### Deep learning theory

(Hinton G., 2006) specifically proposed the principle of deep learning in 2006. Deep learning's basic purpose is to set up a deep neural network to replicate the human brain's learning and interpretation process. Compared to traditional machine learning theories, the most significant difference in deep learning is the focus on supervised learning from the large data set by multi-layer neuron organization. Several deep learning architectures such as Deep Belief Networks (DBN) (Hinton, Osindero, & Teh, 2006), Convolutional Neural Networks (CNN) and Recurrent Neural Networks (RNN) (. Graves, et al., 2009) have been applied in recent years to fields such as computer vision (Ciresan, Meier, & Schmidhuber, 2012), speech recognition, natural language processing, audio recognition, and bioinformatics, and have been shown to deliver state-of-the-art results in these fields.

### CNN for RS image classification

CNN has obtained remarkable results in the classification of images, recognition and other vision tasks in deep learning techniques, and has the highest score in many visual databases such as ImageNet, Pattern Analysis, Statistical Modeling and Computational Learning Visual Microsoft Popular Objects in Context (MS-COCO) and Object Classes (PASCAL VOC). The basic structure of the standard CNN for image classification is stacks of "convolutional-pooling" layers as multi-scale extractors of features and subsequent numbers of fully connected layers as classifiers. A lot of research has appeared in recent years on CNN-based remote sensing image analysis. (Nguyen, 2013) proposed a satellite image classification method using a network of five layers and achieved a classification accuracy of more than 75%. For long-term route planning, (Wang, 2015) used a three-layer CNN structure and Finite State Machine (FSM) for road network extraction. (Castelluccio, Poggi, Sansone, & Verdoliva, 2015) explored the use of CNNs to identify remote sensing scenes semantically. (Hu, Xia, Hu, & Zhang, 2015) used a pre-trained CNN model to classify various scenes from high-resolution remote sensing imagery. (Zhou, Newsam, Li, & Shao, 2016) used CNN architecture as a high-resolution remote sensing image recovery (HRRSIR) deep-function extractor. A CNN-based architecture was suggested by (Mnih, 2013) to learn contextual features for aerial picture labelling on a large scale. The model creates a complex classification patch instead of outputting a single value image group. (Lagkvist, Kiselev, Alirezaie, & Loutﬁ, 2016) proposed a novel method of classification of remote sensing imaging based on CNN's for five groups (vegetation, soil, path, building and water), outperforming existing classification approaches. In addition to the CNN family approach, (Yuan, Lin, & Wang, 2016) used the Stacked AutoEncoder classifier for a classification experiment using the Manifold Ranking based Salient Band selection.

### CNN

CNN is considered to be one of the best techniques for studying image information and has shown state-of-the-art findings on image recognition, segmentation, identification and retrieval related tasks (Ciresan, Meier, & Schmidhuber, 2012). In industry, companies such as AT&T, Google, Microsoft, Facebook and NEC have developed active research groups to explore new CNN architectures (L. Deng, 2013). At present, most of the frontrunners of image processing competitions use deep CNN-based models.

Many improvements have been made to CNN's learning approach and infrastructure to make CNN applicable to large and complex problems. Such inventions can be defined as refining conditions, regularizing, reformulating systems, etc. Nevertheless, after AlexNet's outstanding success on the ImageNet dataset (A. Krizhevsky, 2012), it is noted that CNN-based applications were prevalent. As a result, major developments have been proposed on CNN since 2012 and were mainly due to processing unit redesign and new block design. Furthermore, Zeiler and Fergus (Fergus, 2013) have introduced the concept of a layer-wise representation of functionality that has changed the trend towards the abstraction of features at low spatial resolution in deep architectures such as VGG (Zisserman, 2015). On the other hand, the Google group has introduced an interesting idea of splitting, transforming, and merging, and the corresponding block is known as the inception block. For the very first time, the invention block provided the idea of branching within a framework, which enables features to be abstracted from various spatial scales (K. He, 2015). The concept of skip connections introduced by ResNet for the training of deep CNNs became famous in 2015 and was subsequently used by most of the following networks, such as Inception-ResNet, WideResNet, ResNext, etc. (Komodaki, 2015).

### Basic CNN Components

Today, CNN is considered to be the most widely used ML technique, particularly in vision-related applications. CNN recently demonstrated state-of-the-art results in various ML applications. Because of CNN possesses both a successful extraction capability and a great ability to discriminate, it is therefore used in the ML system; it is mostly used for extraction and classification of features.

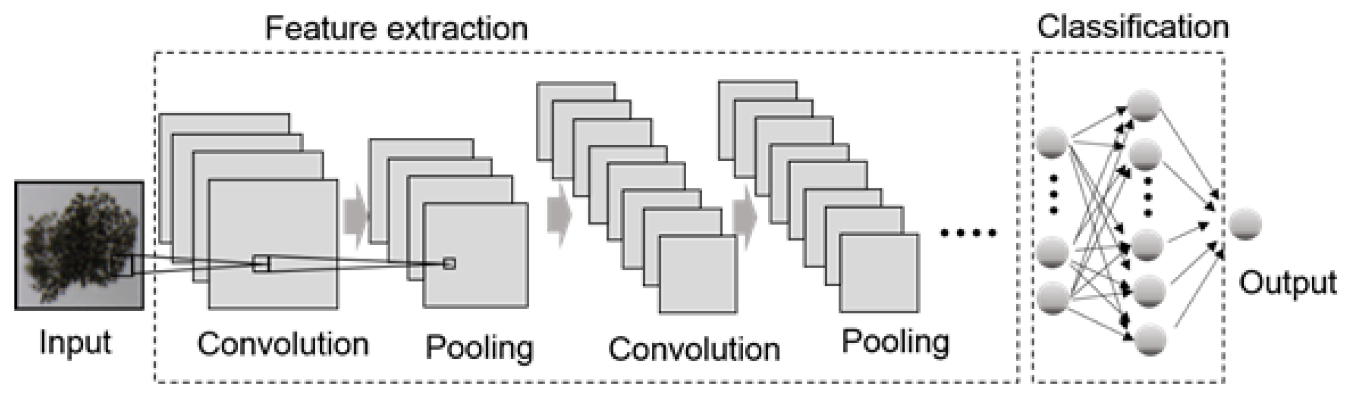


Figure 2: An example of the design of a CNN.

Figure 2 illustrates the CNN architecture generally consists of alternating layers of convolution and pooling accompanied by one or more fully connected layers at the top. Sometimes Fully connected layer is replaced by a global average pooling layer. In addition to the several learning stages, the CNN output optimisation is also assisted by various regulatory units such as batch normalisation and dropout (Bouvrie, 2006). The configuration of CNN components plays a key role in designing new architectures and thus achieving enhanced performance. This section discusses briefly the role of these components in CNN architecture.

#### Convolutional Layer

The convolutional layer consists of a collection of kernels of convolution (each neuron functions as a kernel). Such kernels are bound to a small area of the image called a field of reception. It operates by splitting the image into small blocks (receptive fields) and combining them with a specific set of weights (multiplying filter elements with corresponding receptive field elements) (Bouvrie, 2006). The process of Convolution will convey the following:

|  |  |  |
| --- | --- | --- |
|  |  | (11) |

Where the input image is represented by , x, y displays the spatial localization, and represents the th convolutionary kernel of the Kth row. The division of the image into small blocks helps to extract locally correlated pixel values. Also known as the feature motif is this locally aggregated information. A different set of image features are extracted with the same set of weights by sliding the convolution kernel over the entire image. This convolution operation 7 weight sharing feature makes the CNN parameter more efficient than fully connected networks. Convolution operation can also be categorized into different types based on filter type and size, padding type, and convolution direction (Bouvrie, 2006). Furthermore, if the kernel is symmetric, it becomes a correlation operation (Ian Goodfellow, 2015).

#### Pooling Layer

Feature motifs that can occur at different locations in the image as the production of a convolution cycle. Once characteristics are extracted, their exact location becomes less important as long as their approximate position is preserved relative to others. It is an interesting local process to pool or downsample like convolution. It summarizes similar information in the receptive field neighbourhood and generates the dominant answer in this local area (Bouvrie, 2006).

|  |  |  |
| --- | --- | --- |
|  |  | (12) |

Equation (12) indicates the pooling operation in which represents the th output feature map, shows the th input feature map, whereas determines the form of the pooling operation. Using the pooling process helps generate a combination of features invariant to translation changes and slight distortions (D. Scherer, 2010). Reduced functional map size to invariant feature set governs not only network complexity but also allows generalization by overfitting reduction. CNN (K. He, Spatial pyramid pooling in deep convolutional networks for visual recognition, 2015) uses various types of pooling methods, such as max, average, L2, overlap, spatial pyramid pooling etc.

#### Activation Function

The activation mechanism acts as a tool for decision making and helps to understand a complex pattern. It can speed up the learning process by choosing an appropriate activation function. The activation function is defined in equation (13) for a converted feature map.

|  |  |  |
| --- | --- | --- |
|  |  | (13) |

In the above equation, is an output of a convolution operation assigned to the activation function; adding non-linearity and returning a transformed output for th layer. Numerous activation functions such as sigmoid, tanh, maxout, ReLU, and variants of ReLU such as leaky ReLU, ELU, and PReLU (al, 2018; B. Xu, 2015; LeCun, 2007) are used in literature to inculcate nonlinear combination of features. ReLU and its variants are however preferred over other activations as it helps to overcome the problem of the vanishing gradient (Hochreiter, 1998).

#### Batch Normalization

Batch normalization is used within feature maps to address issues related to internal covariance change. The internal covariance shift is a change in the distribution of values of hidden units that delays the convergence (forcing the learning rate to a low value) and requires careful parameter initialization. Batch normalization for a transformed feature map is shown in equation (14).

|  |  |  |
| --- | --- | --- |
|  |  | (14) |

In equation (14), represents a standardized feature map, are the input feature map, and mean and variance of a feature map respectively for a mini-batch. Batch normalization unifies the map value distribution by taking it to zero mean and unit variance (Bouvrie, 2006). It also smoothes the gradient flow and serves as a controlling factor, thereby helping to improve the network's generalization.

#### Dropout

Dropout implements network regularisation, which eventually increases generalisation by automatically skipping a certain likelihood of some units or connections. Multiple connections in NNs that know a non-linear relationship are often co-adapted, leading to overfitting (G. E. Hinton, 2012). A random drop in some links or units creates multiple thin network architectures, and finally, one representative network with small weights is chosen. This selected architecture is then considered to be an approximation of all the networks proposed (N. Srivastava, 2014).

#### Fully Connected Layer

A fully connected layer is typically used for classification purposes at the end of the network. It is a global operation, unlike pooling and convolution. It takes data from the previous layer and the output of all the previous layers is evaluated globally (K. He, Going deeper with convolutions, 2015). This allows a non-linear combination of selected features used to classify data (Wang W. R., 2016).

### Data Augmentation

The goal of Data Augmentation is to extend the dataset to resolve data representation limitations and reduce the problem of overfitting. It increases the efficiency of the model and avoids unbalanced learning. Data increases were commonly reported in many areas to mitigate data scarcity (Ding, 2016). The technique also eliminates over-fitting, which can occur when adding model-related data instances and over-complex models (Wang J. P., 2017). Sadly, there was not much work recorded with data increase techniques in the satellite image domain.

The most widely used image transformations are in satellite image clipping, spinning, flipping, moving, sorting and translating (Ding, 2016; Wang J. P., 2017), while materials were not available in the sense of satellite image super-resolution to form a strong opinion on which augmentation techniques were more useful.

### VGG

Simonyan et al. proposed a simple and effective design concept for CNN architectures with the successful use of CNNs for image recognition. They had a modular layer template architecture called VGG (Zisserman, 2015). VGG was made up of 16 layers deep (A. Krizhevsky, 2012). Based on these results, the VGG replaces 11x11 and 5x5 filters with a 3x3 filters layer and has shown experimentally that the effects of the large filter are triggered by the simultaneous positioning of 3x3 filters (receptive and smaller-size (5x5 and 7x7)). By decreasing the number of parameters, the use of small filters offers additional advantages of low code complexity. VGG regulates network complexity by putting a 1x1 interlayer between convolutional layers, which also learns a linear combination of the resulting characteristic maps. For tuning the network, the maximum concentration is placed after the convolutional layer and spatial resolution is maintained by the padding (F. J. Huang, 2007). For both the classification of images and localization problems, VGG showed good results. The main limitation associated with VGG was the high computational cost. VGG suffered from a high computational burden due to the use of approximately 140 million parameters, even with the use of small size filters.

Since this research used very few training images for classification (Shuying Liu, 2015) concluded that the very deep CNN could be used to ﬁt small datasets with simple and proper modiﬁcations and don’t need to re-design speciﬁc small networks. We believe that if a model is strong enough to ﬁt a large dataset, it can also ﬁt a small one.

### TensorFlow-GPU(v1.14.0)

According to the official documentation, TensorFlow™ is an open-source software library for numerical computation using data flow graphs. Nodes in the graph represent mathematical operations, while the graph edges represent the multidimensional data arrays (tensors) communicated between them. The flexible architecture allows you to deploy computation to one or more CPUs or GPUs in a desktop, server, or mobile device with a single API. TensorFlow was originally developed by researchers and engineers working on the Google Brain Team within Google's Machine Intelligence research organization for the purposes of conducting machine learning and deep neural networks research, but the system is general enough to be applicable in a wide variety of other domains as well (Google, 2019)

### Tensorboard(v1.14.0)

The official Tensorflow documentation says that the computations that use TensorFlow for - like training a massive deep neural network - can be complex and confusing. To make it easier to understand, debug, and optimize TensorFlow programs, TensorFlow developers have included a suite of visualization tools called TensorBoard. TensorBoard can be used to visualize TensorFlow graphs, plot quantitative metrics about the execution of graphs, and show additional data like images that pass through it. The following figure shows a fully configured Tensorboard. (Google, 2019)

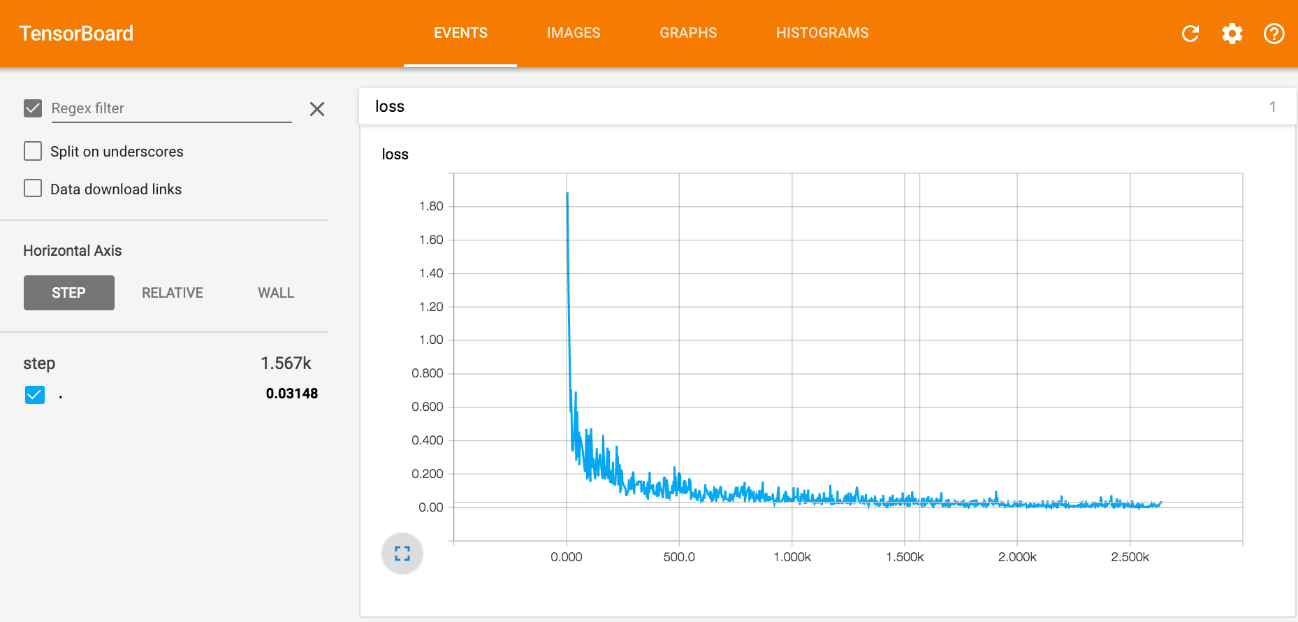


Figure : An instance of a fully configured Tensorboard

### Keras(v2.3.1)

The official document of Keras says that Keras is a Python-written high-level neural network Interface(kind of high-level API) that can run on top of TensorFlow, CNTK, or Theano. It was designed to allow rapid innovation with a focus and also it is important to perform good research to be able to go from idea to conclusion with the least possible time.

Use Keras if you need a deep learning library that:

* Enables easy and quick prototyping (via user-friendliness, modularity and extensibility).
* Supports both convolutionary and recurrent networks, as well as both varieties.
* Runs seamlessly on CPU and GPU.

(Team, 2019)

### Cuda(v10.0)

According to the official documentation, CUDA is a parallel computing platform and programming model developed by NVIDIA for general computing on graphical processing units (GPUs). With CUDA, developers are able to dramatically speed up computing applications by harnessing the power of GPUs.

In GPU-accelerated applications, the sequential part of the workload runs on the CPU which is optimized for single-threaded performance while the compute-intensive portion of the application runs on thousands of GPU cores in parallel. When using CUDA, developers program in popular languages such as C, C++, Fortran, Python and MATLAB and express parallelism through extensions in the form of a few basic keywords.

The CUDA Toolkit from NVIDIA provides everything developers need to develop GPU-accelerated applications. The CUDA Toolkit includes GPU-accelerated libraries, a compiler, development tools and the CUDA runtime. (Nvidia, 2019).

Although this is good for the SVM, bad for training the CNN model. So we use the data augmentation approach to increase the number of images per class (Ding, 2016; Wang J. P., 2017). There are 550 images per class were acquired. This is a considerable large dataset compared to the previous dataset. This dataset split as 80% for training 10% for validating and 10% for testing for the CNN model.