**1)**A

2)B

3)B

4)B

5)A

6)B

7) D

8)D

9) B,C

10)A,B,C

11) Convolutional neural networks refer to a sub-category of neural networks: they, therefore, have all the characteristics of neural networks. However, CNN is specifically designed to process input images. Their architecture is then more specific: it is composed of two main blocks.

**The first block** makes the particularity of this type of neural network since it functions as a feature extractor. To do this, it performs template matching by applying convolution filtering operations. The first layer filters the image with several convolution kernels and returns “**feature maps**”, which are then normalized (with an activation function) and/or resized.

**The second block** is not characteristic of a CNN: it is in fact at the end of all the neural networks used for classification. The input vector values are transformed (with several linear combinations and activation functions) to return a new vector to the output.

The different layers of a CNNThere are four types of layers for a convolutional neural network: the **convolutional** layer, the **pooling** layer, the **ReLU correction** layer and the **fully-connected** layer.

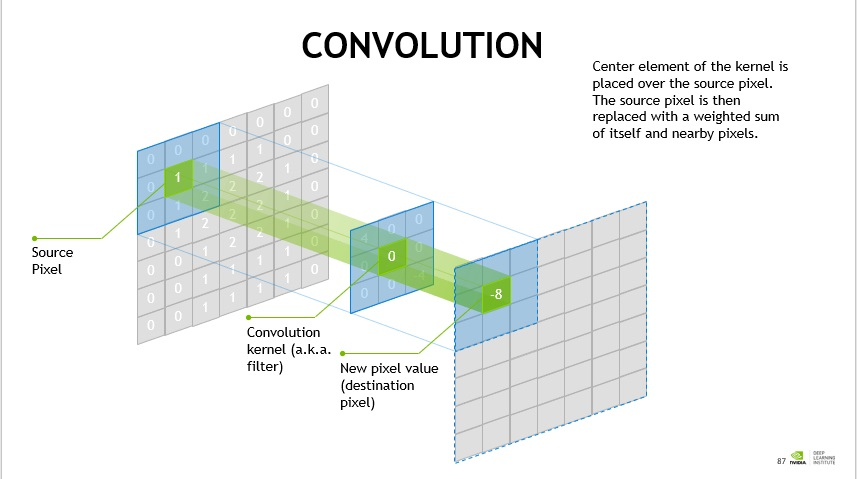
## 12) The convolutional layer

The convolutional layer is the key component of convolutional neural networks, and is always at least their first layer.

Its purpose is to detect the presence of a set of features in the images received as input. This is done by convolution filtering: the principle is to “drag” a window representing the feature on the image, and to calculate the convolution product between the feature and each portion of the scanned image. **A feature is then seen as a filter**: the two terms are equivalent in this context.

The convolutional layer thus receives several images as input, and calculates the convolution of each of them with each filter. The filters correspond exactly to the features we want to find in the images.

We get for each pair (image, filter) a **feature map**, which tells us where the features are in the image: the higher the value, the more the corresponding place in the image resembles the feature.



Convolutional layer

Unlike traditional methods, features are not pre-defined according to a particular formalism (for example SIFT), but learned by the network during the training phase! Filter kernels refer to the convolution layer weights. **They are initialized and then updated by backpropagation using gradient descent**.

13) **Average pooling** involves calculating the **average** for each patch of the feature map. This means that each 2×2 square of the feature map is down sampled to the **average** value in the square.

**Max pooling** selects the brighter pixels from the image. It is useful when the background of the image is dark and we are interested in only the lighter pixels of the image.

14) **Padding** is simply a process of adding layers of zeros to our input images so as to avoid the problems mentioned above. This prevents shrinking as, if p = number of layers of zeros added to the border of the image, then our (n x n) image becomes (n + 2p) x (n + 2p) image after **padding**.

There are couple of reasons padding is important:

1. It's easier to design networks if we preserve the height and width and don't have to worry too much about tensor dimensions when going from one layer to another because dimensions **will just "work"**.
2. It allows us to design **deeper networks**. Without padding, reduction in volume size would reduce too quickly.
3. Padding actually **improves performance by keeping information at the borders**.

15) **The convolutional layer**

Its purpose is to detect the presence of a set of features in the images received as input. This is done by convolution filtering: the principle is to “drag” a window representing the feature on the image, and to calculate the convolution product between the feature and each portion of the scanned image.

**The pooling layer**

This type of layer is often placed between two layers of convolution: it receives several feature maps and applies the pooling operation to each of them.

The pooling operation consists in **reducing the size** of the images while **preserving their important characteristics**.

We get in output the same number of feature maps as input, but these are much smaller.

The pooling layer **reduces the number of parameters and calculations in the network**. This improves the efficiency of the network and avoids over-learning.

**The ReLU correction layer**

ReLU (Rectified Linear Units) refers to the real non-linear function defined by *ReLU(x)=max(0,x)*.

The ReLU correction layer replaces all negative values received as inputs by zeros. It acts as an **activation function**.

**The fully-connected layer**

The fully-connected layer is always the last layer of a neural network, convolutional or not — so it is not characteristic of a CNN.

This type of layer receives an input vector and produces a new output vector. To do this, it applies a **linear combination** and **then possibly an activation function** to the input values received.

The last fully-connected layer classifies the image as an input to the network: it returns a vector of size N, where N is the number of classes in our image classification problem. Each element of the vector indicates the probability for the input image to belong to a class.

The convolutional neural network learns weight values in the same way as it learns the convolution layer filters: during the training phase, by **backpropagation of the gradient**.