Architecture of a Neural Network

Authors: Russell Pekala, Satish Desai, Marvin Marshak
Home Institution: Harvard University

Summer Sponsor: Physics REU

Theoretical Underpinnings of Neural Networks

- A neural network is a system of interconnected nodes.
- Information is passed from the input nodes, to the thinking nodes, to the output nodes according to the weights.
- Propagated information is compared to known result (supervised learning).
- Weights are adjusted to minimize a loss function and hopefully improve accuracy.

1. Input matrix is modified by the first set of weights.

$$\begin{bmatrix} x_{11} & x_{12} \\ x_{21} & x_{22} \\ x_{31} & x_{32} \end{bmatrix} \begin{bmatrix} w_{11}^{(1)} & w_{12}^{(1)} & w_{13}^{(1)} \\ w_{21}^{(1)} & w_{22}^{(1)} & w_{23}^{(1)} \end{bmatrix} = \begin{bmatrix} z_{11}^{(2)} & z_{12}^{(2)} & z_{13}^{(2)} \\ z_{21}^{(2)} & z_{22}^{(2)} & z_{23}^{(2)} \\ z_{31}^{(2)} & z_{32}^{(2)} & z_{33}^{(2)} \end{bmatrix}$$

2. Nonlinearity is added element-wise. $a_{ij}^{(2)} = f(z_{ij}^{(2)})$

3. Matrix propagation to next layer.

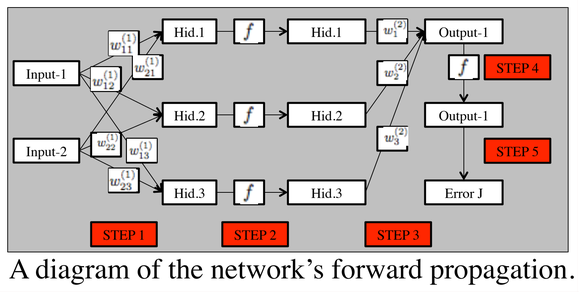
$$\begin{bmatrix} a_{11}^{(2)} & a_{12}^{(2)} & a_{13}^{(2)} \\ a_{21}^{(2)} & a_{22}^{(2)} & a_{23}^{(2)} \\ a_{31}^{(2)} & a_{32}^{(2)} & a_{33}^{(2)} \end{bmatrix} \begin{bmatrix} w_1^{(2)} \\ w_2^{(2)} \\ w_3^{(2)} \end{bmatrix} = \begin{bmatrix} z_1^{(3)} \\ z_2^{(3)} \\ z_3^{(3)} \end{bmatrix}$$

4. More nonlinearity, prediction calculated. $a^{(3)} = f(z^{(3)}) = \begin{bmatrix} f(z_1^{(3)}) \\ f(z_2^{(3)}) \\ f(z_3^{(3)}) \end{bmatrix} = \begin{bmatrix} \hat{y}_1 \\ \hat{y}_2 \\ \hat{y}_3 \end{bmatrix} = \hat{y}$

5. Error is calculated. $J(\hat{y}) = \sum_{i=1}^{3} \frac{1}{2}(y_i - \hat{y}_i)^2$

6. Weights are adjusted. Steepest Gradient Descent (SGD) is used. $w_i = w_i - \frac{\partial J}{\partial w_i} * r$

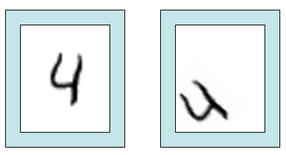
7. Process is repeated several thousand times.



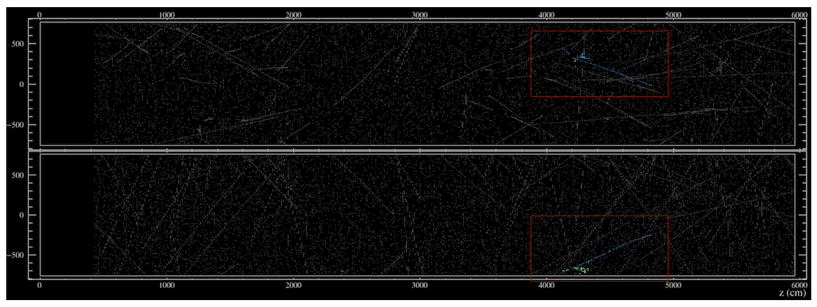
A diagram of the network's forward propagation.

Convolutional Layers in Neural Networks

- Seek to remove location invariance by using image convolutions.
- Convolutions distort images locally.
- Convolution layers therefore pick out *features* of an image.
- The "weights" in these layers become the weights in convolution filters.



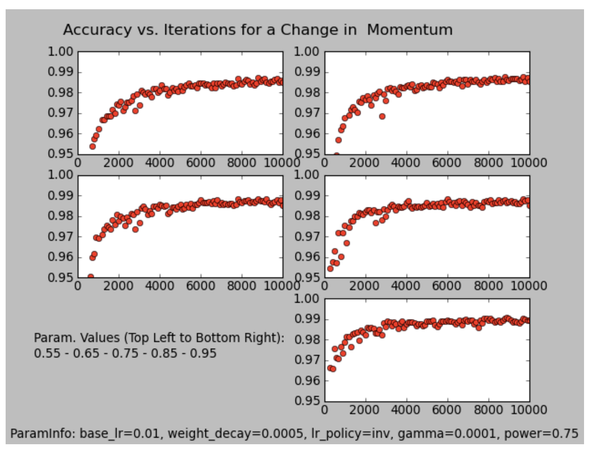
The left picture could be identified as a 4 in either a traditional or a convolutional neural network. The right off-center picture would only be identified as a 4 by a CVN.

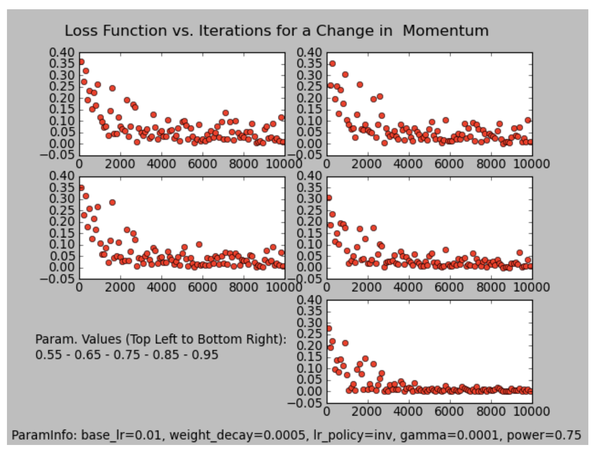


An example NOvA Event display for a muon neutrino CC interaction. Its identifying traits are not invariant in location within the detector.

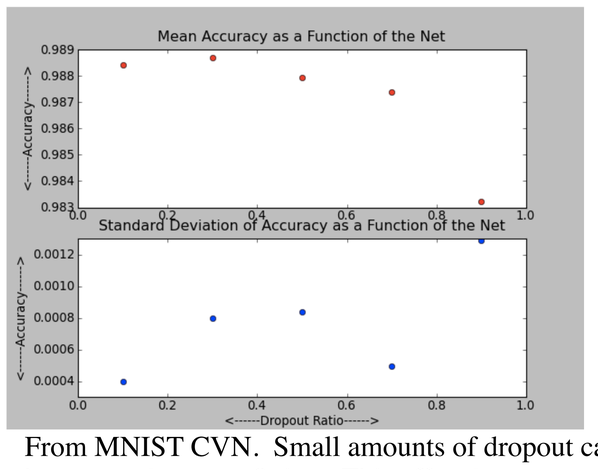
Adjustments for Smoother and Faster Convergence

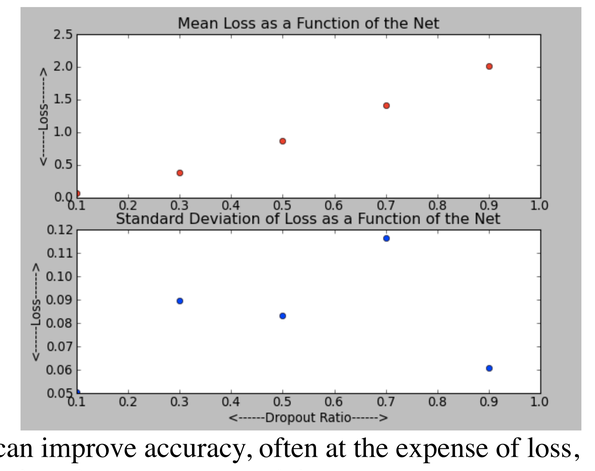
- The base learning rate (from step 6 above) assigns how steep to move in direction of the gradient.
- Momentum allows for the convergence process to skip over local minima in an effort to find global minima of the loss function.
- Dropout and weight decay are parameters that prevent over-fitting.
- Other adjustments to the SGD algorithm can also be easily programmed into caffe (Jia 2014).





From MNIST CVN. Accuracy and loss improve both faster and with less bumpiness with an increase in momentum, up until at least momentum .95.

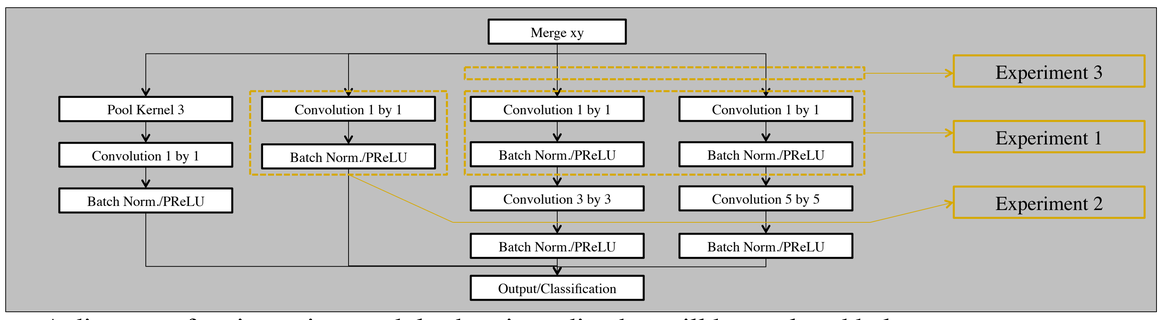




From MNIST CVN. Small amounts of dropout can improve accuracy, often at the expense of loss, by preventing over-fitting. This effect is greater when there are more weights.

Understanding NOvA Network Structure

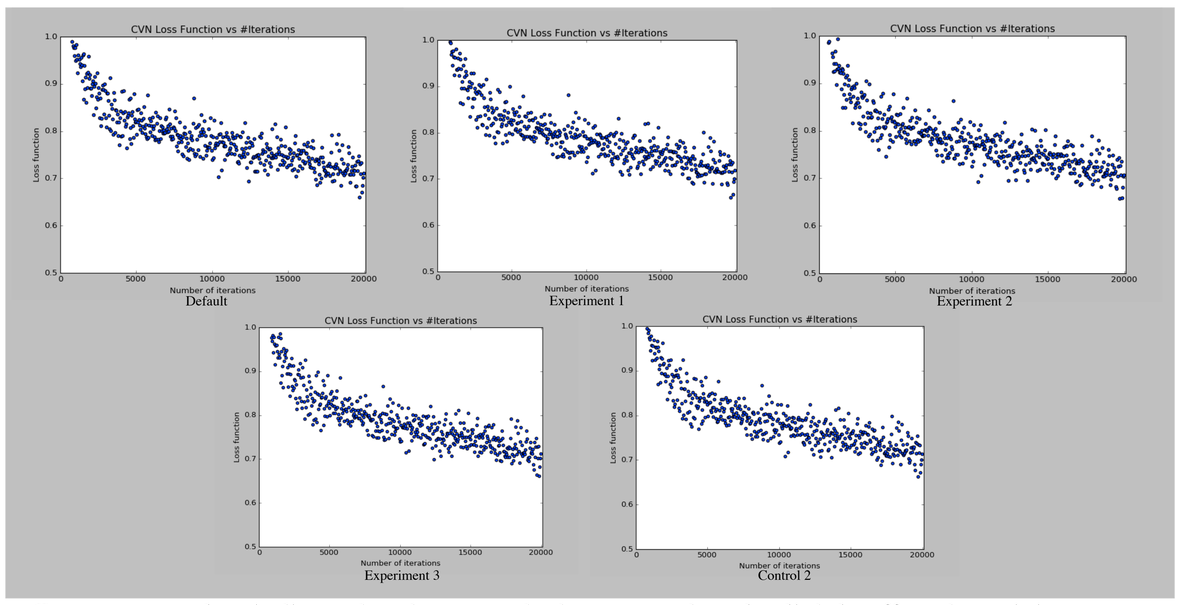
- Independently processed X and Y views are connected and processed in inception module.
- This later processing hasn't been optimized much since what it sees isn't well understood.



A diagram of an inception module showing edits that will be explored below.

Adjusting the Structure of a Neural Network

- I focused on investigating what could be optimized after the X and Y views had been merged.
  - Experiment 1: Remove 1×1 convolutions before both the 3×3 and 5×5 convolution layer.
  - Experiment 2: Remove independent 1×1 convolution layer.
  - Experiment 3: Add 3×3 pooling before 3×3 and 5×5 convolution streams.
  - Control 2: Remove the net's first convolution layer (long before the xy merge).



Convergence plots indicate that the network changes made only slightly affect the training process.

	Loss (ΔLoss)	Volatility (ΔVolatility)	Time (ΔTime) / seconds
Default Net	.721 (NA)	.0256 (NA)	27302 (NA)
Control 2	.723 (+ 0.35%)	.0252 (- 1.21%)	22015 (-19.37%)
Experiment 1	.724 (+ 0.47%)	.0248 (- 3.10%)	22088 (-19.10%)
Experiment 2	.717 (- 0.54%)	.0252 (- 1.54%)	26532 (- 2.82%)
Experiment 3	.719 (- 0.24%)	.0256 (+ 0.26%)	26987 (- 1.15%)

Statistics of the last 10% of training iterations. Improved performance after the removal of 1×1 convolution layers before the 3×3 and 5×5 convolution layers (Experiment 2) indicate that the model had most likely been suffering from over-fitting.

Further Improvements

- It's suggested that different types of images are better analyzed by different convolution filters (Zeiler 2013). Since the inception modules being used here are optimized for Google's image classification tasks, it's likely better filters could be found to analyze the streaks that are common in neutrino interaction images.
- Confusion matrices and other tools can help to describe what types of classification errors a neural network is making. Understanding these mistakes can help guide neural net architecture work by identifying filters that enhance differences between distinct event types specifically.

References

- Jia, Yangqing, Evan Shelhamer, Jeff Donahue, Sergey Karayev, Jonathan Long, Ross Girshick, Sergio Guadarrama, and Trevor Darrell. "Caffe." Proceedings of the ACM International Conference on Multimedia - MM '14 (2014): n. pag. Web.
- LeCun, Yan, Corinna Cortes, and Christopher J.C. Burges. "THE MNIST DATABASE". *MNIST Handwritten Digit Database*. N.p., n.d. Web 08 Aug. 2016.
- Rocco, D. R. (2016). *Muon Neutrino Disappearance in NOνA with a Deep Convolutional Neural Network Classifier* (Doctoral Dissertation). Retrieved from the University of Minnesota Digital Conservatory.
- Zeiler, Matthew D. *Visualizing and Understanding Convolutional Networks*. *12 Nov. 2013*.

Acknowledgements

- Thanks to the NSF for summer research funding and to Alex Kamenev for being accommodating.