(Established under Galgotias University Uttar Pradesh Act No. 14 of 2011)

ARTIFICIAL NEURAL NETWORK (ANN) DEVELOPMENT USING DEEP LEARNING

A Report for the Evaluation 3 of

Project 2

Submitted by

ABHISHEK

(1613101026 / 16SCSE101415)

in partial fulfillment for the award of the degree of

Bachelor of Technology

IN

Computer Science and Engineering

SCHOOL OF COMPUTING SCIENCE AND ENGINEERING

Under the Supervision of

Mr. Abhay Kumar,

Assistant Professor

APRIL / MAY- 2020

TABLE OF CONTENTS

Abstract

One of the classical problems in the field of computer vision and machine learning and subsequently deep learning is image classification. While Deep Learning solves the much difficult hurdles like feature extraction and presents us with better optimizations like gradient descent and Adam optimizer, most deep learning models still need a lot of raw computational power to train models on local Graphical Processing Units (GPUs) or Tensor Processing Units (TPUs) in the cloud. All of this computational power is not readily available in all environments and systems and hence the concept of pre-trained models can help to reduce training time by a huge margin. Initial models get trained on large array of GPUs and do feature extraction. The classification part is for the end-user to customise in accordance to the problem at hand and can be completed in very less time.

We tackled the multi-class classification botanical problem of identifying flowers of 5 types, namely, Sunflower, Rose, Dandelion, Daisy, and Tulip. The feature extraction part is done with the model (Google’s Inception-v3) and fully connected softmax layers were trained on local machine on a Nvidia GeForce GTX 950 (with CUDA activated) within 30 minutes time and total steps/epochs were 4000 only. The total number of training images is 3,500

(approx.). The finished model produced results with final test accuracy as 91.9% on new images (N=664).

Introduction

Convolutional neural networks (CNNs) opened a door to a new era of classification. Multi-class classification of the order of couple dozen classes is now possible. Using existing CNN model of layering can be efficient as well as robust in classification tasks, but requires a lot of data and computational power to classify images with higher degree of accuracy. This hardware include GPUs, FPGAs, and ASICs like Google's TPU and IBM TrueNorth [1]. Despite the attractive qualities of CNNs, and despite the relative efficiency of their local architecture, they have still been prohibitively expensive to apply in large scale to high-resolution images. The feature extraction task taken at hand in this paper is done by using Inception-v3 [2].

The network training process is then relatively simpler with a GPU and the final model for a custom use case is ready after training on custom classes for approx. 30 min to 1 hr. This paper does it for five flowers' 3.500 images with each class having approx. 700 images but it can be extended to any multi-class classification problem with no additional overhead.

Existing System

In Liu. et.al’s work, Noval general K-nearest neighbour classifier GKMNC (fuzzy k-nearest) was used for visual classification. Sparse representation-based method described in Tropp et.al. for learning and deriving the weights coefficients and FISTA (Beck et.al.) was used for optimization. CNN-M (in Chatfield et.al.), a pretrained CNN was used for image features extractions then marginal PCA (Jolliffe et.al.) is applied to reduce the dimension of the extracted features. In Sun et.al., AlexNet (Hinton et.al.) model, a deep neural network is used to learn scene image features. During the training phase, series of transformation such as convolution, max pooling, etc are performed to obtained image features. Then two classifier SVM (Cortes et.al.) classifier and Softmax (Nasrabadi et.al.) classifier are trained using extracted features from the AlexNet model. In Zhang et.al., Spatial pyramid pooling was used in CNN to eliminate the fixed size input requirements. for this new network structure SPP-net was used, which can generate a fixed length representation regardless of image size. Standard back propagation algorithm was used for training, regardless of the input image. In Fritz et.al., Kernalized version of Naive bayes Neabour (Murphy et.al.) was used for image classification and

SVM (Cortes et.al.) classifier was trained on Bag-Of-Features (Z.S. Harris) for visual classification.

In Cord et.al., Extension of the HMAX (Poggio et.al.), a four level NN has been used for image classifications. The local filters at first level are integrated into last level complex filters to provide a extensiible description of object regions. In Zisserman er.al., Nearest neighbor classifier (Murphy et.al.) was used for visual classifications. SIFT (D. G. Lowe) descriptor to describe shape, HSV (A. R. Smith) values to describe colours and MR filters to describe texture were used.

Proposed System

Transfer Learning –using existing neural network architecture and retraining portion of it to a particular use case. The network remembers the older learnings and use them to learn newer ones.

•E.g. Learn to ride a bicycle ----helps---> Learn to ride a motorcycle

•This solves the problem of having to train the model on computationally expensive hardware (e.g. GPUs) for a long time.

•Because, *feature extraction* task is taken care of by the pre-trained off the shelf CNN architecture being utilised.

•The *classification*task is done by re-training the last layers of the model for a given use case.

Implementation and Architecture Diagrams

There are many CNN architectures being used today. Ex-ResNet, Inception-v1, Inception-v3, AlexNet, VGGNet, etc.

•The CNN model designed here is based on inception-v3. It is a 42 layer deep, pretrained CNN model trained on ImageNet (12 million images in 1,000 classes), which was 1st runner up in ImageNet Large Scale Visual Recognition Competition, ILSVRC (2012).

•The inception model has two parts; *feature extraction*and *classification*. we make use of the feature extraction part of inception model and retrained the classification layer with our own dataset.

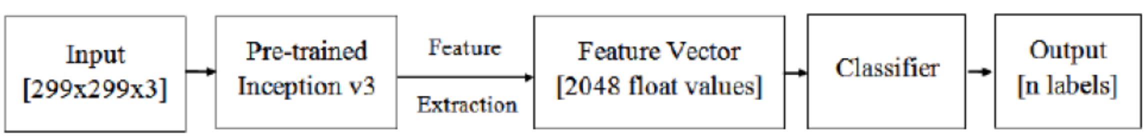

Fig. 2. Transfer Learning Implementation Pipelines.

A. Feature Extraction Task

One of the most computationally intensive tasks in neural network approach is feature extraction. The transfer learning model aims to make this task independent of the use case. One of the ways it does that is to exhaustively train (primary) a custom CNN architecture on millions of images and labels.

A number of such models are already available e.g. AlexNet, VGGNet, Google LeNet/Inception-v1, ResNet.

B. Classification Task

The model is then exported and then subjectively trained (secondary) for a given use case. This training will not require an elaborate dataset, nor will it require powerful computational hardware. The model does not perform feature extraction again during this secondary training but trains and stores weights for new use case among the softmax layers.

Result

The learning method used in this experiment is supervised learning. The network is trained with stochastic gradient utilizing the TensorFlow distributed machine learning system using a Nvidia GeForce 950 GTX GPU with batch size 100 for 4,000 epochs. Training set has 90 percent images and Evaluation set has two subsets, test subset has 10 percent of images and validation subset has 10 percent of images, both amounting to approx. 367 images each. Training learning rate = 0.01, training interval = 10. These hyperparameters are customizable in *retrain.py* file.

Training accuracy was 89.7% at the beginning of the training process and starts to increase, after completion of all training steps it reached to 98.0%. Validation accuracy was 75.8% during initiation of training and validation process and final validation accuracy was 93.0%. Final training accuracy was 91.9.

The model is able to classify class label images with high precision.

Below image of dandelion tested produces the following results –

Evaluation time (1-image): 1.387s

dandelion (score=0.99877)

daisy (score=0.00097)

tulips (score=0.00018)

sunflowers (score=0.00007)

roses (score=0.00001)

Another image test results – Evaluation time (1-image): 1.231s

sunflowers (score=0.93991)

dandelion (score=0.03731)

daisy (score=0.01330)

tulips (score=0.00671)

roses (score=0.00277).

Conclusion

In this paper, the classification layers (fully-connected and softmax) of pre-trained Inception-v3 model was re-trained successfully by implementing transfer learning technique. The model yields a final test accuracy of 91.9 percent on 5 classes of flower images dataset. Further this model can be used to retrain for much more classes within small time and with reasonable computational hardware to classify plants and other flowers' images as well as other objects.

References

[1] Y. LeCun, "1.1 Deep Learning Hardware: Past, Present, and Future," 2019 IEEE International Solid- State Circuits Conference - (ISSCC), San Francisco, CA, USA, 2019, pp. 12-19.
[2] Christian Szegedy and Vincent Vanhoucke and Sergey Ioffe and Jonathon Shlens and Zbigniew Wojna, “Rethinking the Inception Architecture for Computer Vision” in 1512.00567- arXiv, 2015
[3] Q. Liu, A. Puthenputhussery, and C. Liu, Novel general knn classifier and general nearest mean classifier for visual classification," in Image Processing (ICIP), 2015 IEEE International Conference on. IEEE, 2015, pp. 1810-1814
[4] J. M. Keller, M. R. Gray, and J. A. Givens, A fuzzy k-nearest neighbor algorithm," IEEE transactions on systems, man, and cybernetics, no. 4, pp. 580-585, 1985.
[5] J. A. Tropp, Greed is good: Algorithmic results for sparse approximation," IEEE Transactions on Information theory, vol. 50, no. 10, pp. 2231-2242, 2004.
[6] A. Beck and M. Teboulle, A fast iterative shrinkage-thresholding algorithm for linear inverse problems," SIAM journal on imaging sciences, vol. 2, no. 1, pp. 183-202, 2009.
[7] K. Chatfield, K. Simonyan, A. Vedaldi, and A. Zisserman, Return of the devil in the details: Delving deep into convolutional nets," arXiv preprint arXiv:1405.3531, 2014.
[8] I. Jolliffe, Principal component analysis," in International encyclopedia of statistical science. Springer, 2011, pp. 1094-1096.
[9] J. Sun, X. Cai, F. Sun, and J. Zhang, Scene image classification method based on alex-net model," in Informative and Cybernetics for Computational Social Systems (ICCSS), 2016 3rd International Conference on. IEEE, 2016, pp. 363-367.
[10] A. Krizhevsky, I. Sutskever, and G. E. Hinton, Imagenet classification with deep convolutional neural networks," in Advances in neural information processing systems, 2012, pp. 1097-1105.
[11] C. Cortes and V. Vapnik, Support-vector networks," Machine learning, vol. 20, no. 3, pp. 273-297, 1995.
[12] N. M. Nasrabadi, Pattern recognition and machine learning," Journal of electronic imaging, vol. 16, no. 4, p. 049901, 2007.
[13] K. He, X. Zhang, S. Ren, and J. Sun, Spatial pyramid pooling in deep convolutional networks for visual recognition," in European conference on computer vision.Springer, 2014, pp. 346-361.

[14] T. Tuytelaars, M. Fritz, K. Saenko, and T. Darrell, The nbnn kernel," in Computer Vision (ICCV), 2011 IEEE International Conference on. IEEE, 2011, pp. 1824-1831.
[15] K. P. Murphy et al., Naive bayes classifiers," University of British Columbia, vol. 18, 2006.
[16] Z. S. Harris, Distributional structure," Word, vol. 10, no. 2-3, pp. 146-162, 1954.
[17] C. Theriault, N. Thome, and M. Cord, Extended coding and pooling in the hmax model," IEEE Transactions on Image Processing, vol. 22, no. 2, pp. 764-777, 2013.
[18] M. Riesenhuber and T. Poggio, Hierarchical models of object recognition in cortex," Nature neuroscience, vol. 2, no. 11, p. 1019, 1999.
[19] M.-E. Nilsback and A. Zisserman, A visual vocabulary for flower classification," in Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on, vol. 2. IEEE, 2006, pp. 1447-1454.
[20] D. G. Lowe, Object recognition from local scale-invariant features," in Computer vision, 1999. The proceedings of the seventh IEEE international conference on, vol. 2. Ieee, 1999, pp. 1150-1157.
[21] A. R. Smith, Color gamut transform pairs," ACM Siggraph Computer Graphics, vol. 12, no. 3, pp. 12-19, 1978.
[22] Krizhevsky, Alex & Sutskever, Ilya & Hinton, Geoffrey. (2012). ImageNet Classification with Deep Convolutional Neural Networks. Neural Information Processing Systems. 25. 10.1145/3065386.
[23] Simonyan, Karen & Zisserman, Andrew. (2014). Very Deep Convolutional Networks for Large-Scale Image Recognition. arXiv 1409.1556.
[24] C. Szegedy, Wei Liu, Yangqing Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, “Going deeper with convolutions,” 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Jun. 2015.
[25] K. He, X. Zhang, S. Ren and J. Sun, "Deep Residual Learning for Image Recognition," 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, 2016, pp. 770-778.
[26] J. Deng, W. Dong, R. Socher, L. Li, Kai Li and Li Fei-Fei, "ImageNet: A large-scale hierarchical image database," 2009 IEEE Conference on Computer Vision and Pattern Recognition, Miami, FL, 2009, pp. 248-255.

[27] S. B. Kotsiantis, I. Zaharakis, and P. Pintelas, Supervised machine learning: A review of classification techniques," Emerging artificial intelligence applications in computer engineering, vol. 160, pp. 3-24, 2007.
[28] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mane,´ R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viegas, O. Vinyals, P. War- ´ den, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. Software available from tensorflow.org