Imagenet classification

No AI used

Paper Review

Summary & Review & Suggested improvements.

→ Current approaches used ML (small datasets)

→ For to detect/recognise objects in real life, we ~~used~~ must use larger dataset & more powerful Models → CNNs

→ We can vary the depth & breadth of CNNs

→ CNNs better than std. NN with similar sized layers as they are easier to train, best is slightly worse (performance)

→ But CNNs are expensive for high-reso. large scale images. So, we use GPUs + 2D conv. implementations We do not severely overfit.

→ ImageNet Dataset used → Human labelled, 1.2 million training, 50000 validation & 150000 test images.
↳ top 1 error & top 5 error.
rescaled to 256 × 256.

→ Architecture → 8 layers → 5 Conv.
→ 3 full connected.

• Neurons output in for input x is $f(x) = \tanh(x)$.
or $f(x) = \frac{e^x}{1+e^x}$ → Both are saturating.
as a result, they are slower than non saturating nonlinearity (ReLUs).
Deep CNN work with ReLUs train faster than equivalent tanh units.
ReLU faster than tanh neurons.

• net spread across two GPUs due to memory constraint
Used GPU parallelization (GPUs communicate only in certain layers).

• Two GPU net faster than one GPU net.

• Normalizations not required generally. But ~~BS~~ local normaliser is good for generalization.

The normalisation appears to be 2 dim. in nature.
Normalisation is also useful for covering the images of different lightning conditions → brightness normalization which improves performance.

- Overfitting is difficult for training models with overlapping pooling. I believe that, different parts of Neural Network would prepare different weights for the same pixel as a result of which we would get the averaged out value of the weights for that pixel. The error rate are also less.

- There are 5 CN Layers and 3 are fully connected last one has 1000 way Softmax → 1000 class labels → Maximises the multinomial logistic regression objective.

- 2nd, 4th & 5th CL layers connected to those kernels on the same GPU. 3rd CL layer connected to all kernel maps of layer 2. We are normalizing at 1st & 2nd CL. ReLU non linearity is applyied to every layer. This would make the computation faster as well as the running on parallised GPUs will further ↑ the speed. normalising will be crucial for ↑ the accuracy by bringing the brightness normalisation. I suggest that in order to improve the computation speed, we can directly go by doing the "histogram intensity transform" that would reduce the computations required by ↓ the normalisations as no learning would be involved in that.

- As given in the paper that they used 96 kernels of 11×11×3 for filtration followed by 256 kernels of 5×5×48.

I believe that this is computationally too expensive despite using optimised 2D convolutions. I suggest that we can go for the edge detectors over that image and get the binarised edges. Then, we can filter those images and get the filtered format into binarized pixel format. Then we can use some morphological operations over that image to ↓ the width of the boundary and prepare a binarized mask which would be superimposed onto the image rather than convolving. The advantages involve reduced computational power requirements with the similar standard of filtration. We can also enhance the performance of this by hyperspectral splitting as it would be personalised for almost every wavelength filtered. This will also bring down the number of neurons required by bringing down the number of layers. (Combining the operations of layer 1 & 2 into one layer performing the above mentioned operations).

→ Reducing overfitting.

To reduce overfitting, they are artificially enlarging the data. I believe that it would consume a lot of computational power towards processing the enlarged part of the data. Therefore, I suggest that the dimensions can be maintained the same with controlling the parameters using Grid Search CV. Another important point that can be mentioned is the dropout concept. It is true that using multiple models for ~~sets~~ prediction gives more accurate results.

I suggest that along with this, we can use an optimized method of ensemble learning with multiple models which will be enacted only in case of overfitting or underfitting instances which will be found by GridSearchCV. The important thing about this suggestion is that it will consume very less time as it would ~~enact the~~ implement all the models only in case of overfitting/ underfitting.

→ Results & Details of learning.

Coming to the end of this review, the accuracy of the model is quite appreciable. But I believe that by implementing the suggestion, the accuracy of the system will be ~~much~~ slightly higher compared to the values given in the paper.