**Deep Neural networks can predict mortality from 12-lead electrocardiogram voltage data.**

Data :

1 775 926 ECGs from 397 840 patients (194 845 mortality events during the year after the ECG acquisition), Age, Sex, One\_year\_survival\_status

Deep Convolutional Neural Network

Data split :

train = 1 392 384 (144 214 ev)

validation = 28 357 (11 662 ev)

test = 355 185 (38 969)

The DNN is mostly used for image recognition. The essential difference between RNN and DNN is that DNN assumes that the inputs are images.

DNN are inspired by the way we, human beings, are recognizing things. (face = eyes + nose +mouth, on the wright place and what’s a nose ? Shape, color …)

In a DNN there are different types of layers: the regular one called dense layer (the same as in the RNN), the convolution layer and the pooling layer.

A dense layer is a layer where all of the neurons are connected to each other.

A dense layer learns global patterns in its global input space while the Convolutional Layers learn local patterns in small windows of two dimensions. Moreover, a CL can learn patterns and preserve spatial relationships. That allows CNN to learn complex images. (A first CL will learn color when a second edges then a third an object…)

A pooling Layer

In the article the DNN consist on:

* Inputs = ECG
* 1-dimentional convolution layer:

The 1-dimention part means that inputs are in black and white. (A CL is represented as a tensor (like a matrix but in 3D) where the 2 first dimensions are the spatial axes (height and width) and the 3rd one is called the depth and it correspond to the RGB code so it’s a vector of dimension 3, here it’s a vector of one dimension so just a number between 0 for black and 1 for white)

* Batch normalization :

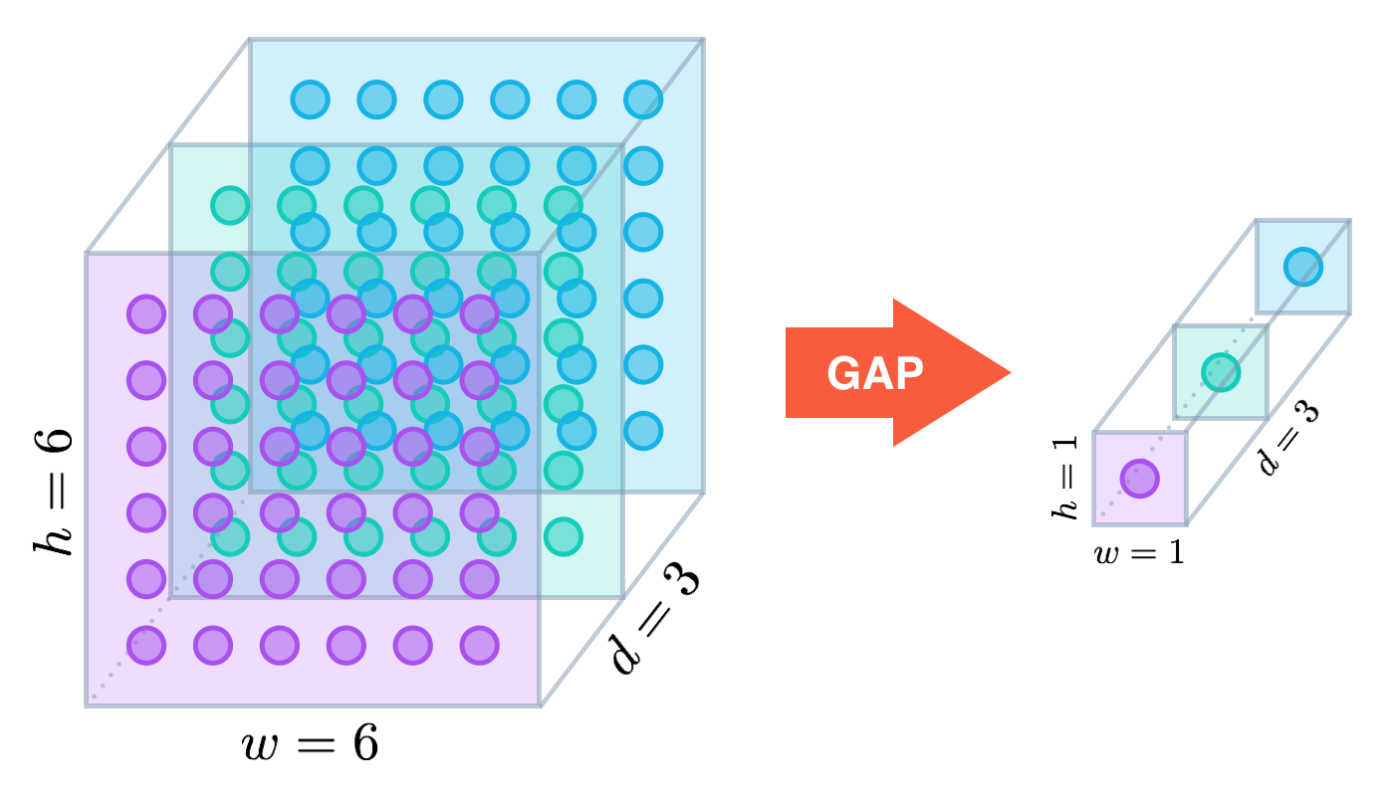
Which is a method used to reduce Internal Covariate Shift. Internal Covariate Shift is the change of network activations due to the change in network parameters during training. To be brief, in a deep NN there are many hidden layers. The first hidden layers is fed by the inputs layer and feed the second hidden layer who feed itself the 3rd hidden layer… So in the training process the weighs and bias change so the distribution of the network activations (the coefficient on the arrows!) change and that is a problem. The BN idea is to normalize the layer’s coefficients by the mean and the variance of the training set. But its shape makes it impossible (too large!)so we split the training in mini-batches and the normalization of each layer is done with the mean and variance of the inputs of each layer.

* Rectified linear units (ReLU) activations :

It’s an activation function (such as logit) define by f(x) = x+.

* Global Average Pooling

It is a method used to reduce the total number of parameters to estimate in, a model in order to minimize overfitting. In DNN case we transform a 3D tensor of dimension h x w x d to a tensor with dimension 1 x 1 x d. How ? by taking the average of each h x w.



* Adam optimization algorithm:

It’s an extension of the gradient stochastic algorithm.

* Layer with dropout:

It consists on drop the units which are not activated during the training. It forces the DNN to learn the same thing with different units. It prevents from overfitting.

* Binary Cross-entropy loss:

Cross-entropy loss, or log loss, measures the performance of a classification model whose output is a probability value between 0 and 1. Cross-entropy loss increases as the predicted probability diverges from the actual label. So predicting a probability of .012 when the actual observation label is 1 would be bad and result in a high loss value. A perfect model would have a log loss of 0.

Below is the formula where y is the actual label and p(y) the output (i.e. the predicted label).



* Epochs :

One Epoch is when an ENTIRE dataset is passed forward and backward through the neural network only once. Since one epoch is too big to feed to the computer at once we divide it in several smaller batches.

Why several epochs? Because we are adjusting the w and b iteratively with an optimization algorithm (here in the article it’s the Adam optimization algorithm).

So as the number of epochs increases, more number of times the weight are changed in the NN and the curve (of the loss) goes from under fitting to optimal to overfitting.

.