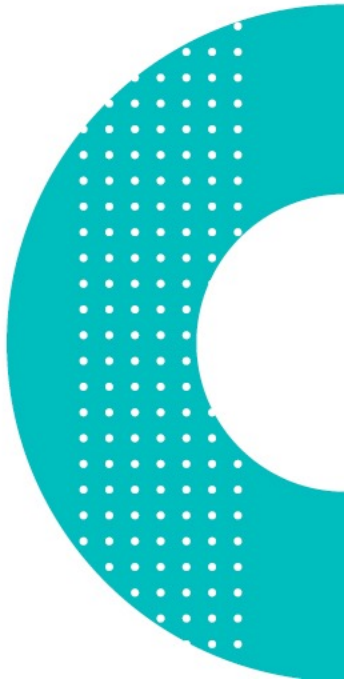
- Tutte queste caratteristiche sono state utilizzate nell'**Inception Block**

Inception Block

- Alla base dell'architettura "Inception" c'è il cosiddetto **Inception Block**
- Usato nella architettura GoogLeNet, vincitrice dell'ImageNet Large Scale Visual Recognition Challenge del 2014
- Idea:
  - Apprendere feature **locali** contemporaneamente a feature **globali**

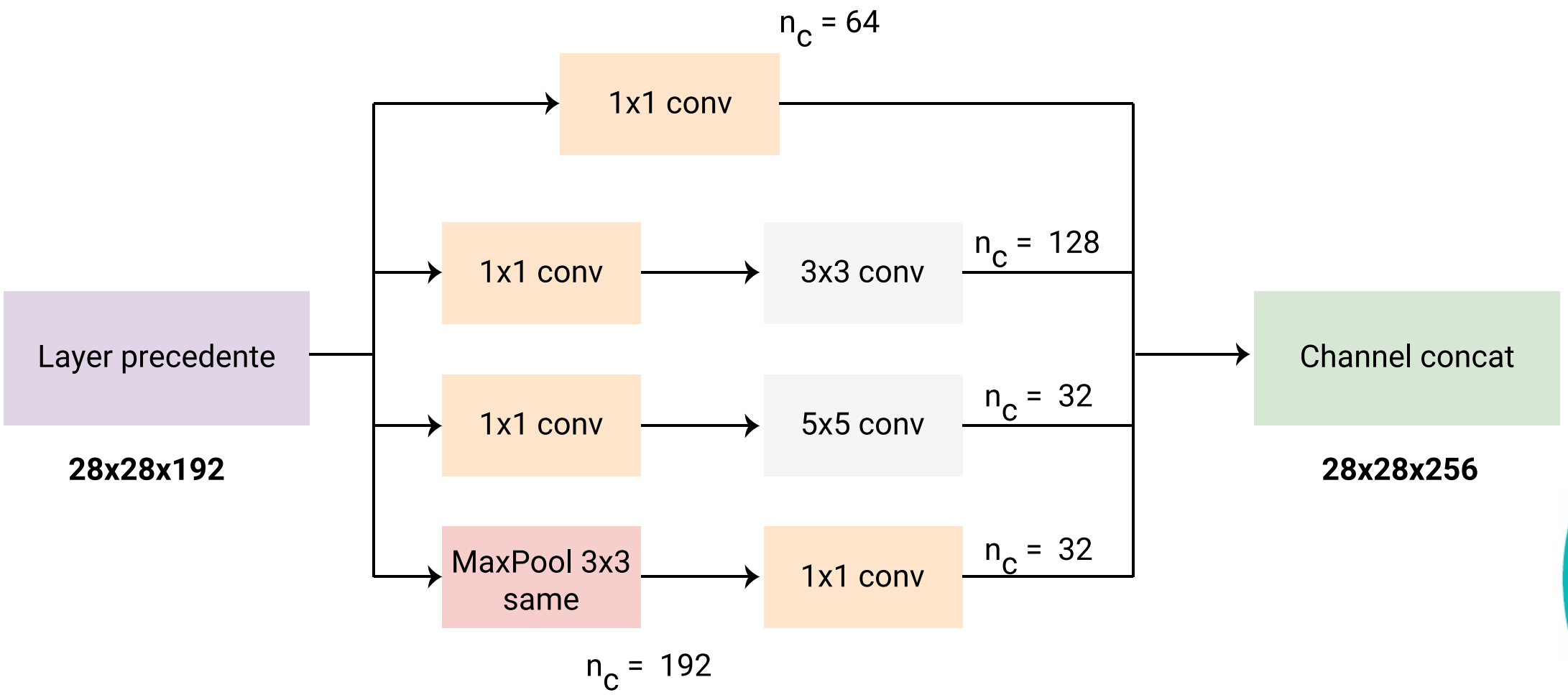
Inception Block

- Basato su una **parallelizzazione delle operazioni** e su una **concatenazione** di filtri in uscita da un certo livello di input
  - Convoluzione 1x1 con 64 filtri in uscita
  - Convoluzione same 3x3 con 128 filtri in uscita
  - Convoluzione same con 32 filtri 5x5 in uscita
  - MaxPool
- L'uscita è **28x28x256**
- Al posto di scegliere una convoluzione sola, le scegliamo tutte concatenandole dopo

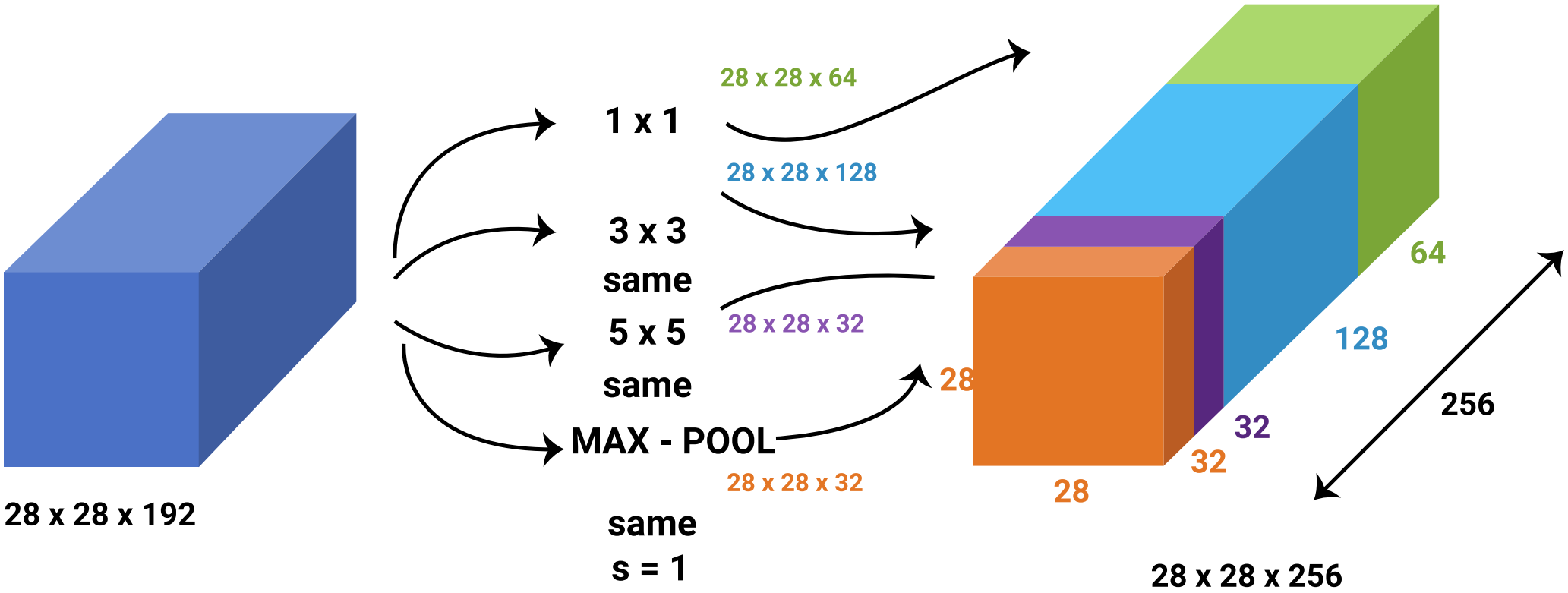


1x1 CONVOLUTION

Inception Block



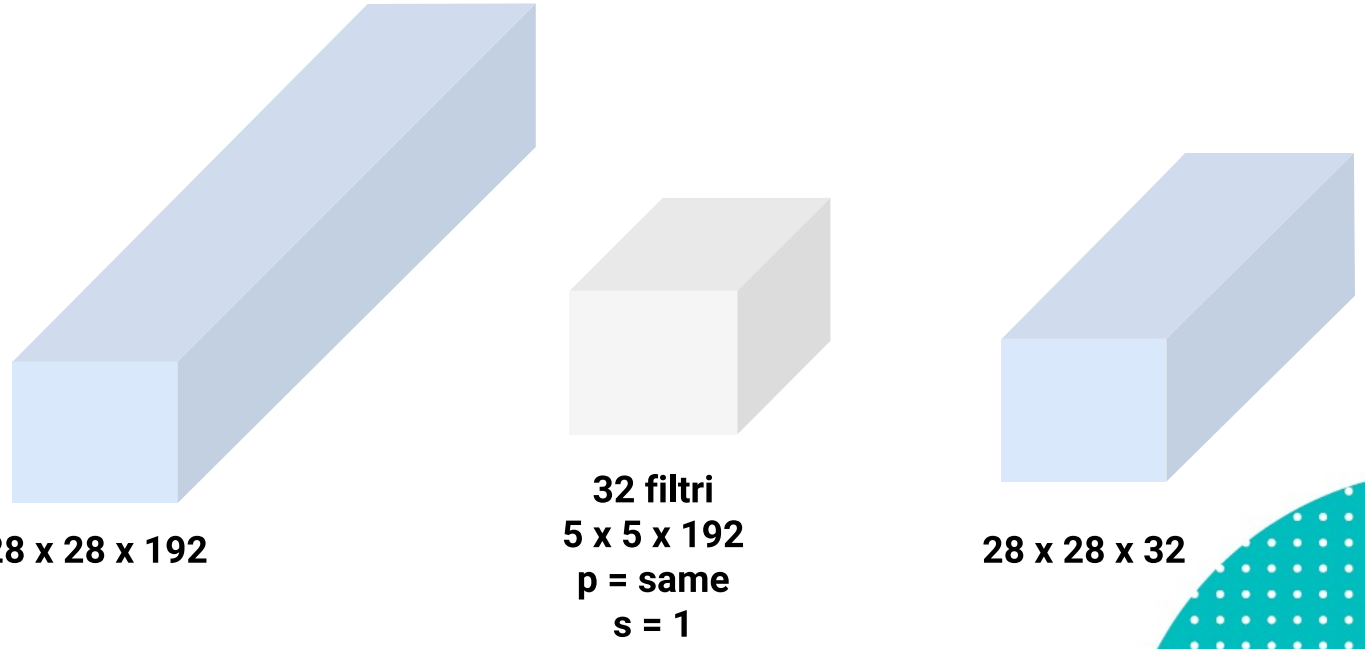
Inception Block



1x1 CONVOLUTION

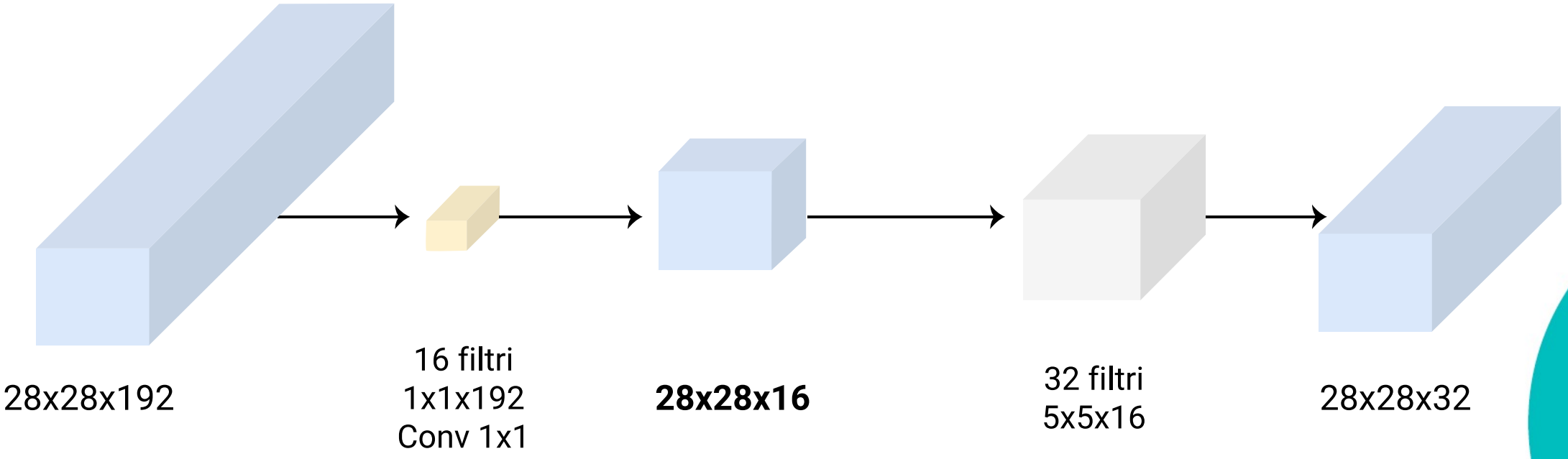
Inception Block

- Costo computazionale del **solo** blocco convolutivo 5x5 same
  - Input: 28x28x192
  - 32 filtri 5x5x192
  - Output: 28x28x32
- Num. Totale di moltiplicazioni da effettuare
  - 28x28x32x**5x5x192 =~120M**



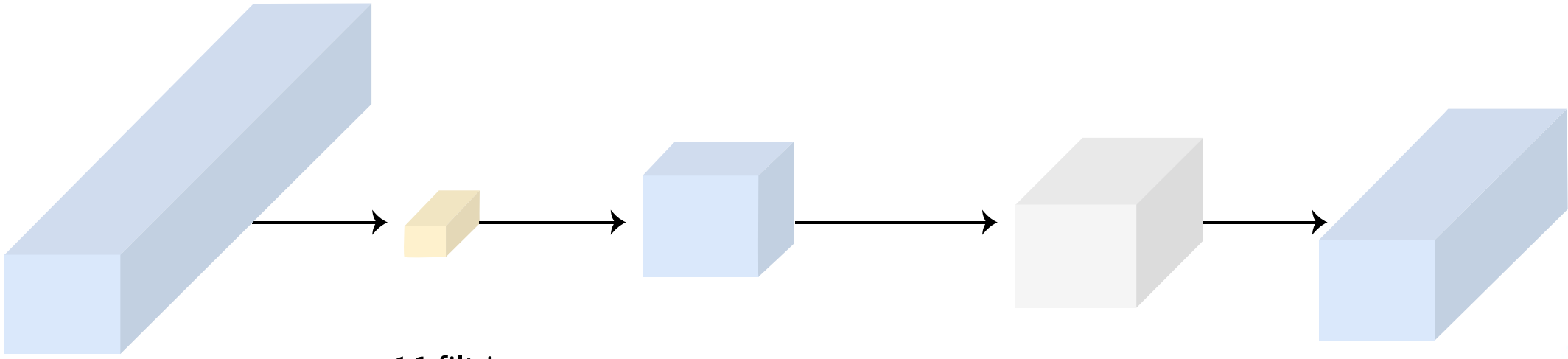
Inception Block

- Usando una convoluzione 1x1 (bottleneck):

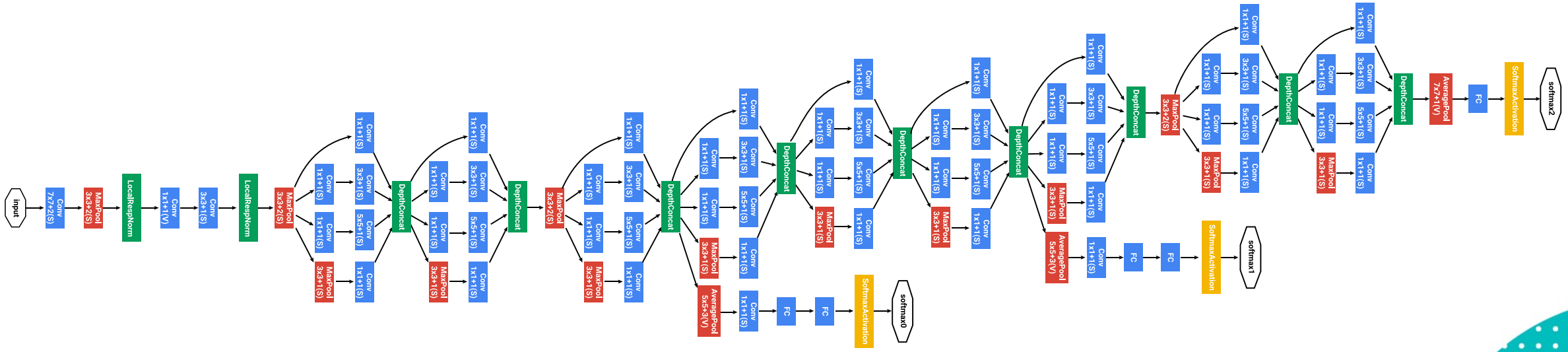


Inception Block

- Costo computazionale: 28x28x192x16 + 28x28x16x5x5x32 =~ **12M** (<<120M)



Inception Network



- Questa implementazione prende il nome di GoogLeNet (in onore a LeNet-5)
- Presenza di **classificatori ausiliari** nei cosiddetti «side branch»

[redacted] **1x1 CONVOLUTION**