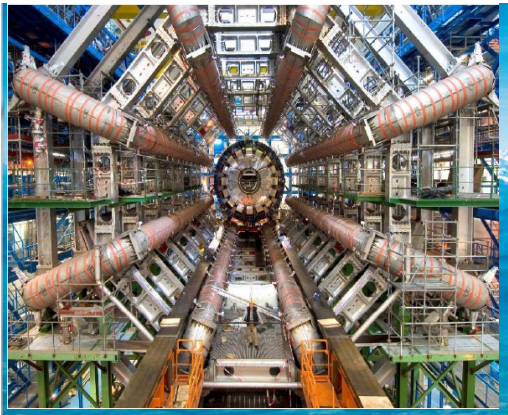
Identifying objects in ATLAS through machine learning techniques

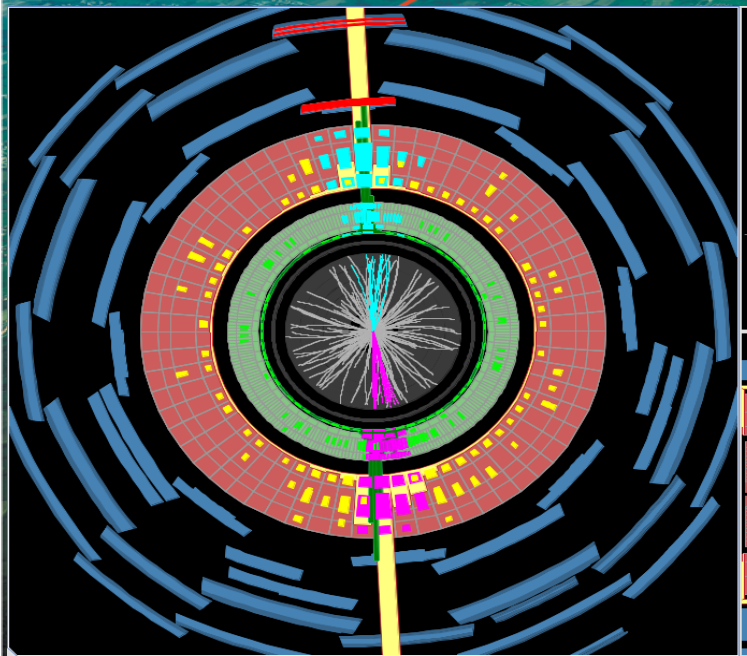
By

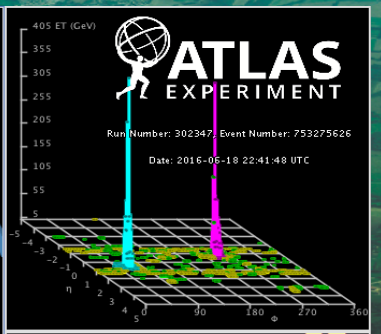
Wasikul Islam (Oklahoma State University),
Nesar Soorve Ramachandra (University of Kansas),
J. Taylor Childers (Argonne National Laboratory)

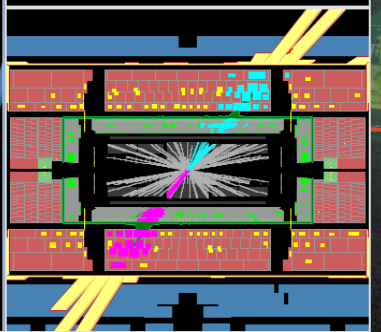
US-LUA meeting 2017

The ATLAS detector at CERN







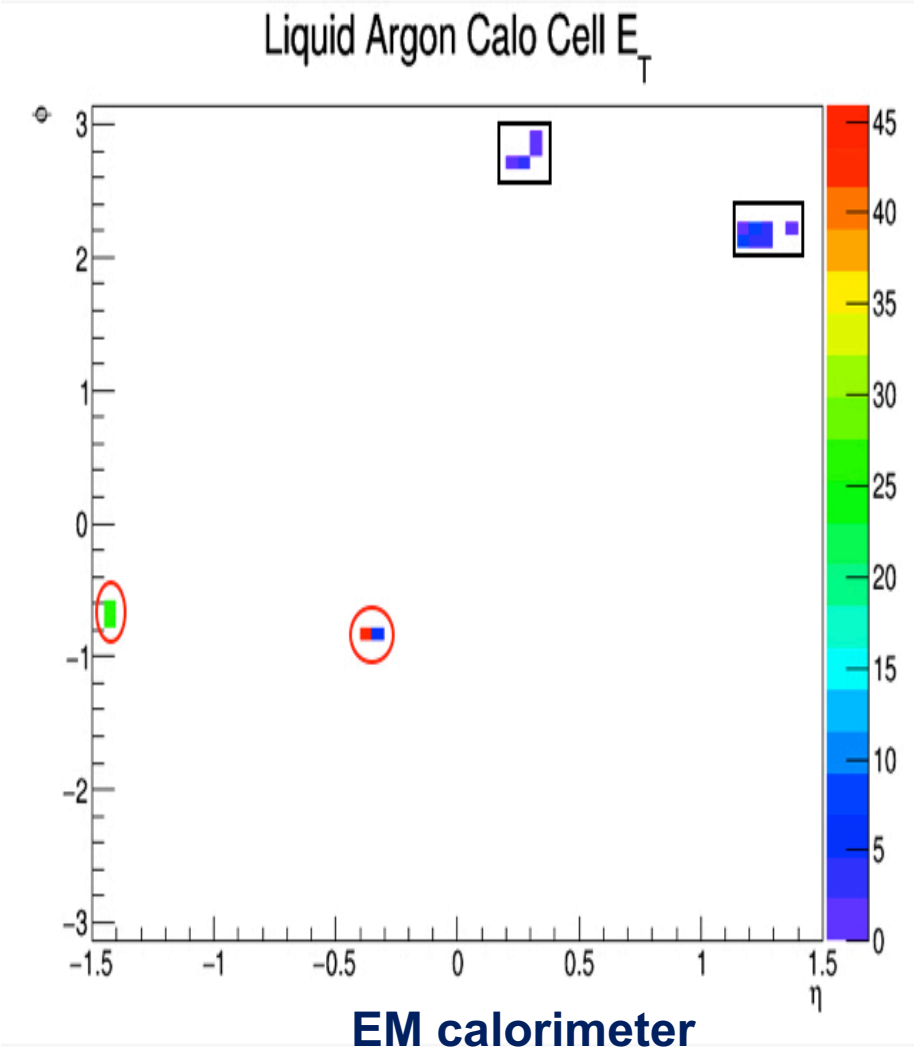


Event display at the detector

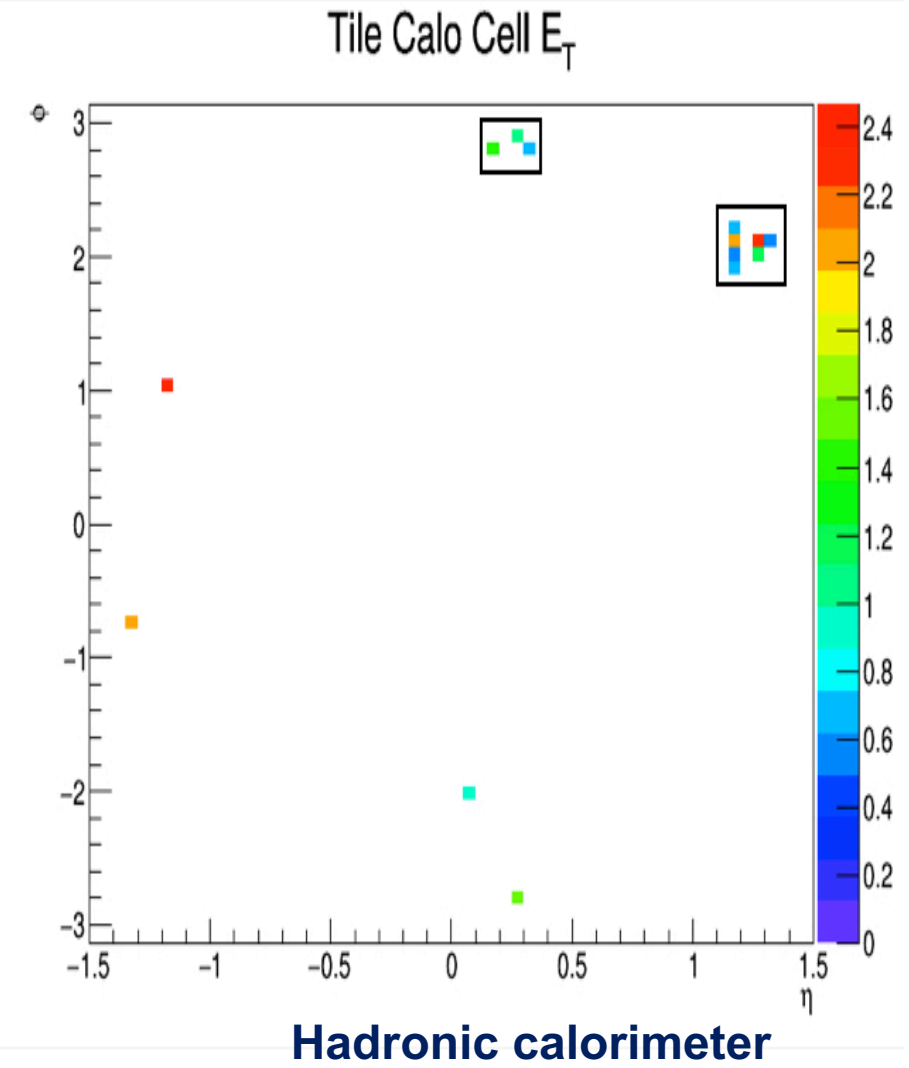
Our strategy for using Machine learning techniques for object identification

- Creating 2D images from the hits in calorimeter for Zee/Zmumu/Ztautau + 2 jet events .
- Using truth information, creating subimages containing leptons and jets.
- Starting from the popular Cifar10 model, defined in keras framework. [link: https://github.com/fchollet/keras/blob/master/examples/cifar10_cnn.py]
- Using model for classifying objects of 4 classes I.e; Electrons, Muons, Tau, and Jets.
- Then starting to vary hyper parameters (Learning rate, decay rate, batch size, number of epochs, loss functions, optimizers etc.) to optimize the accuracy of the model.

Imaging the hits in the calorimeter



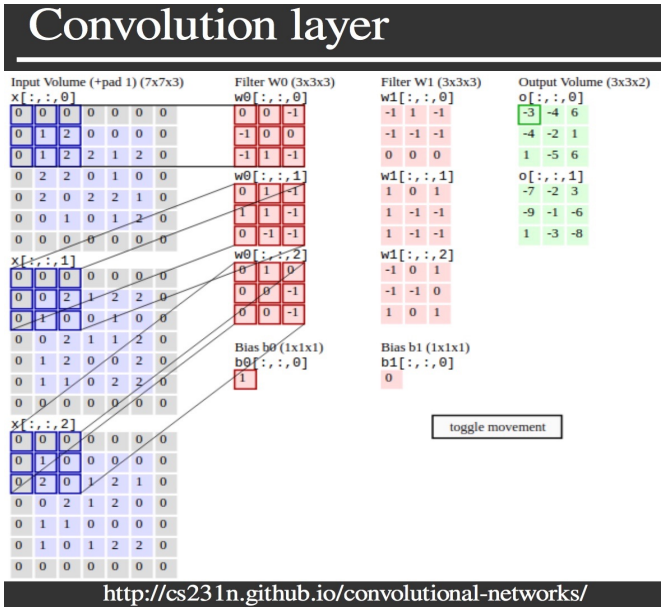
EM calorimeter



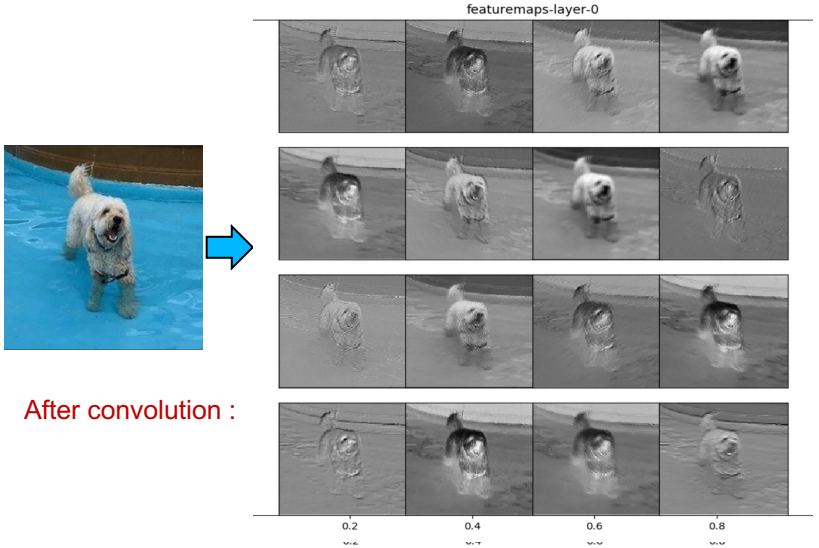
Hadronic calorimeter

- **Red circles are denoting electrons & Black rectangles are the jets at calorimeters above.**

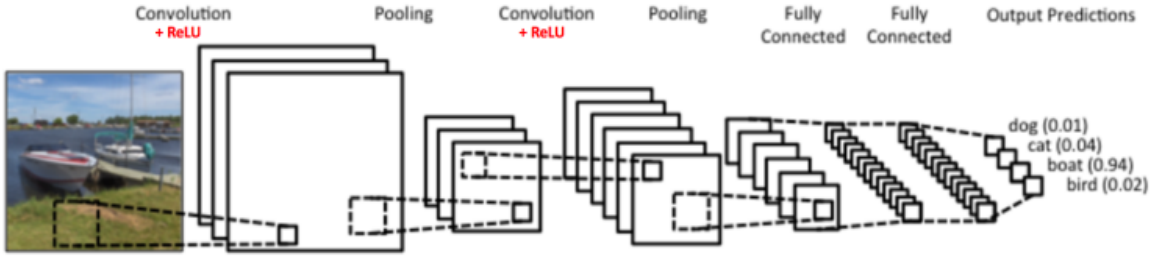
Description of Convolutional neural network



- In the CNN, after we provide inputs, we have different operations like Convolution, Activation, MaxPooling, Flattening, Densing, Drop out etc.
- Also it uses (different) 'Optimizers' to minimize the 'loss function' and to optimize the output of the model.
- It has some training and testing operations. And while training the neural net, there are operations of internal validation as well.



After convolution :



An example how image classification works using CNN

Ref.:https://indico.hep.anl.gov/indico/getFile.py/access?contribId=5&resId=0&materialId=slides&confId=1192

Our best model

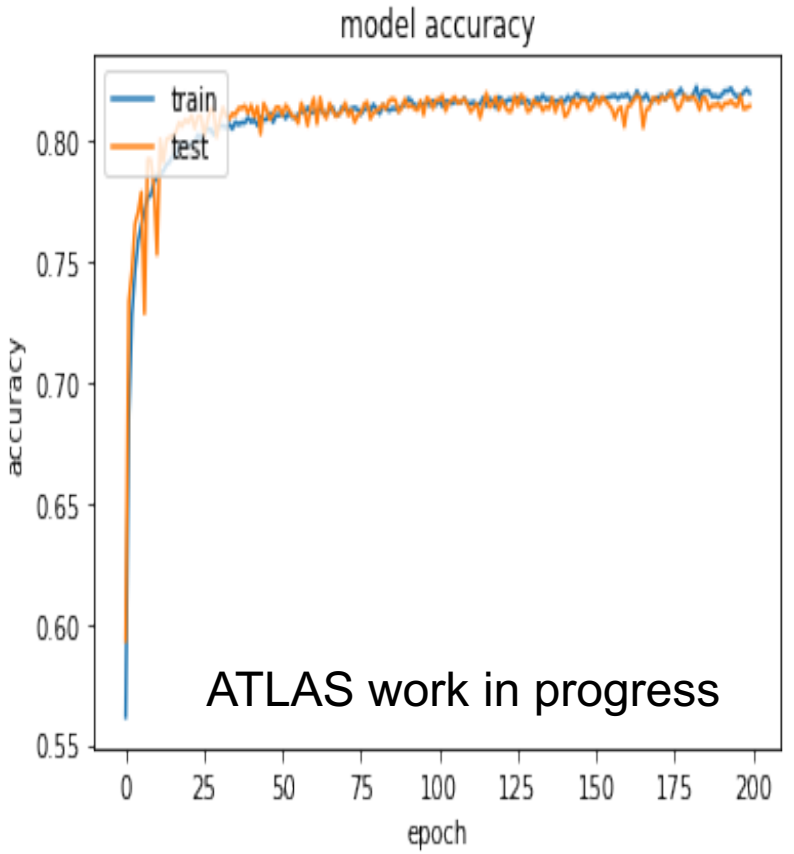
We played with following different parameters to find our best model:

Optimizers : RMSprop, SGD, Adam

Activation function: relu, selu

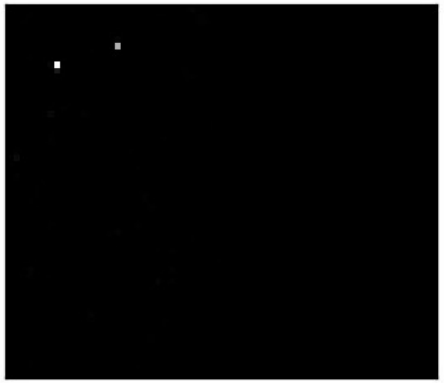
Loss funstion: Categorical Cross entropy, Mean Squared Error

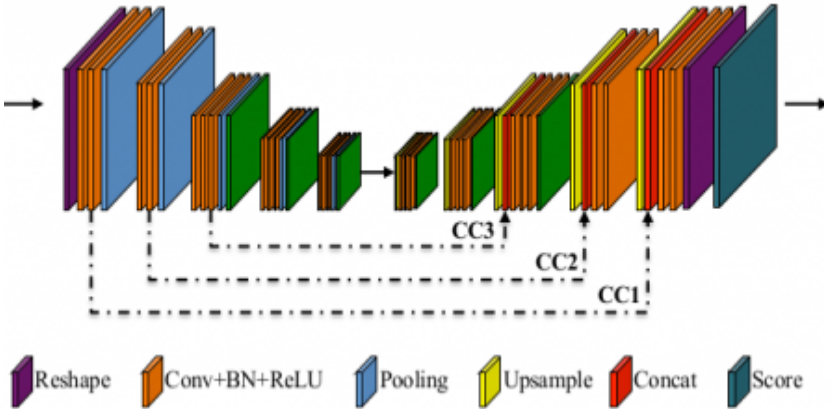
Learning rate, decay & structure of the neural network.

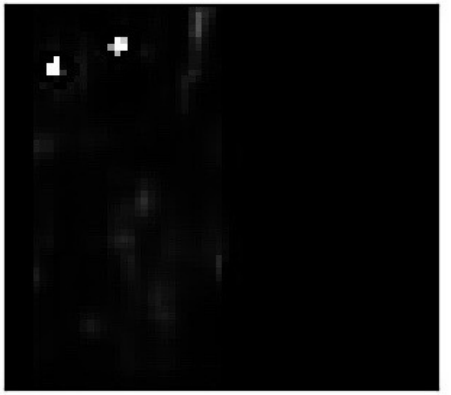


Auto-encoder

To deal with the sparse nature of ATLAS data, we have started investigating Auto-encoder. It will help us saving our computation time.





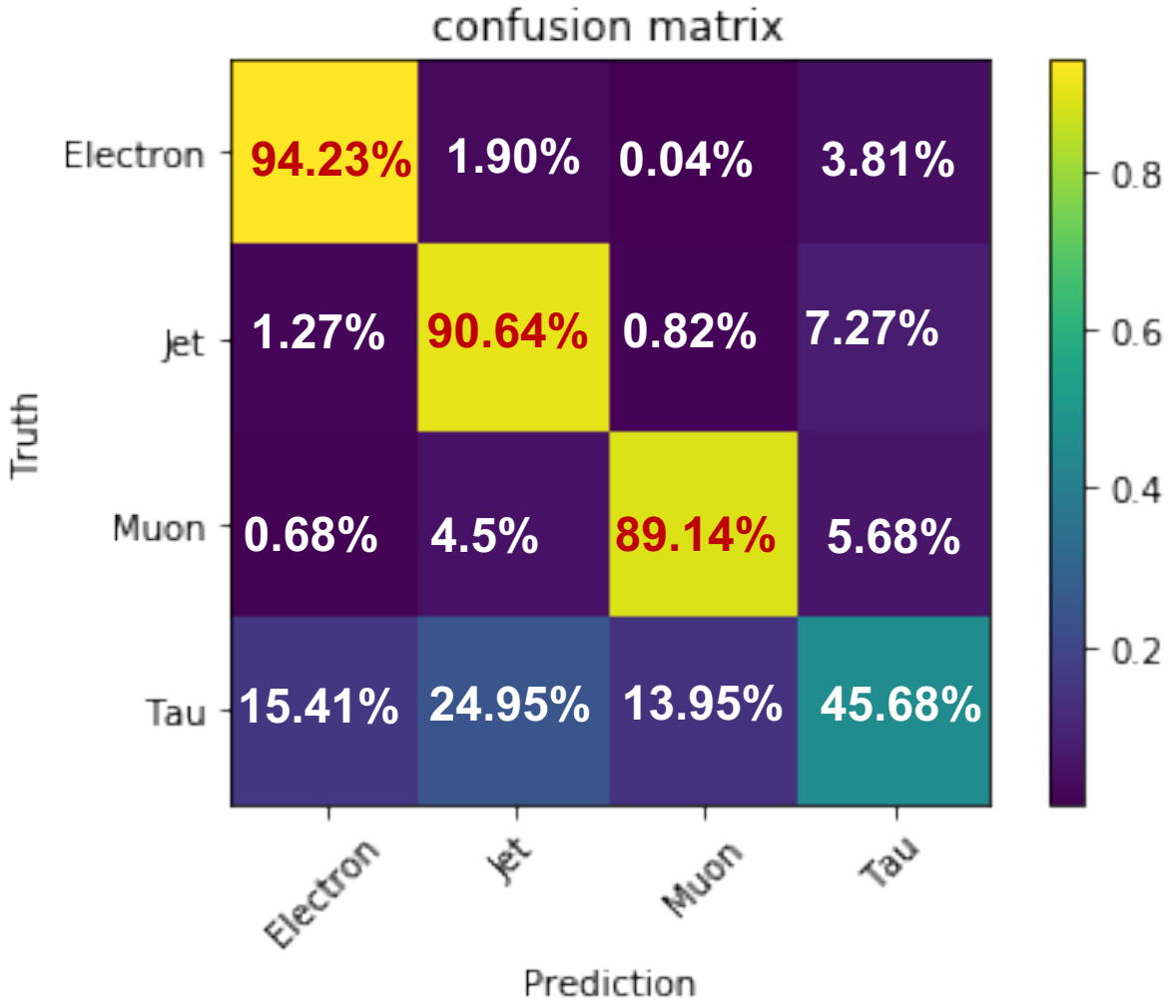


Original input images

Decoded images from the auto-encoder

ATLAS work in progress

Confusion Matrix from the our model



ATLAS work in progress

Conclusion & plans ahead…

- Our model of Deep Neural Network is working well for identifying leptons & jets with our ATLAS data.
- We are collaborating with Machine Learning experts from Argonne Leadership Computing Facility to build custom models.
- Adding Pile up to 2D studies to see how noise changes results
- We are also investigating classification of full ATLAS events.
- We also plan to build 3D representations of the ATLAS detector with tracker hits and calocell energy to study classification using 3D CNNs.


Thank You!

Backup slides

Optimization:

- Finding (one or more) minimizer of a function subject to constraints
- Most of the machine learning problems are, in the end, optimization problems.

Loss function: Categorical cross entropy :

For discrete p and q this means

$$H(p,q) = -\sum_{x} p(x) \log q(x).$$

Loss function: Mean Squared error:

$$\mathrm{MSE} = \frac{1}{n}\sum_{i=1}^{n}(\hat{Y}_i - Y_i)^2$$

Stochastic gradient descent

This method performs a parameter update for **each** training example $x^{(i)}$ and label $y^{(i)}$.

Update equation

$$\theta = \theta - \eta * \nabla_\theta J(\theta; x^{(i)}; y^{(i)})$$

We need to calculate the gradients for the whole dataset to perform **just one update.**

Advantage

It is usually **much faster** than batch gradient descent.

It can be **used to learn online.**

Disadvantages

It performs frequent updates with a **high variance** that cause the objective function to fluctuate heavily.

Ruder, Sebastian. "An overview of gradient descent optimization algorithms." *arXiv preprint arXiv:1609.04747* (2016).

RMSprop

RMSprop as well divides the learning rate by an exponentially decaying average of squared gradients.

RMSprop

$$E[g^2]_t = 0.9E[g^2]_{t-1} + 0.1g_t^2$$

$$\theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{E[g^2]_t + \epsilon}} g_t$$

Ruder, Sebastian. "An overview of gradient descent optimization algorithms." *arXiv preprint arXiv:1609.04747* (2016).

Adam

Adam's feature :

Storing an exponentially decaying average of past squared gradients v_t like Adadelta and RMSprop

Keeping an exponentially decaying average of past gradients m_t, similar to momentum.

Counteracting these biases in Adam

$$\hat{m}_t = \frac{m_t}{1-\beta_1^t}$$

$$\hat{v}_t = \frac{v_t}{1-\beta_2^t}$$

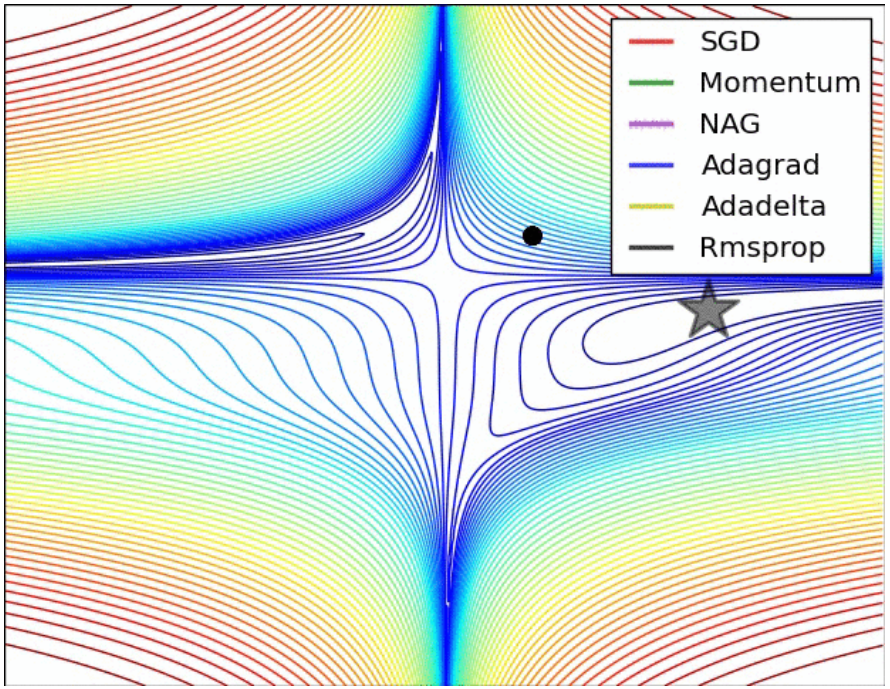
Adam

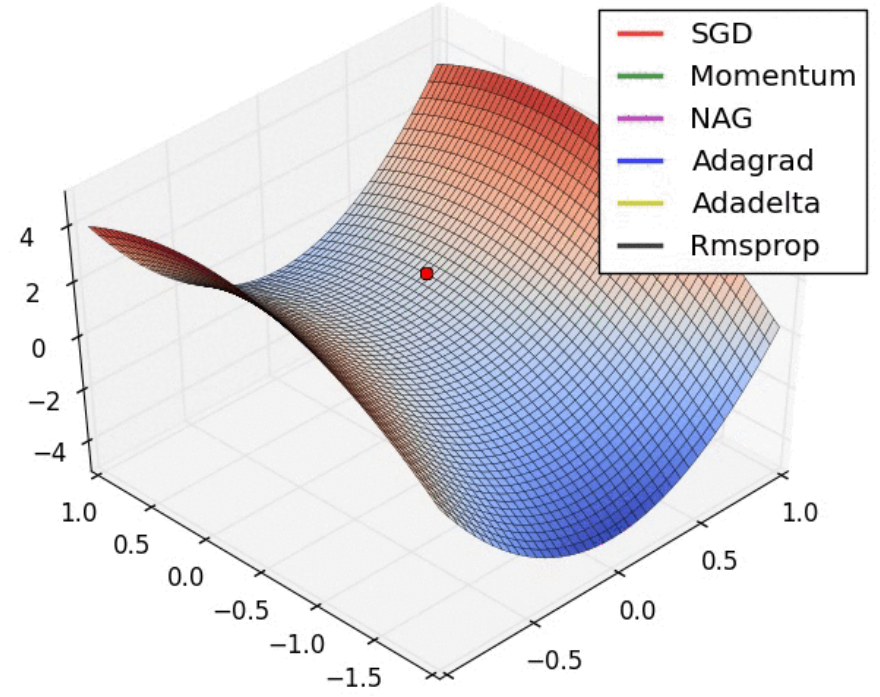
$$\theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{\hat{v}_t} + \epsilon}\hat{m}_t$$

Ruder, Sebastian. "An overview of gradient descent optimization algorithms." *arXiv preprint arXiv:1609.04747* (2016).

Visualization of algorithms

As we can see, Adagrad, Adadelta, RMSprop, and Adam are most suitable and provide the best convergence for these scenarios.





Ruder, Sebastian. "An overview of gradient descent optimization algorithms." *arXiv preprint arXiv:1609.04747* (2016).