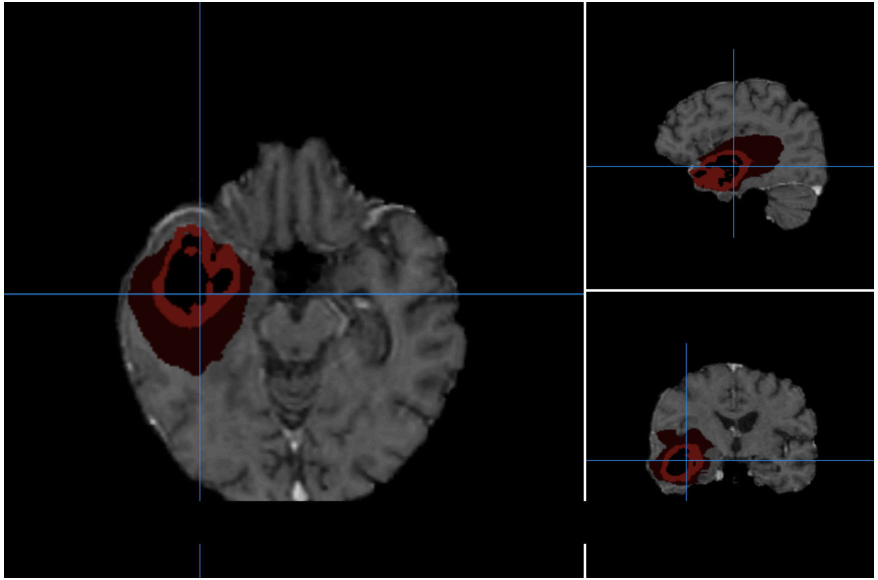
Deep Learning 101
(Brain Tumor Segmentation)

Ankit Modi

qure.ai

22 April2017

Problem Statement



Identify precise location of tumor within brain

Segment tumor regions

Estimate tumor volume

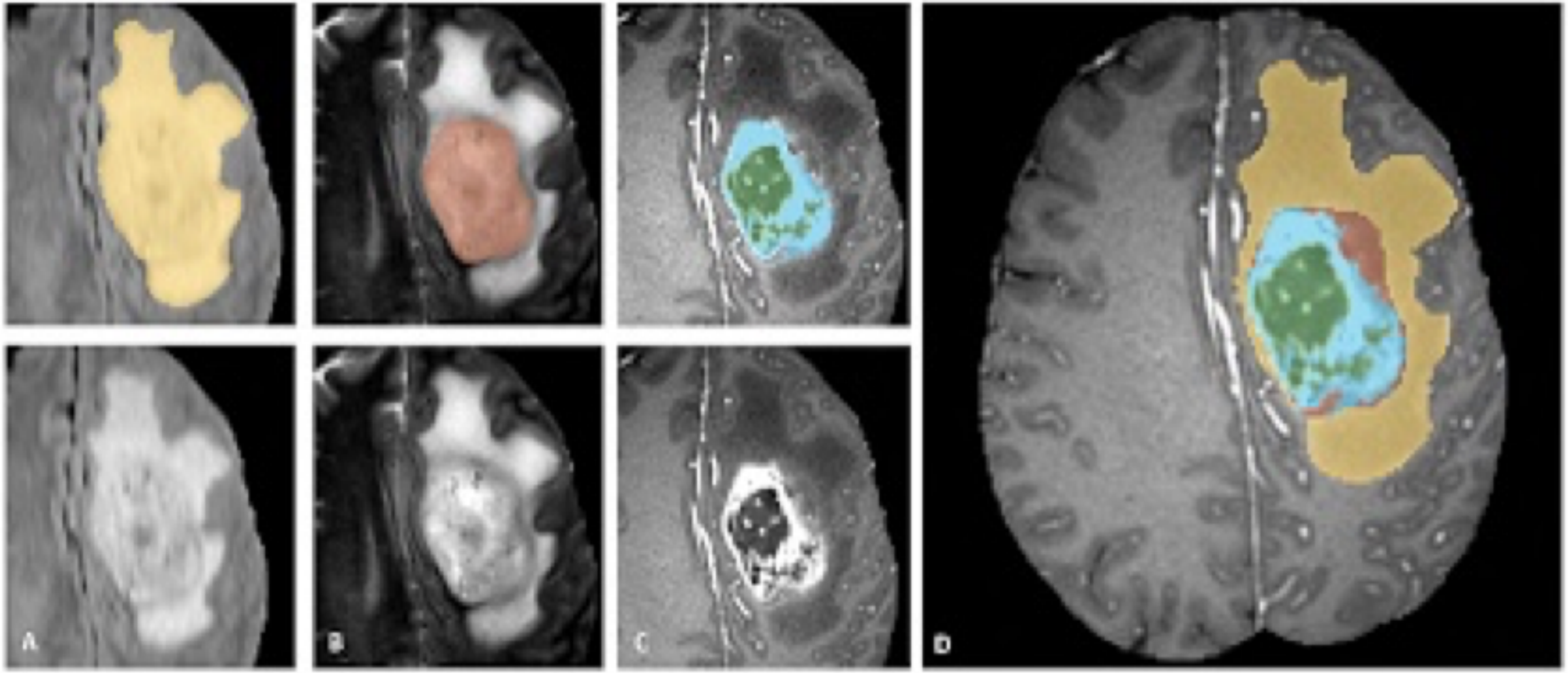


FIGURE: Manual annotation through expert raters. Shown are image patches with the tumor structures that are annotated in the different modalities (top left) and the final labels for the whole dataset (right). The image patches show from left to right: the whole tumor visible in FLAIR (Fig. A), the tumor core visible in T2 (Fig. B), the enhancing tumor structures visible in T1c (blue), surrounding the cystic/necrotic components of the core (green) (Fig. C). The segmentations are combined to generate the final labels of the tumor structures (Fig. D): edema (yellow), non-enhancing solid core (red), necrotic/cystic core (green), enhancing core (blue). (Figure from the BRATS TMI reference paper.)

Image source: [9]

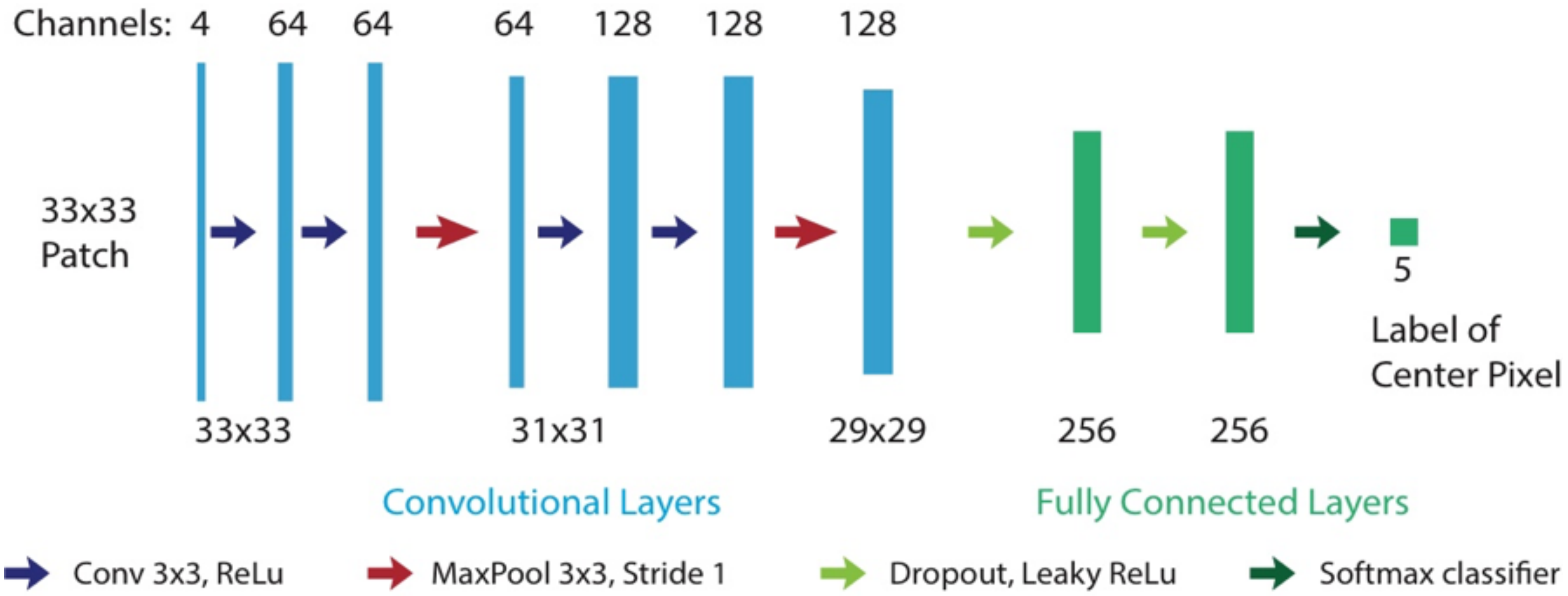
Dataset

- 2 private and 1 public dataset – 300 patients
- (80-20)% split for training and validation
- 4 modalities per patient: T1, contrast enhanced T1 (T1c), T2 and Flair
- 155 slices of size 240 X 240 in each modality

Implementation

Pre-processing → Convolutional Neural Network (CNNs) → 2-stage patch based training → Ensemble of CNNs → Post-processing (Removal of isolated clusters)

Network Architecture



qure.ai

Convolution

$$\begin{matrix} 1 & 1 & 1 \\ 0 & 1 & 1 \\ 0 & 0 & 1 \end{matrix} \quad * \quad \begin{matrix} 1 & 0 & 1 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{matrix}$$

$$1*1 + 1*0 + 1*1 + 0*0 + 1*1 + 1*0 + 0*0 + 0*0 + 1*1 = 4$$

qure.ai

Image source: [1]

Convolution

1	1	1	0	0
0	1	1	1	0
0	0	1	1	1
0	0	1	1	0
0	1	1	0	0

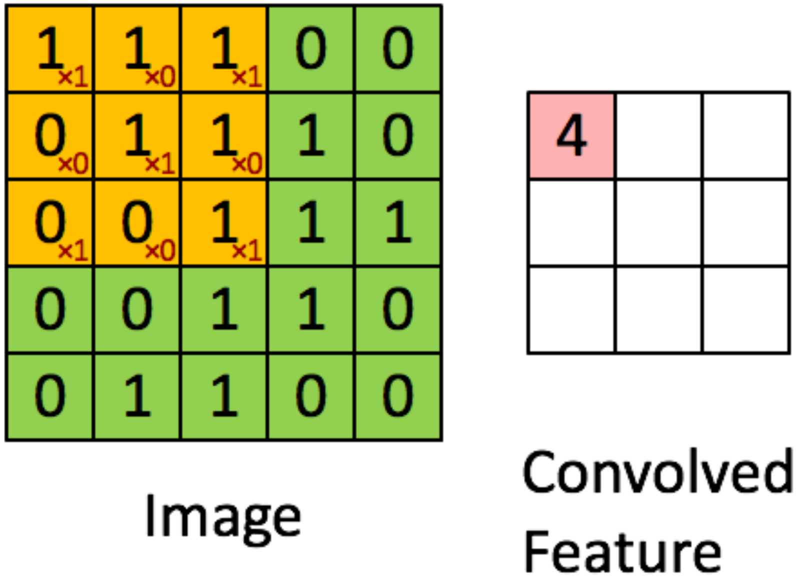
Image

1	0	1
0	1	0
1	0	1

Kernel / Filter

Image source: [1]

Convolution



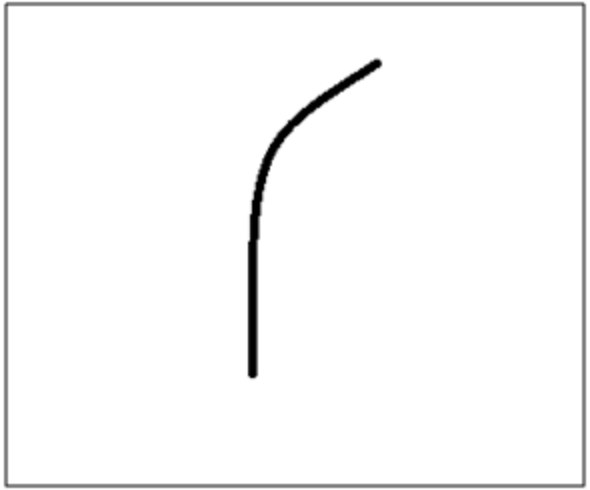
qure.ai

Image source: [1]

Convolution: Real world example

0	0	0	0	0	30	0
0	0	0	0	30	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	0	0	0	0

Pixel representation of filter

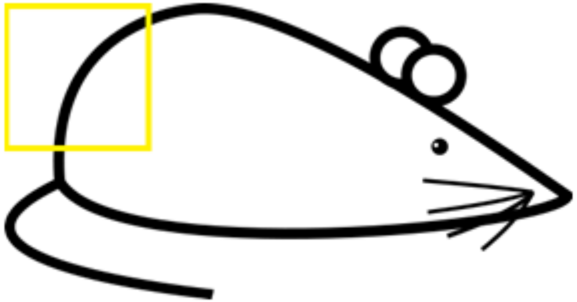


Visualization of a curve detector filter

Image source: [2]

Convolution: Real world example

Original image



Visualization of the filter on the image

Image source: [2]

Convolution: Real world example

Visualization of the receptive field

0	0	0	0	0	0	30
0	0	0	0	50	50	50
0	0	0	20	50	0	0
0	0	0	50	50	0	0
0	0	0	50	50	0	0
0	0	0	50	50	0	0
0	0	0	50	50	0	0

Pixel representation of the receptive field

*

0	0	0	0	0	30	0
0	0	0	0	30	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	0	0	0	0

Pixel representation of filter

Multiplication and Summation = (50*30)+(50*30)+(50*30)+(20*30)+(50*30) = 6600 (A large number!)

Image source: [2]

Convolution

Visualization of the filter on the image

0	0	0	0	0	0	0
0	40	0	0	0	0	0
40	0	40	0	0	0	0
40	20	0	0	0	0	0
0	50	0	0	0	0	0
0	0	50	0	0	0	0
25	25	0	50	0	0	0

Pixel representation of receptive field

*

0	0	0	0	0	30	0
0	0	0	0	30	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	0	0	0	0

Pixel representation of filter

Multiplication and Summation = 0

Image source: [2]

Convolution: Another Example

Image source: [3]

Non linearity: Why do we need it ?

Image source: [4]

Non linearity: Why do we need it ?

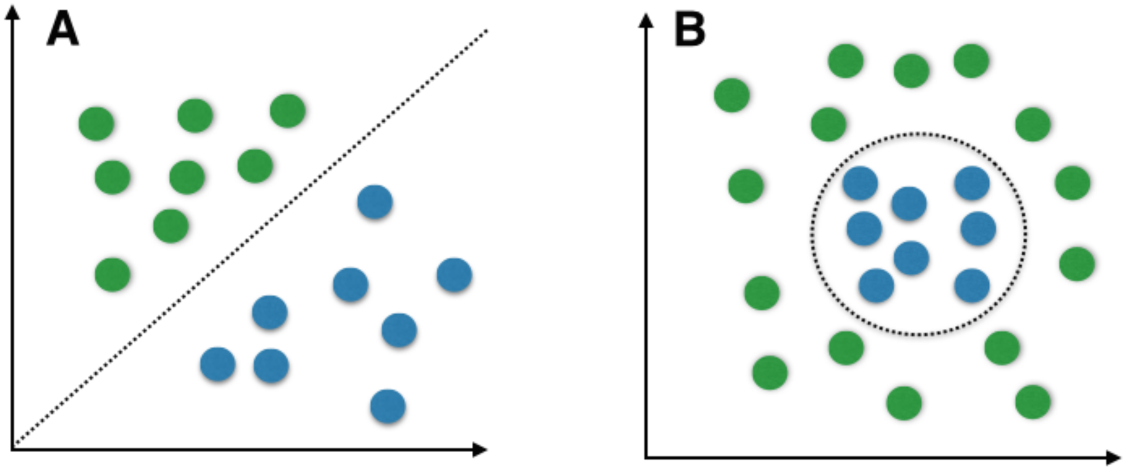
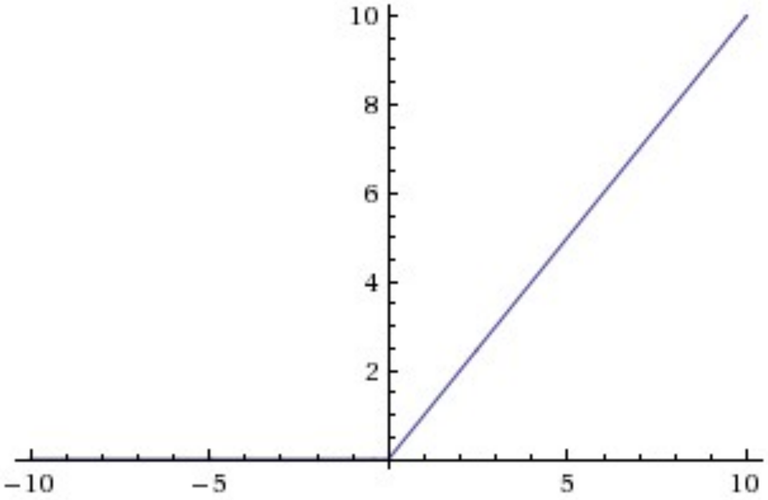


Image source: [4]

ReLU: Rectified Linear Units



$$output = \max(0, input)$$

qure.ai

Image source: [5]

How does ReLU solve the problem ?

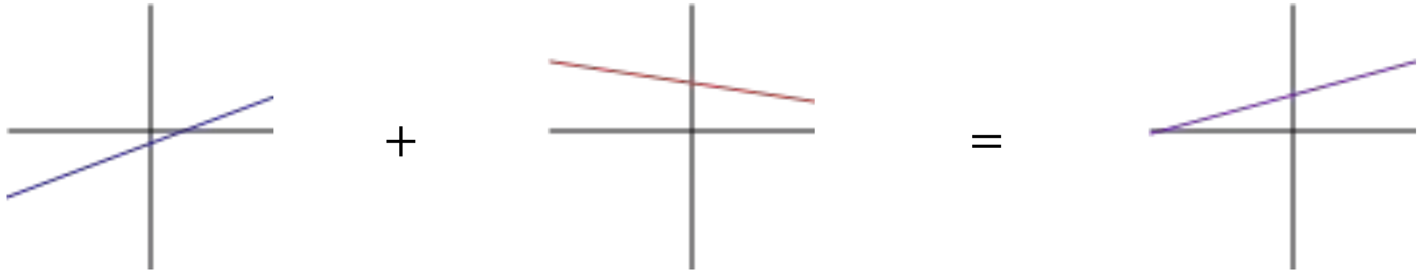
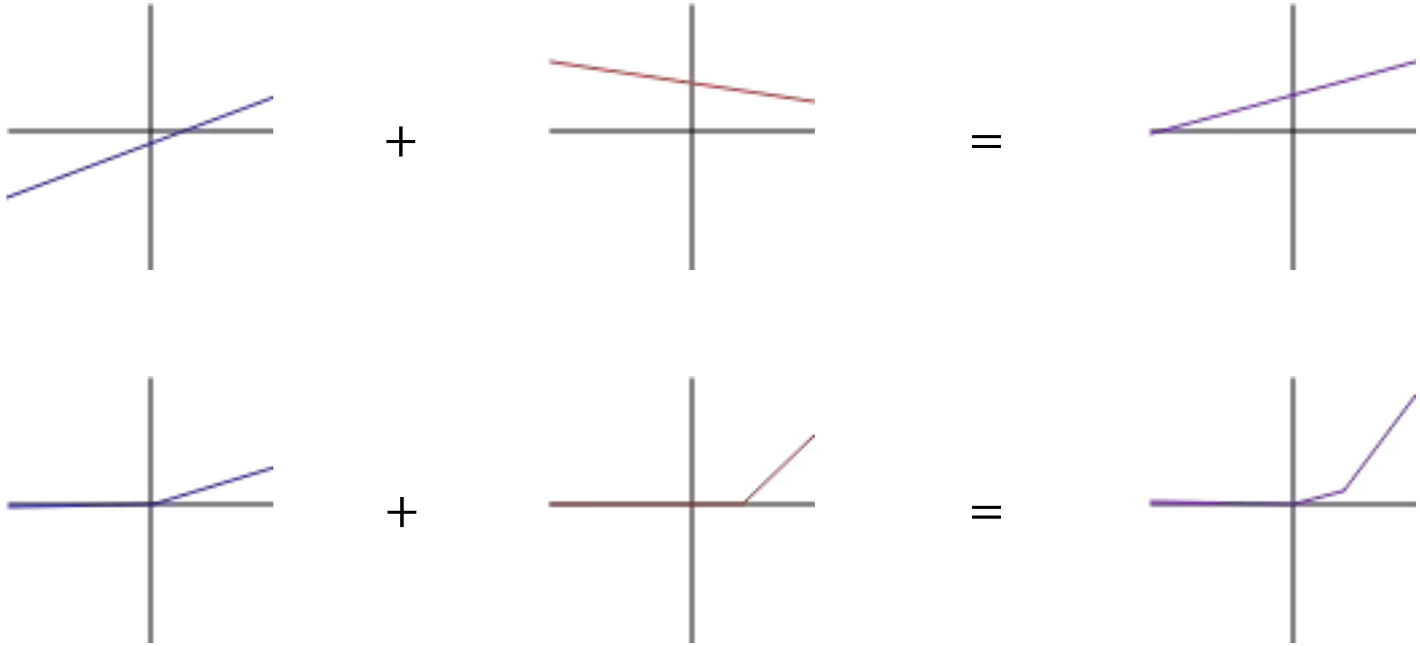


Image source: [6]

How does ReLU solve the problem ?



qure.ai

Image source: [6]

ReLU: Example

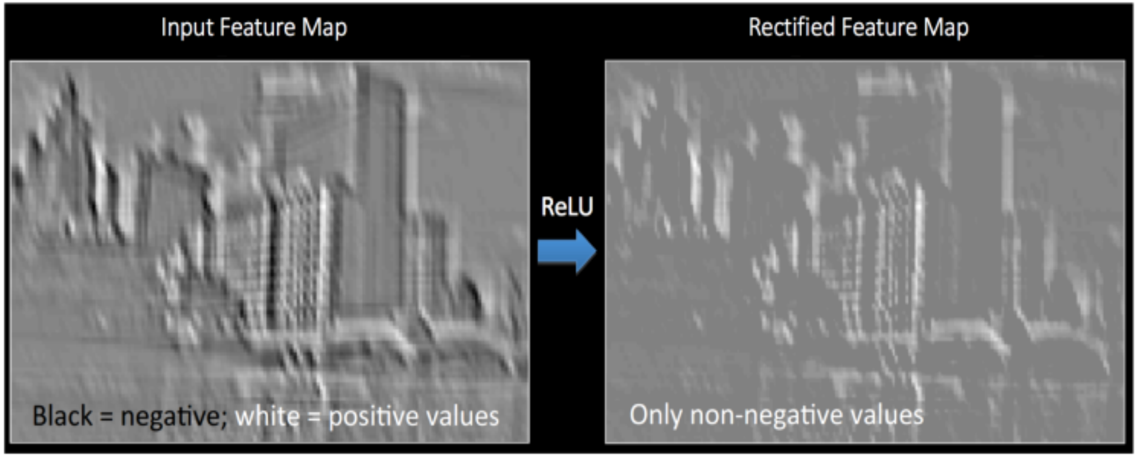


Image source: [3]

Pooling

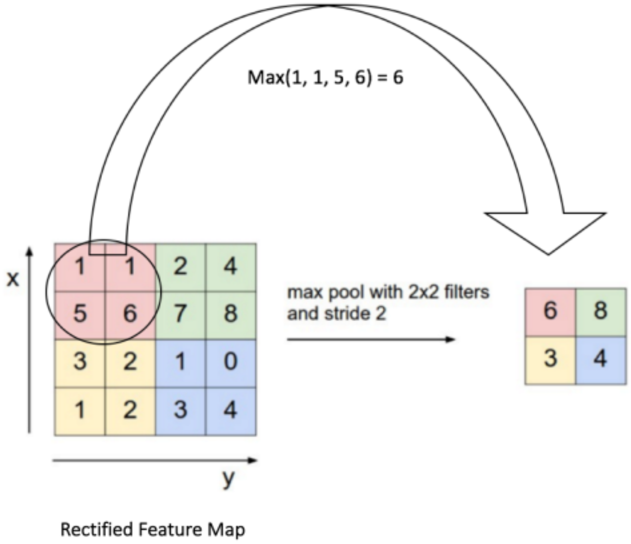


Image source: [4]

Pooling: Example

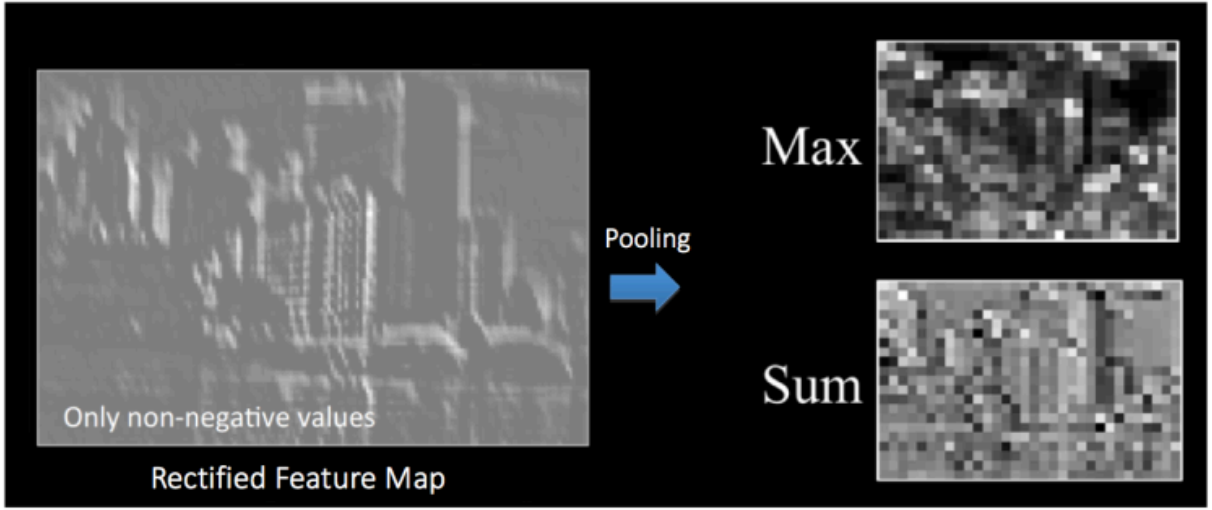


Image source: [3]

Pooling

Why ?

- Reduces the spatial size of representation

- Reduces the amount of parameters and computation

Does it lose valuable information ?

Pooling

Why ?

- Reduces the spatial size of representation

- Reduces the amount of parameters and computation

Does it lose valuable information ?

"The pooling operation used in convolutional neural networks is a big mistake and the fact that it works so well is a disaster."

~ Prof. Geoffrey Hinton [Source]

Overfitting

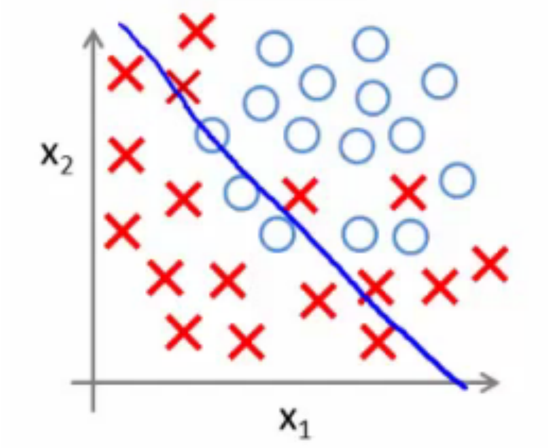


Image source: [8]

Overfitting

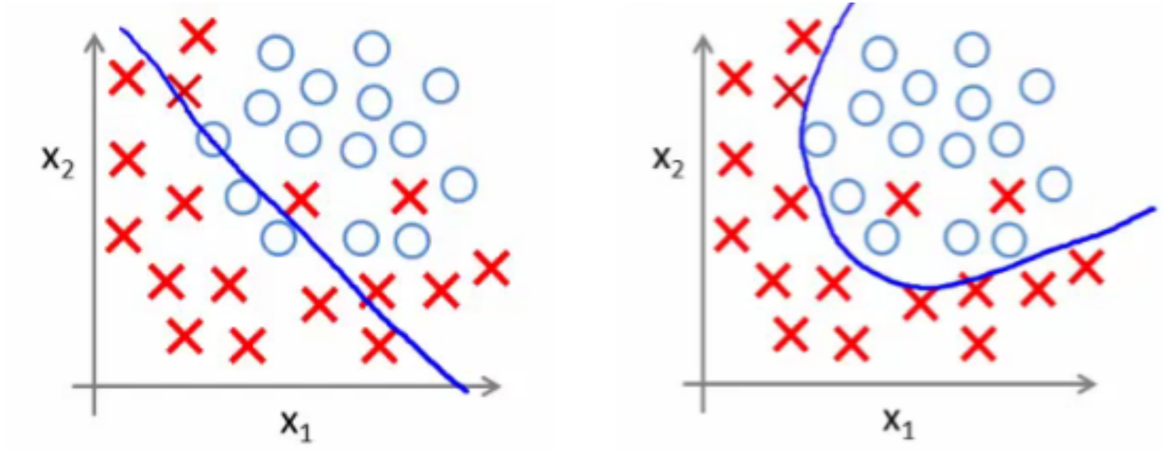
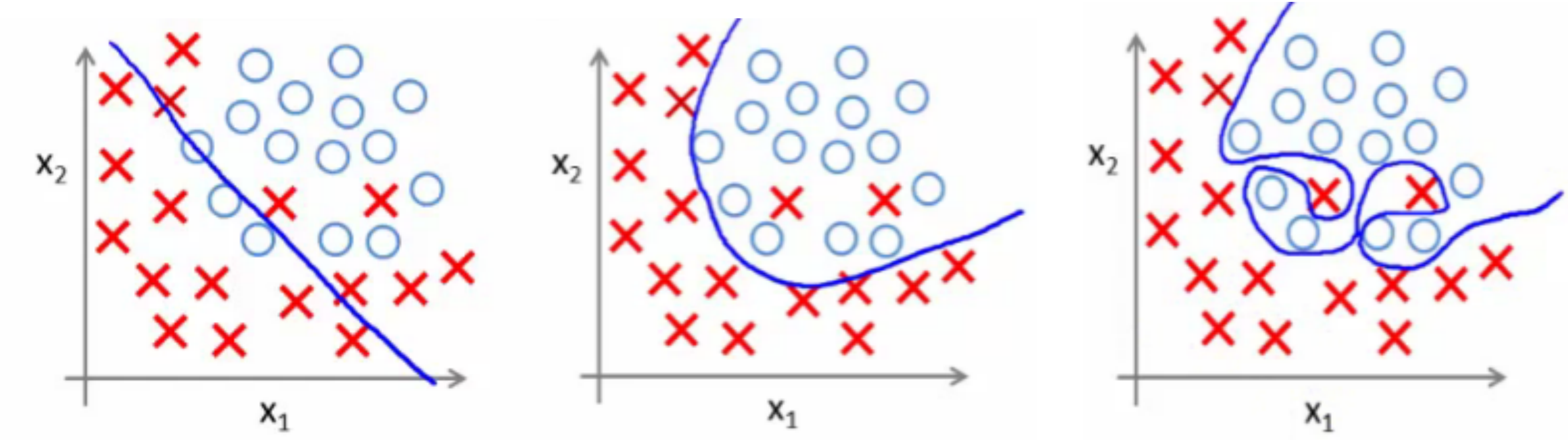


Image source: [8]

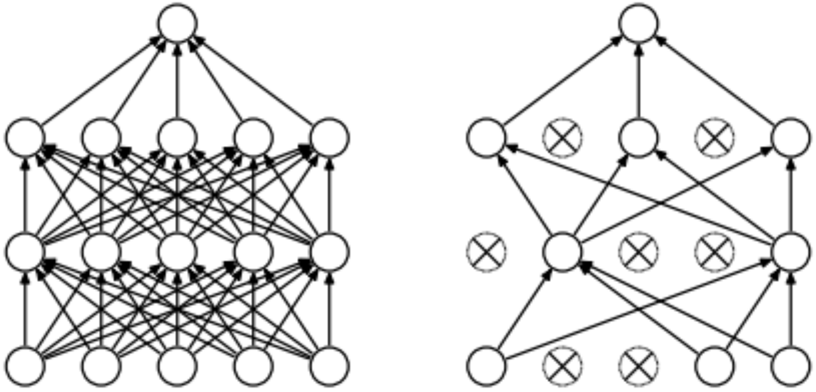
Overfitting



qure.ai

Image source: [8]

Dropout



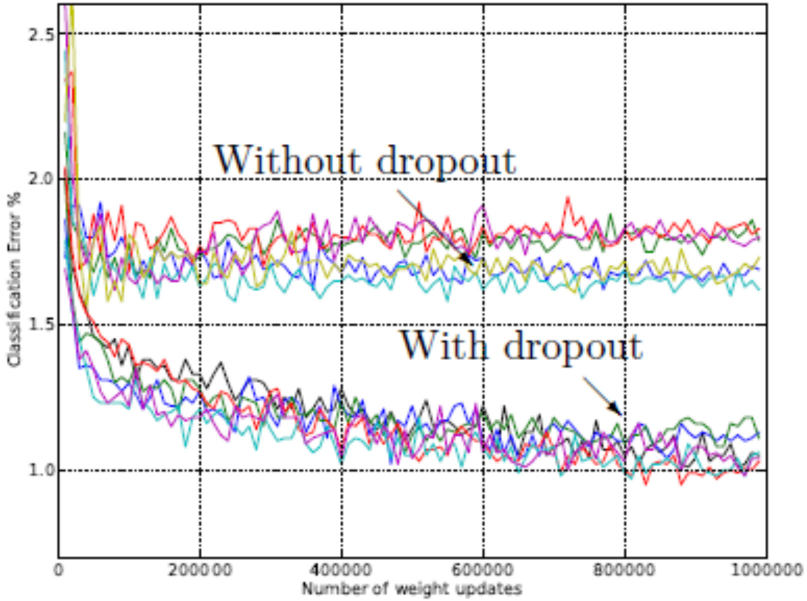
(a) Standard Neural Net

(b) After applying dropout.

Figure 1: Dropout Neural Net Model. **Left**: A standard neural net with 2 hidden layers. **Right**: An example of a thinned net produced by applying dropout to the network on the left. Crossed units have been dropped.

Image source: [7]

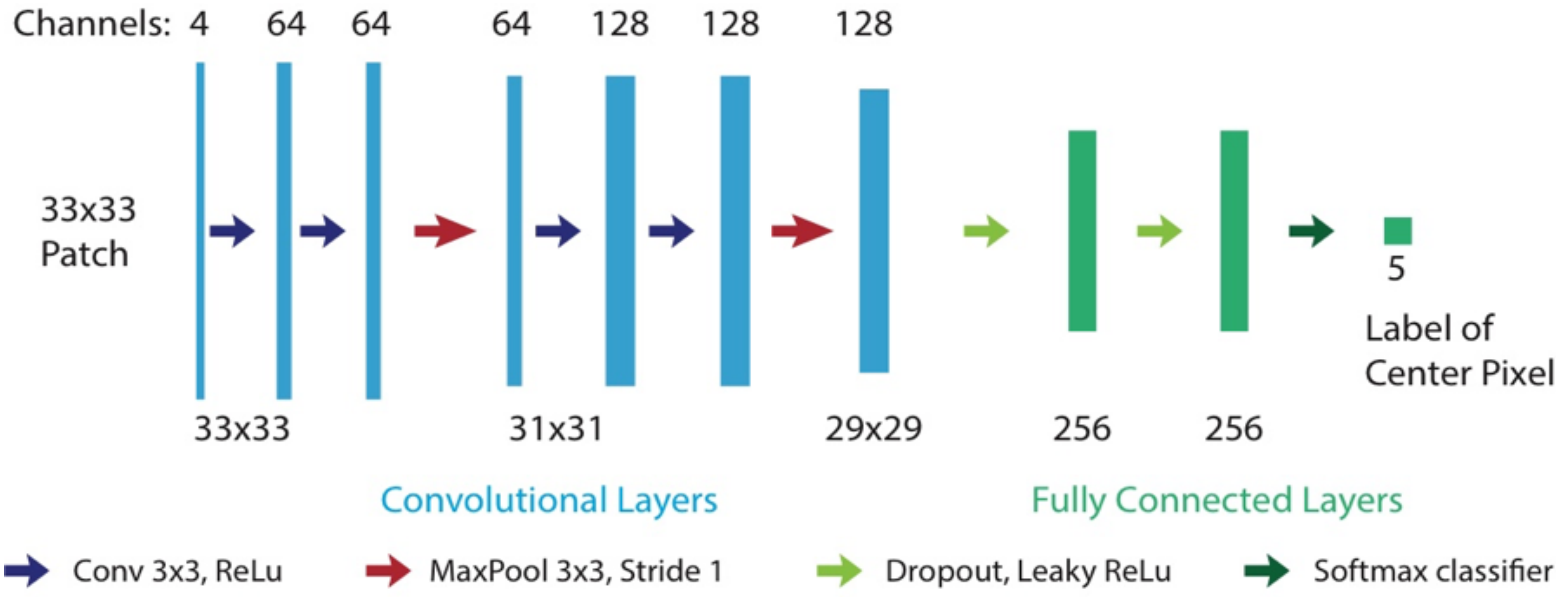
Dropout: Empirical Evidence



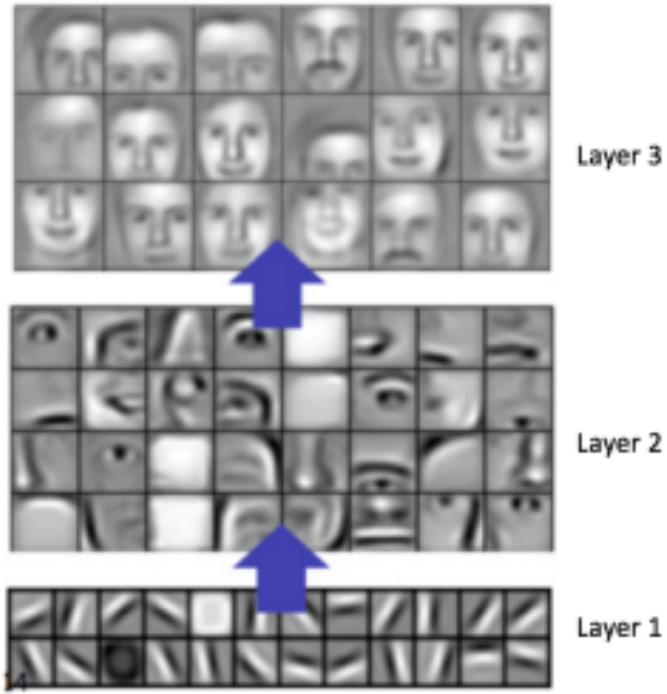
qure.ai

Image source: [7]

Network Architecture



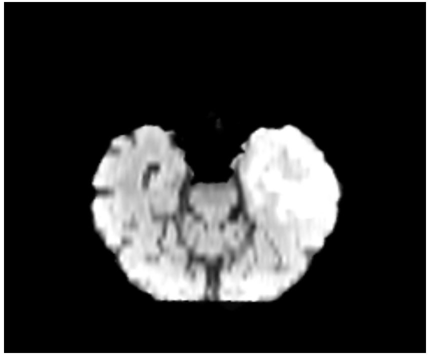
Convolution: Stacked layers



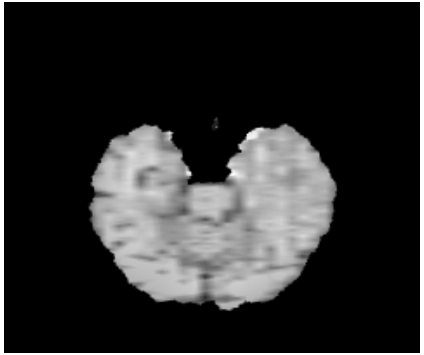
qure.ai

Image source: [3]

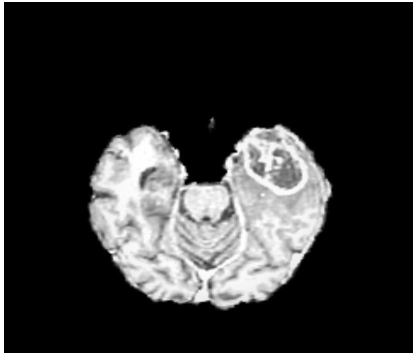
Visualizing results



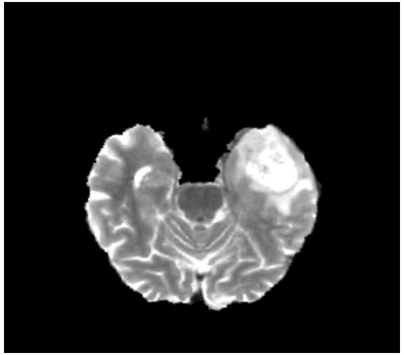
Flair



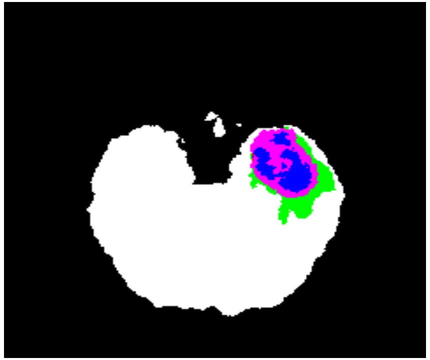
T1



T1C

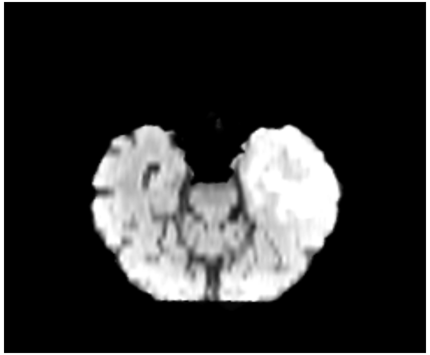


T2

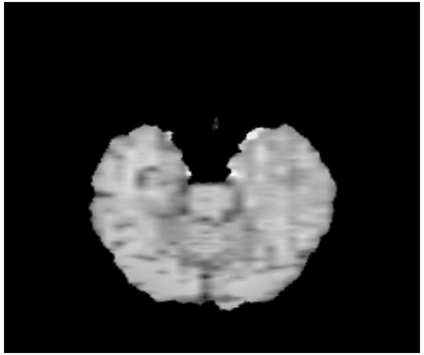


Ground truth

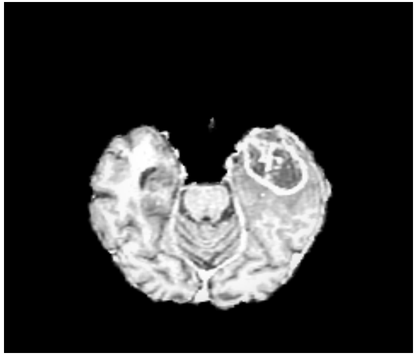
Visualizing results



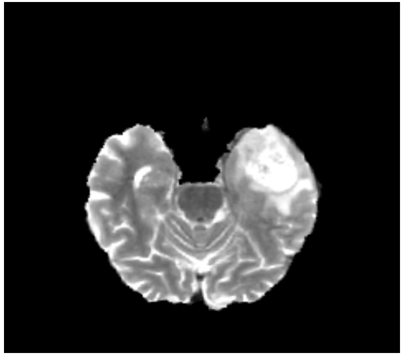
Flair



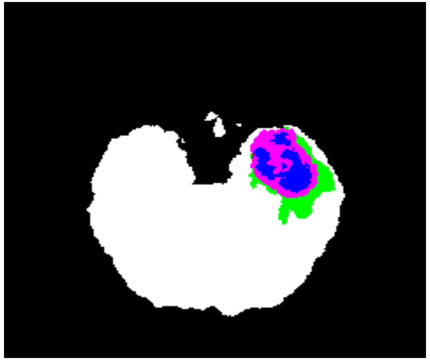
T1



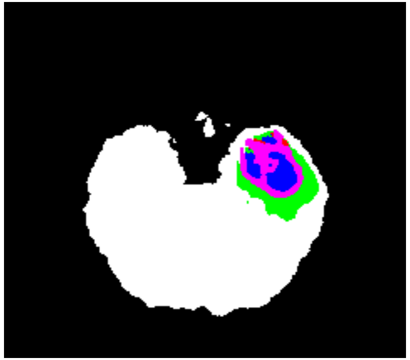
T1C



T2

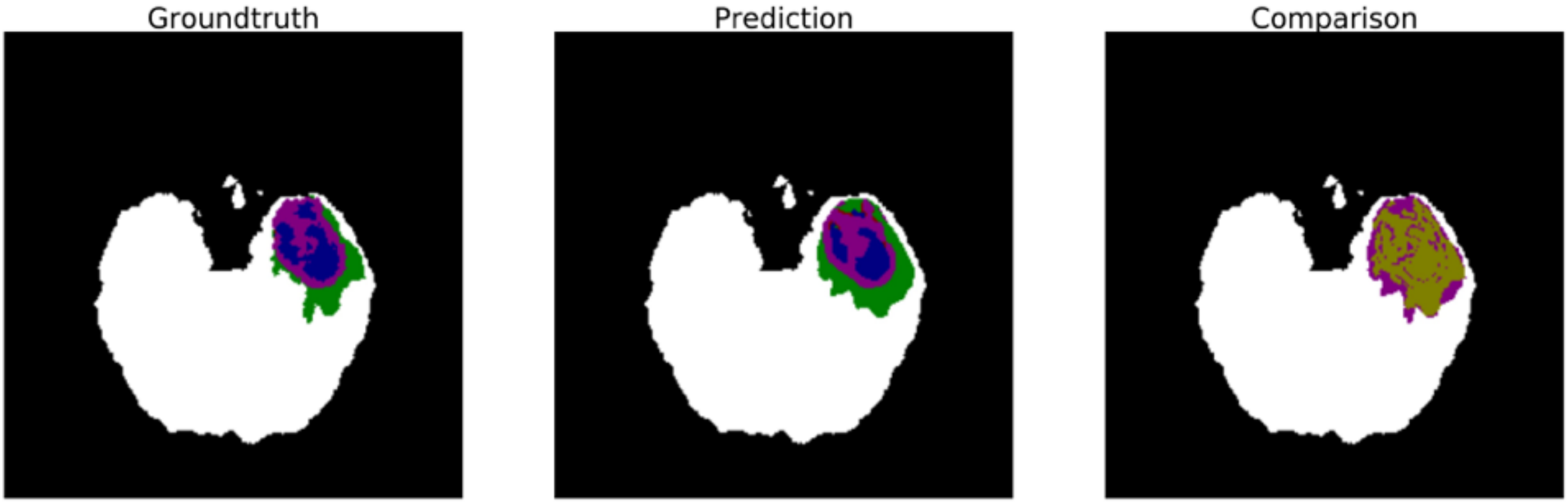


Ground truth

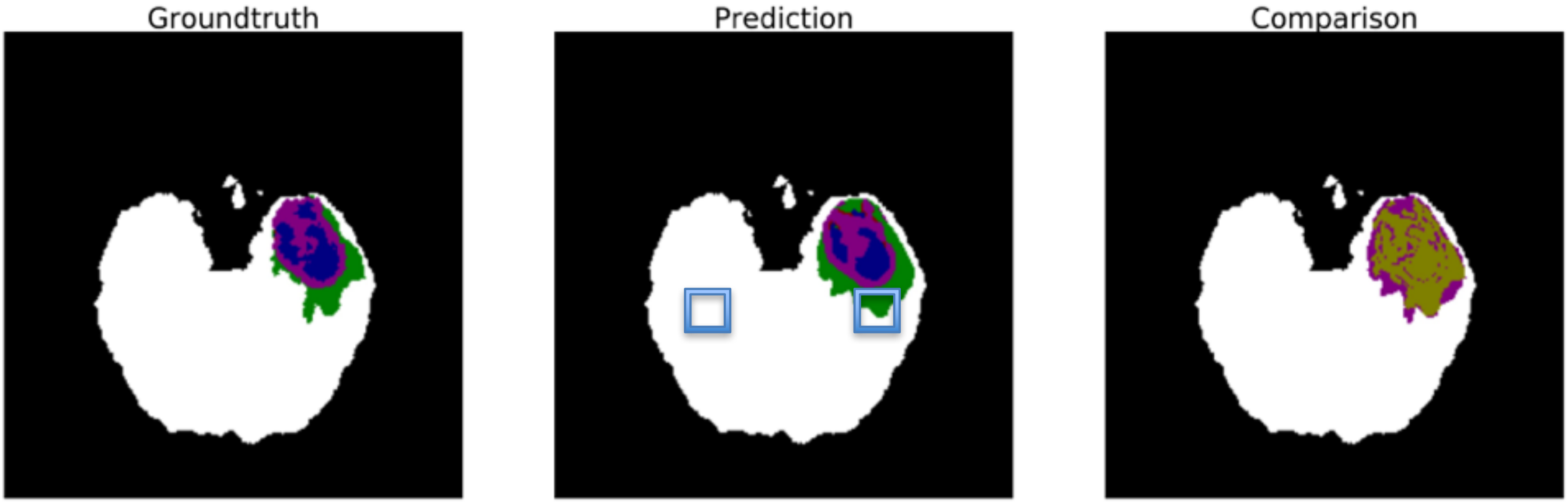


Prediction

Error analysis from one of the base models

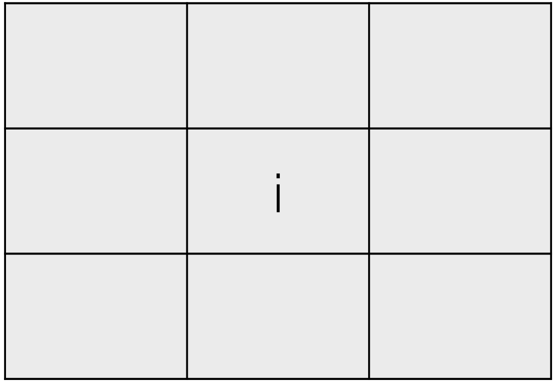


Identifying pixels on Edge



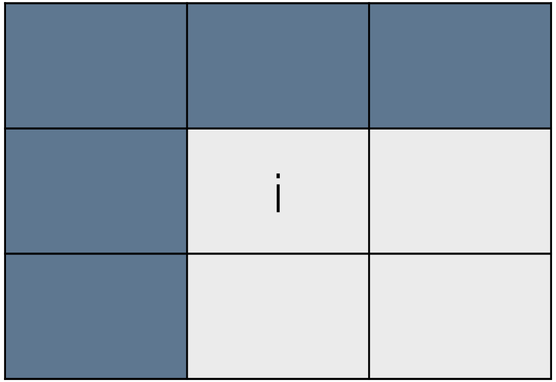
Solution: Pixel wise weighted loss function

$$w_i = \frac{N + \#\text{ patch pixels with label different from patch } i}{N + \#\text{ patch pixels with label same as patch } i}$$



Normal patch

$$w = \frac{N + 0}{N + 9}$$

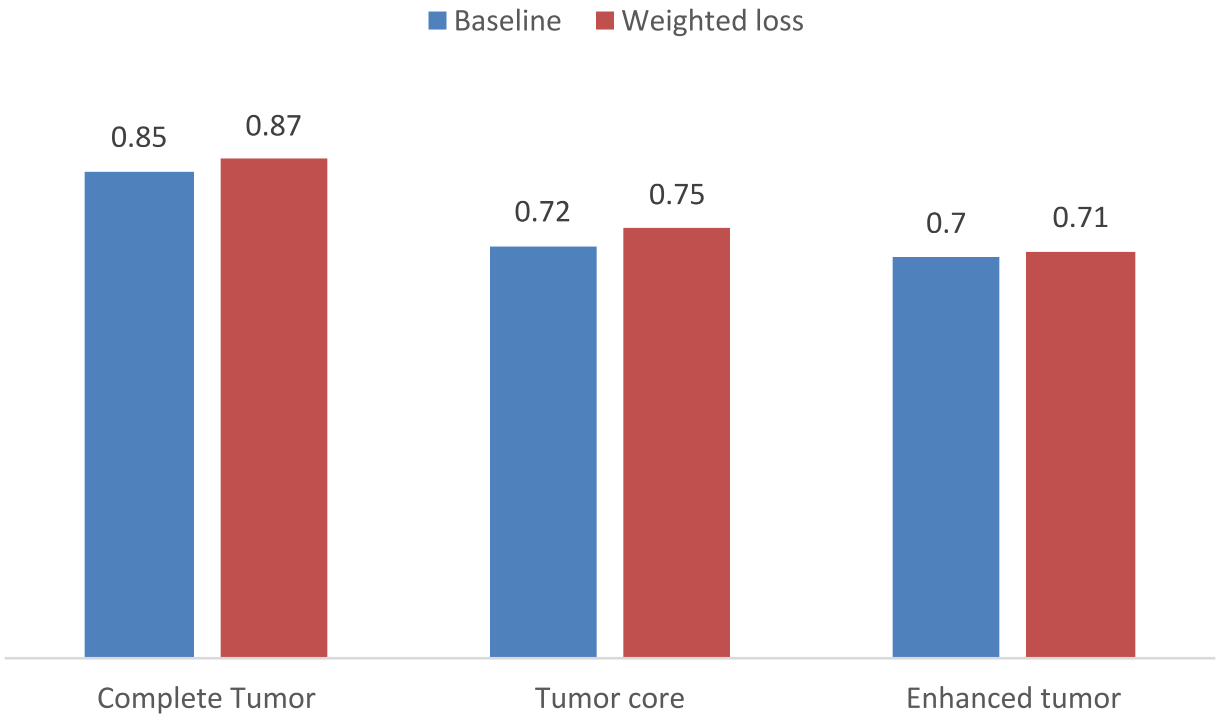


Patch at edge

$$w = \frac{N + 5}{N + 4}$$

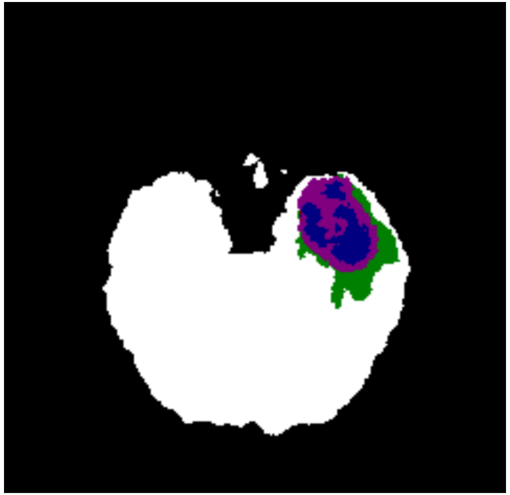
N: smoothing hyper-parameter

Dice score comparisons

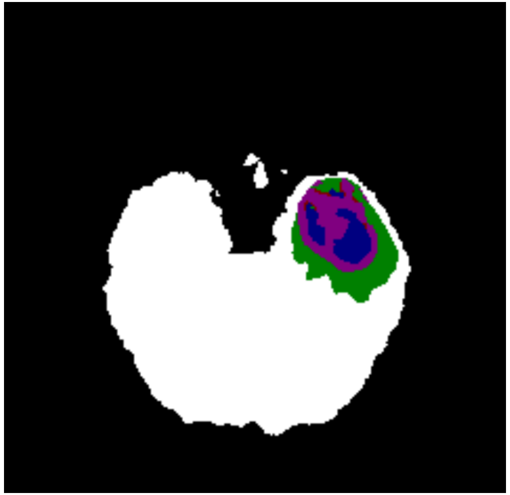


Results comparison

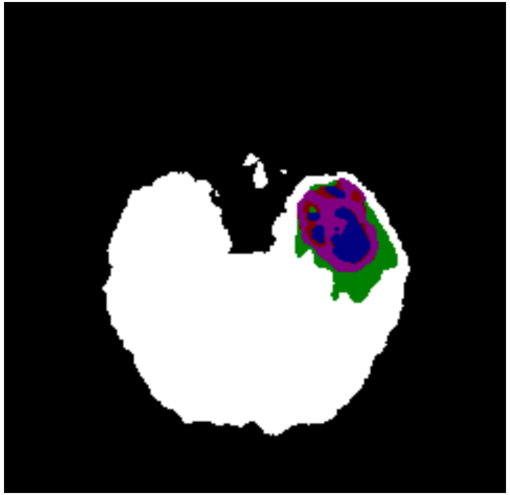
Ground Truth



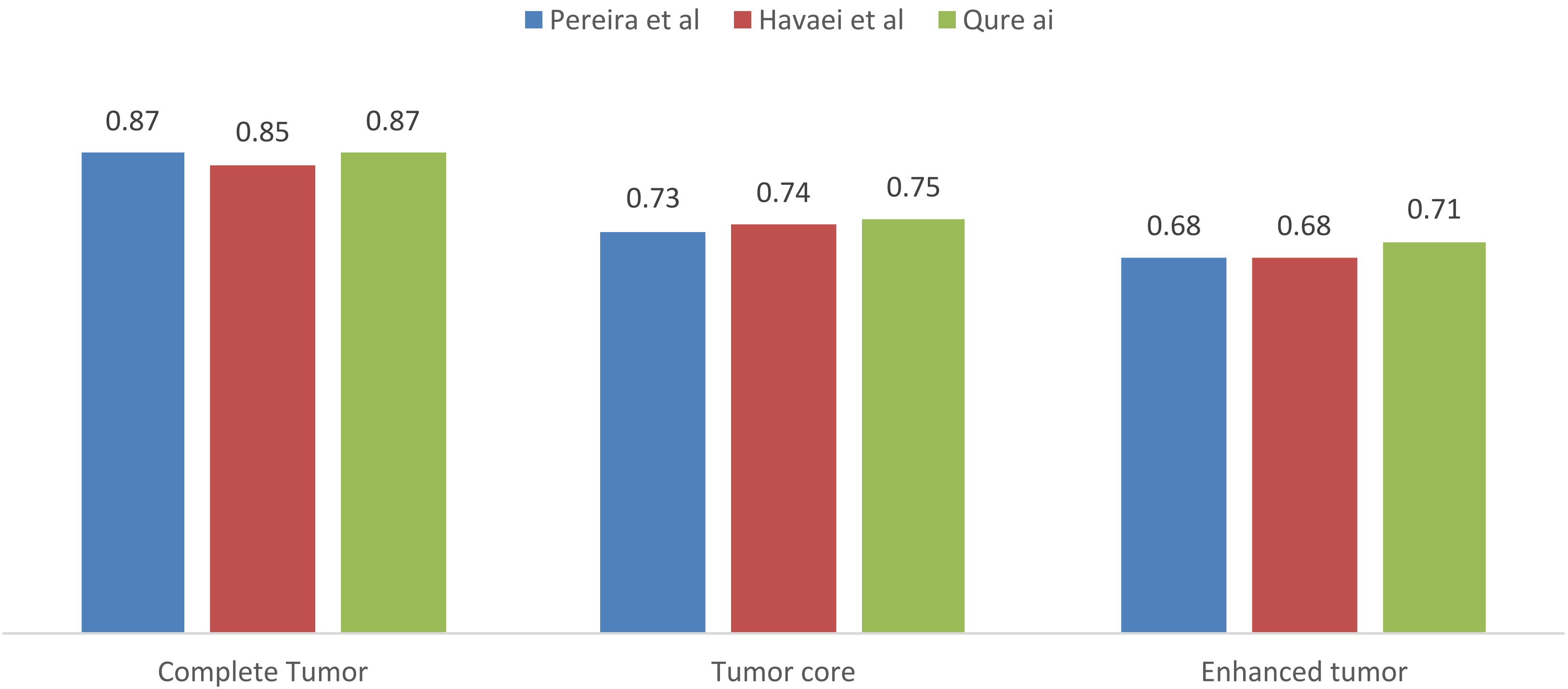
Baseline Model



Weighted Model



Results comparison



qure.ai

* Paper accepted

Demo: Automated report generation

Link

References and Sources

1. http://deeplearning.stanford.edu/wiki/index.php/Feature_extraction_using_convolution
2. https://adeshpande3.github.io/adeshpande3.github.io/A-Beginner's-Guide-To-Understanding-Convolutional-Neural-Networks/
3. https://ujjwalkarn.me/2016/08/11/intuitive-explanation-convnets/
4. http://sebastianraschka.com/Articles/2014_naive_bayes_1.html
5. http://cs231n.github.io/neural-networks-1/
6. https://www.quora.com/Why-does-deep-learning-architectures-use-only-non-linear-activation-function-in-the-hidden-layers/answer/David-Kobilnyk?srid=308s
7. http://wikicoursenote.com/wiki/Dropout
8. http://www.holehouse.org/mlclass/07_Regularization.html
9. http://braintumorsegmentation.org/