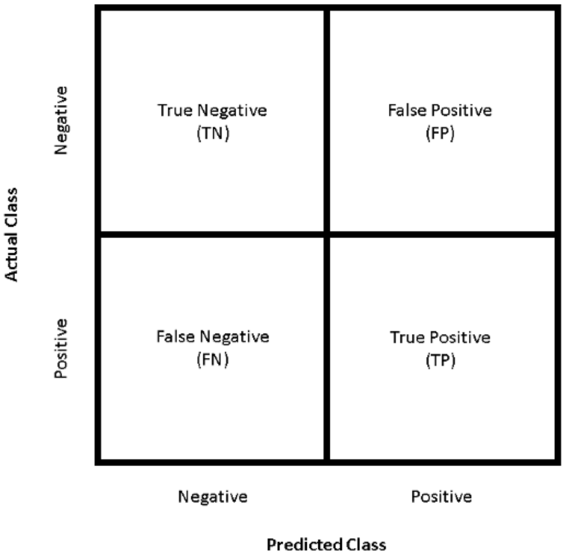
Interpretable Convnet for Disease Diagnosis

Introduction

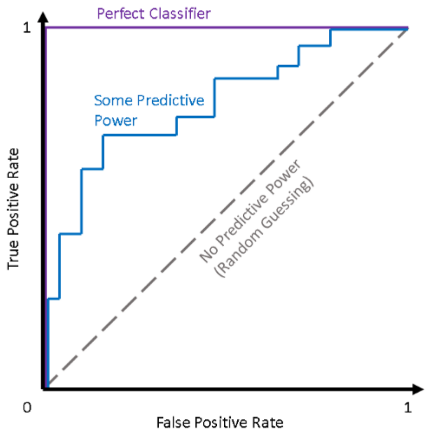
1. How it works

1.1 Glossary

- Neural Network: A common Machine Learning method which utilises labelled data. A model trained on the data could give predictions on unseen inputs (in this specific case classification of images). Neural Nets consist of specialised layers which are themselves built upon neurons that do funny mathematics such as a function or just basic arithmetics. The field is highly based on linear algebra.
- AUC (Area Under Curve): In some datasets, discrete classes could be greatly imbalanced in quantity. This is an especially dangerous pitfall since it could lead to the false determination of a model's quality, if the metric is accuracy. A cancer prediction model is effectively worthless if it has an accuracy of 99%, if 99% of the dataset consists of negative cases (in which the model could basically specify that *all* cases are negative and still receive a "good" result). AUC is an alternative metric to solve this issue.



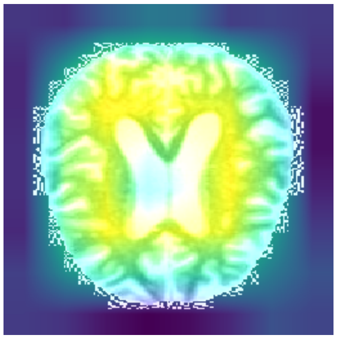
		Predicted Class	
		Negative	Positive
Actual Class	Negative	True Negative (TN)	False Positive (FP)
	Positive	False Negative (FN)	True Positive (TP)



  Accuracy is measured by $\frac{TP+TN}{TP+FP+TN+FN}$ while AUC is based on sensitivity and specificity, in other words $\frac{TP}{TP+FN}$ and $\frac{FP}{FP+TN}$. This would result in the measurement of the model's ability to distinguish between a positive or a negative case being contained to one specific class, which eliminates the problem of skewed data. In multiclass predictions, AUC is simply measured for every class, then averaged out.

Model quality will then be portrayed by plotting sensitivity and specificity at every classification threshold (a threshold of 80% means the model will return a negative if it has 75% confidence on an input) on a graph. AUC is the total area under the curve, with perfect AUC = 1, random guessing AUC = 0.5, and classification models falling somewhere in between.

- GradCam (Gradient Camera): An algorithm which visualises the significance of image regions to the final classification. The algorithm can be used to judge the model and interpret its predictions. As a result, it could localise regions of medical interest (brain section with shrinkage, lung section with infection…) or determine unwanted artefacts.



1.2 Methodology

The process could be divided into data preparations and processing, model training, and model evaluation.

Data prep

The whole data consists of labelled images (224px x 224px). All datasets were collected from verified sources (this statement has yet to be verified), cleaned, and standardised. Each respective dataset was divided into a training and validation set (80%) and a test set (20%).

The skin cancer dataset was collected from the ISIC Challenge — an annual AI based cancer classification contest. All images were directed to the skin lesion with (or without) a malignant tumour. Two classes including **Benign - 52.1%** and **Malignant - 47.9%** sum up to around 12500 images.

The Alzheimer's dataset was collected from OASIS and ADNI — labs with the purpose of brain MRI[1] research, and who have kindly open sourced their data to kaggle. Three classes including **AD - Alzheimer's Disease - 18%**, **MCI - Mild Cognitive Impairment - 41.8**, and **CN - Cognitively Normal - 40.2%** sum up to around 11500 images.

The Covid-19 and pneumonia dataset was collected from Kaggle — a data science platform by our reverend father google whom we shall bleed for in the name of the shattered one. All images were X-Rays directed at the chest cavity, specifically the lungs. Three classes including **Covid - 30%**, **Pneumonia - 42.7%**, and **Normal - 27.4%** sum up to around 16700 images.

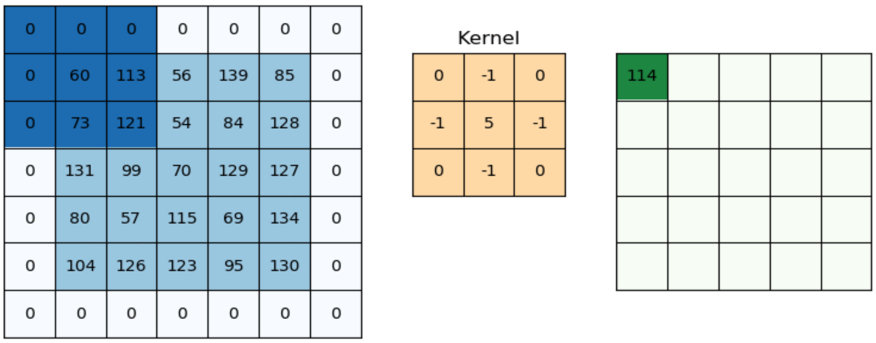
Model architecture

The model is called a Convolutional Neural Network. Quite convoluting

Main layer models:

- Conv2D: Kernels with parameters that are trained to extract features from the images. With many training iterations they can generalise and effectively spot out characteristics of a specific disease. All kernels were 3px x 3px in size, but increased linearly in quantity as the model gets deeper in order to extract increasingly complex features (first layer may extract particular features such as single points or lines but deeper layers extract more abstract features like curves and even rough outlines). In addition, every image is 'padded' with a layer of 0 value pixels in order to allow the kernel to iterate through every original pixel equally.

[1] Magnetic Resonance Imaging



0	0	0	0	0	0	0
0	60	113	56	139	85	0
0	73	121	54	84	128	0
0	131	99	70	129	127	0
0	80	57	115	69	134	0
0	104	126	123	95	130	0
0	0	0	0	0	0	0

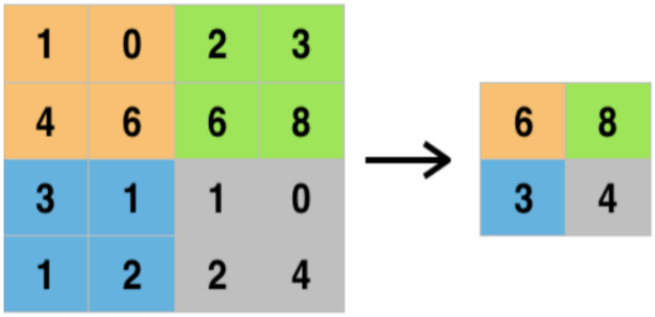
Kernel

0	-1	0
-1	5	-1
0	-1	0

114				

Conv2D extracting an image. The kernel values are tuned in training.

- MaxPooling: This layer outputs the largest pixel value in a 2px x 2px region. This decreases the dimensions of the data gradually and thus saves memory. More importantly, it eliminates many trivial and detailed pixels that do not contribute to the general patterns of the image, which prevents the model from overfitting, and improves its accuracy.



1	0	2	3
4	6	6	8
3	1	1	0
1	2	2	4

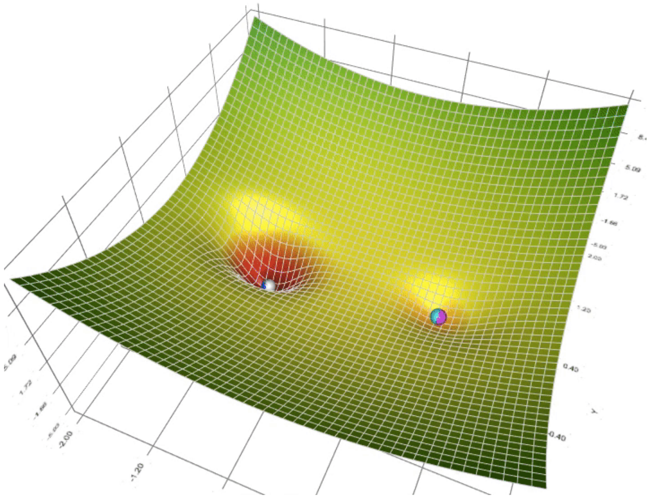
6	8
3	4

MaxPooling extracting a 4x4 to a 2x2

- BatchNormalization: This layer normalises the data by redefining the mean and standard deviation. Helps with regularisation — the model becomes more generalised.
- Dense: This layer consists of neurons that are each fully connected to every neuron from the previous layer. They use the features extracted from Conv2D layers to do the actual classifications like a traditional Neural Network.

Some other training algorithms:

- Loss function: To measure the differentiations between the model's predictions and the actual classes. Respectively Categorical Cross Entropy (for multiclasses) or Binary Cross Entropy (for binary classes), depending on the data.
- Optimizer: Function to minimise Loss. I used Stochastic Gradient Descent, which uses calculus to progressively find the local minima, akin to walking down a hill. (it's actually quite simple, find the gradient ⇒ determine whether to step this way or that way, and how far)



Gradient Descent

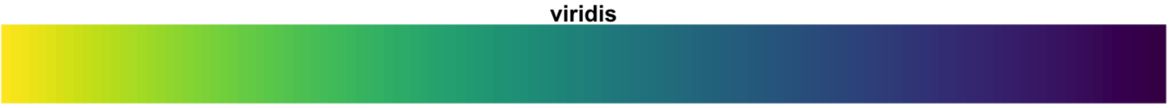
- Activation functions: Functions used at layers. ReLU (0,max) is used at Conv2D and Dense layers to remove negative values, and Softmax (or sigmoid in binary cases) is used at the final Dense layer to calculate each class' probability.

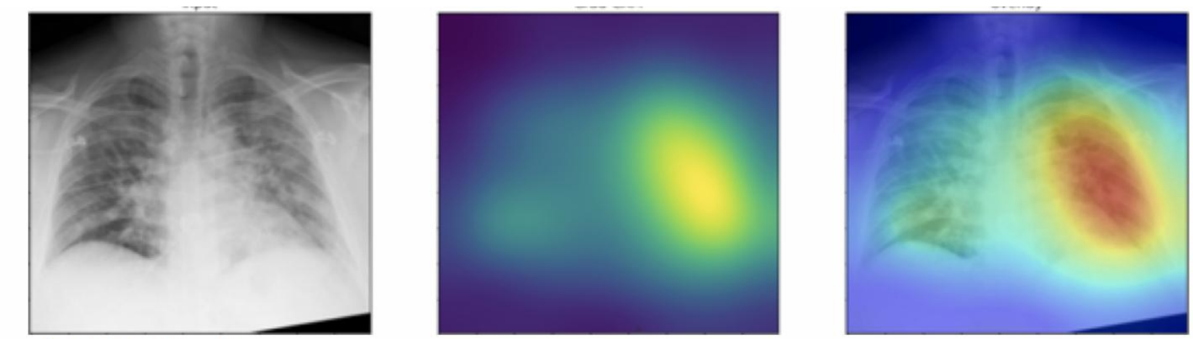
All 3 models for each disease have the same architecture (apart from the final layer) and were all trained and tested on Colab notebooks. Training was stopped early if the Loss didn't change in a certain amount of time to prevent overfitting. They were subsequently saved to Github to be cloned into another Colab notebook for classification.

GradCam

To use GradCam, a submodel has to be built. This submodel is built with the last Conv2D layer stacked onto all the Dense layers. This allows the final — most complex features (from all the Conv2Ds) to be in gradient forms.

Next, a colour gradient is applied to the gradients. More important gradients —> less important gradients are converted to yellow —> purple (viridis colormap).





Gradcam detects Covid infection in the left lung

Finally, using matrix addition we can overlay the gradients onto the image.

2. Abilities

- Can decently accurate malignant tumours, Covid-19, pneumonia, and Alzheimer’s.
- Can visualise areas of interest.
- Can provide confidence to aid human decisions.
- Can secure all data after classification.

3. Development platforms

- Language: Python

- Machine learning libraries: (tensorflow, keras, numpy, tf-explain …)

- Script runner and free gpus: Google Colab

4. Statistics and conclusions:

- Final results:

- Cancer model — Accuracy: **90.1%** — AUC: **89.4%**
- Covid - Pneumonia model — Accuracy: **95.3%** — Average AUC: **96.3%**
- Alzheimer's model — Accuracy: **83.7%** — Average AUC: **86.3%**