Problem Overview

Pneumonia is an inflammation of the lungs which can be caused by either a fungal, bacterial or viral infection. Of these bacterial or viral infection are the more common cause0source. According to the *The European Lung white book; Respiratory Health and Disease in Europe* it had a mortality rate in Ireland of 32.96 per 100,000 in 2011[1], which, when compared to the total death rate in Ireland for 2011 of 6.2 per 1,000, means that pneumonia was responsible for 5.3% of deaths in Ireland in 2011. 0More prevalent in children and elderly.

There are a number of methods to reliably rule in or out pneumonia when diagnosing a patient, such as the lack of certain symptoms or the presence of rates and bronchial breathing sounds, but chest radiography is generally considered the best method of confirming a pneumonia diagnosis[2]. There has been a great amount of research into the use of deep learning in medical diagnosis in recent years, in particular for use image-based diagnosis, such as MRI, CAT or X-Rays[3]. 0Examples of ML for diagnosis. These models are often able to achieve comparable, or sometimes higher, detection rates of these diseases compared to doctors, making them very useful tools.

This example problem aims to demonstrate how a machine learning model could be developed for detecting pneumonia from chest x-rays. It uses chest x-rays from the *Labeled Optical Coherence Tomography (OCT) and Chest X-Ray Images for Classification* dataset[4][1] produced by Daniel Kermany, Kang Zhang and Michael Goldbaumusing and is used under the Creative Commons *CC BY 4.0* license. This dataset contains 5856 images split into 3 directories; a training directory of 5216 images; a testing directory containing 624 images; and a validation directory containing 16 images. Pneumonia is prevalent in around 75% of the training presentations, and 50% of the test presentations. A number of deep

[1] Version 2

learning methods will be explored, but only convolutional neural network (CNNs) models will be used, as almost all real world models for image-based medical diagnosis use CNNs[3].

Convolutional Neural Networks

Convolutional neural networks (CNNs) are a class of neural networks which use convolutional layers to extract translation invariant *features* from an *feature map*, which can be any matrix $m \in \mathbb{R}^{w \times h \times d}$[5], such as an image. These features can then be passed as a vector to a neural network and the neural network can then be trained as usual. CNN classification models can generally be broken up into two distinct blocks of layers; a set of layers for feature extraction, and a set of layers for classification.

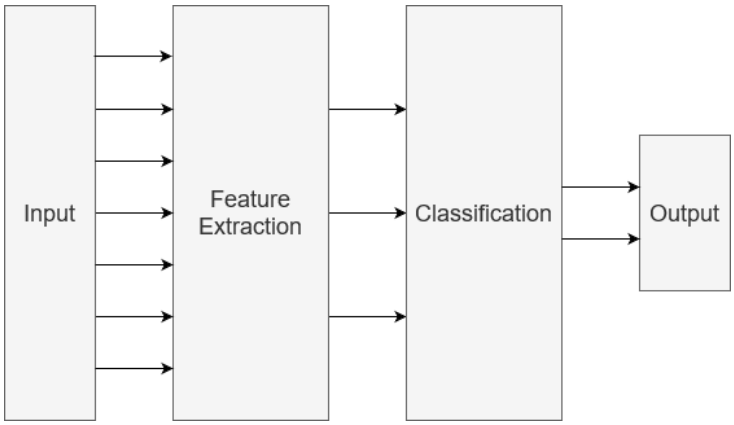

Figure 1: Block diagram of typical CNN

Despite being shown to be effective at solving machine vision tasks and fully trainable by back-propagation since 1989[6], it was not until as recently as 2012, when a CNN achieved a top-5 test error rate of 15.3%, compared to next highest of 26.3% in the ImageNet challenge[7] that they were considered state-of-the-art.

There are three types of layer typically found in the feature extraction block of a CNN model; Convolutional layers, pooling layers, and normalization layers[5]. Keras[8] has a number of built in implementations of these layers to allow their use in models.

Convolutional Layers

Similar to how perceptrons were designed to approximate the neurons in the brain, units in convolutional layers were designed to replicate the cells found in the visual cortex, which have a receptive field, and respond regions of the visual field[9]. To do this, convolutional layers use a number, n, of filters as their base units, which when convolved with an input feature map produce n output feature maps. The filters, commonly called kernels, are matrices of weights $w \in \mathbb{R}$ of size $k_w \times k_h$ where $k_w, k_h \in \mathbb{N}_{\geq 1}$[5]. This creates a number of hyperparameters needed to define a convolutional layer;

- n, the number of filters,
- (k_w, k_h) the shape of the filters, typically square,
- the activation function for the layer, which should be non-linear (similar to a normal perceptron layer),
- the stride $s \in \mathbb{N}_{\geq 1}$, which defines how much to move the kernel by when doing the convolutional,
- and the padding, which is used to determine the values convolved with the filter when it overlaps the edges of the image.

Keras offers a number of implementations of convolutional layers in the `keras.layers` module, covering several subtypes of convolutional layer, and with a number of arguments to set the hyperparmeters mentioned above;

- Basic convolution; basic convolutional layers with various shapes of filter, such as 1D, 2D and 3D are offered by `Conv1D`, `Conv2D` and `Conv3D` respectively.

- Depthwise separable convolution; when dealing with images with multiple channels, such as an RGB image, the number of multiplications done during convolution can get very large. To reduce the number of multiplications done, depthwise convolution can be done instead, convolving each channel of the image separably, and then the channels can be combined using a $1 \times 1 \times d$ kernel. This produces the same result as a regular convolution, but requires far fewer multiplications, and also fewer parameters[10]. This is implemented in the keras layers `SeparableConv1D` and `SeparableConv2D`.

- Depthwise convolution; if only a depthwise convolution is desired, i.e. a convolution on each input channel individually without combining afterwards, then a `DepthwiseConv2D` layer can be used.

Pooling Layers

Pooling is used to reduce the size of a feature map, while trying to retain the features. This is done by creating a summary of $p \times p$ areas of the image. The summary is created by applying a function to the pooled area, commonly used functions being; max pooling, taking the maximum value in the area; average pooling, taking the mean of an area; l_2 pooling, which takes the l_2 norm of the pooled area; and stochastic pooling, which selects a value for each area using its activation value to compute a probability.[5].

Normalization Layers

Dataset

0Images from dataset, Examples of pneumonia vs normal, etc.

Explored Solutions

While trying to develop a suitable model for this problem, a number of solutions were looked at, each of increasing complexity0maybe change, but each achieving better results.

Simple CNN

The first solution looked at was a simple convolutional neural network, based on 0similar model on kaggle. This was composed of 3 convolutional layers, with a 3×3 kernel size, and rectified linear activation function, each followed by a max-pooling layer, with a 2×2 pool size. After these convolutional layers a dense layer of 64 neurons, again using the rectified linear activation function, before a single neuron output layer, using a sigmoid activation function to create a binary classifier.

0Add some images

0Add training info/accuracy

0Comments

Image Augmentation

Image augmentation is the 0find word, technique? of performing transformations on the training images, to try stop the model from learning the *'noise'* in

the dataset, and instead learn the desired features0source. The hope being that by applying semi-random augmentations to the images before they are shown to the system, undesired information will become more random and the system will learn less about it. The Keras *ImageDataGenerator* has a number of arguments that can be used to apply transformations to images as they are presented, such as:

- Rotations; An integer can be passed to set the limit in degrees for random rotations to apply to the image using the `rotation_range` keyword argument
- Shifts; The image can be shifted vertically or horizontally by a random number of pixels using the `width_shift_range` and `height_shift_range` keyword arguments.
- Mirroring; The image may 50% of the time be mirrored around either the vertical or horizontal axis using the `horizontal_flip` and `vertical_flip` keyword arguments.
- 0maybe add in shear, etc.

By using these, the performance of the simple CNN described previously can be marginally 0(?) improved.
For this example problem, both vertical and horizontal shifts were applied, as well as mirroring across the vertical axis. These transformations were chosen as x-rays may not be centred, so one should still create a plausible input. Similarly as there is no difference between pneumonia in the left and the right lung, flipping the image vertically should not matter. 0info about this training

Transfer Learning

Bibliography

[1] *The european lung white book; respiratory health and disease in europe*, https://www.erswhitebook.org/chapters/acute-lower-respiratory-infections/, Accessed: 2020-02-03.

[2] R. R. Watkins and T. L. Lemonovich, "Diagnosis and management of community-acquired pneumonia in adults," *American family physician*, vol. 83, no. 11, pp. 1299–1306, 2011.

[3] M. Bakator and D. Radosav, "Deep learning and medical diagnosis: A review of literature," *Multimodal Technologies and Interaction*, vol. 2, no. 3, p. 47, 2018.

[4] D. Kermany, K. Zhang, and M. Goldbaum, "Labeled optical coherence tomography (oct) and chest x-ray images for classification," *Mendeley data*, vol. 2, 2018.

[5] M. Thoma, "Analysis and optimization of convolutional neural networkarchitectures," Masters?s Thesis, Karlsruhe Institute of Technology, Karlsruhe, Germany, Jun. 2017. [Online]. Available: https://martin-thoma.com/msthesis/.

[6] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel, "Backpropagation applied to handwritten zip code recognition," *Neural computation*, vol. 1, no. 4, pp. 541–551, 1989.

[7] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," pp. 1097–1105, 2012.

[8] F. Chollet *et al.*, *Keras*, `https://keras.io`, 2015.

[9] D. H. Hubel and T. N. Wiesel, "Receptive fields and functional architecture of monkey striate cortex," *The Journal of physiology*, vol. 195, no. 1, pp. 215–243, 1968.

[10] F. Chollet, "Xception: Deep learning with depthwise separable convolutions," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 1251–1258.