Reflective Journal on the "Chihuahua vs. Muffin" Image Classification Workshop

Summary -

The objective of the workshop "Chihuahua vs. Muffin" was to design a neural network that would be able to separate the two classes of images – Chihuahuas and muffins now known as 'muffin', to expand the knowledge of participants on the prototype classification task. This task brought forth the difficulties which come with the classification of images, namely the need to capture tiny and subtle differences separating up the classes. As the workshop progressed, we made use of "PyTorch" to build, train as well as test a basic binary feed forward deep neural network.

There were two phases in our study design: the datasets were first divided into training and validation sets and screening was carried out on the dataset. Before the neural network was trained, all the input images in the dataset were cloud code enhanced using some transformations such as resizing and normalization. After the respective parts of the network - the input layer, hidden layer/s and output layer were laid out, the model was trained for several epochs. Lastly, the effectiveness of the network was studied by determining and analyzing the accuracy and error rates of the models' performance on the validation data set and displaying some predictions of the model.

Key Concepts Learned -

Several fundamental topics in machine learning and deep learning were consolidated through this workshop. First, image classification is a process in which an image is provided with a label depending on its content. In this instance, the target labels were 'Chihuahua' or 'Muffin'. This demanded the model to recognize the minor details which separated the pictures.

There is also a comfort level with neural networks for me, especially in terms of how to construct them in PyTorch. The_model_was_basic_with three_layers_fully_connected_and the ReLU activation was used as before. This forward pass took an input and fed it through these layers, and the data was changed into predictive output.

Apart from these, I also studied the data loading and data preparation. For datasets used in PyTorch, a preprocessed file is required that the model can handle – in this case, a tensor. With the help of the torchvision.transforms module, we changed the size of images and adjusted their pixel values to specific values. This means that there will be proper data fed into the neural network.

Challenges Encountered -

One of the initial hurdles I faced can be attributed to the task of learning how to organize all the elements regarding the neural network architecture. It was hard to choose the number of layers and the number of neurons, and the first version of the model was rather inaccurate. To counter this, I made modifications to the architecture and adjusted the learning rate of the optimizer on an experimental basis. This took some iterations to strike the right chord between the two extremes of bias and variance.

Another difficulty that I struggled with was getting the validation accuracy to be very high. After a few epochs, on some occasions, the model would reach a plateau and not learn any further. I tackled this issue by varying the batch size to a number suitable and moderately changing the learning rate. This was significant because the enhancement of model performance was related to how well these hyperparameters complimented each other.

Insights Gained -

This workshop proved insightful as it presented the details of training machine learning models for image classification. I understood that model performance depends significantly on the choice of these hyperparameters: learning rate and batch size. The variations within these parameters are small; however, they have a significant impact on how fast and efficiently you train your model as well as on the validation accuracy. The second big thing that was mentioned was data augmentation was an essential precondition of the efficiency of the method. Although not tried in this exercise, extra transformations such as random flips or even rotations may enhance model robustness as it is presented with different sets of inputs during training. This is especially useful when there is noise in the data, something which is always common in real life. In addition, the current advanced models of deep learning are very efficient, but they demand a lot of computation and tuning to be efficient enough. Even the simplest examples of image classification show that this problem cannot be solved without a highly effective method of data preprocessing and the proper selection of the network architecture.

Real-World Applications -

It should be noted that all the techniques received in this work can be useful for addressing various problems in image classification. For example, medical imaging, which is a significant application of computer vision, uses image classification to decode a picture, for example, an X-ray or MRI look for diseases or distortions. Just like that, self-driving cars employ neural networks to categorise objects in the road environment, such as people on the road or road signs to drive efficiently.

Other uses include identification facial recognition systems, recommendation system which rely on image similarity between products, security surveillance systems that use image patterns to detect unusual activity. Neural networks and image classification are not limited to just distinguishing between a Chihuahua and a Muffin and the fundamentals that work behind such applications support other and more complex programmers of artificial intelligence.

Personal Reflection -

In general, the given workshop can be considered one of the most valuable experiences. Reading this code just helped me concretize each point right from loading data to training up to the prediction in building a neural network. Perhaps the most gratifying, was the noticeable enhancement of the model after calibration and training had been done multiple times. This practical session has certainly enlightened me about the many aspects of the machine learning which I would not be able to achieve in the abstract mode, as well as provided me with the confidence to undertake a similar assignment in the future. The workshop also emphasized the fact that prediction with machine learning models requires patience and trial & error. Again and again, theory as a general guide, or at least it is expressed in terms of trial and success. This workshop has encouraged me to learn more about deep learning namely other architectures more suited to images than this task such as CNNs.

References -

Paszke, A., et al. (2019). PyTorch: An Imperative Style, High-Performance Deep Learning Library. Advances in Neural Information Processing Systems, 32, 8024-8035.

Goodfellow, I., Bengio, Y., & Courville, A. (2016). Deep Learning. MIT Press.

PyTorch Documentation: Transforms. (n.d.). Retrieved from https://pytorch.org/docs/stable/torchvision/transforms.html

Bishop, C. M. (2006). Pattern Recognition and Machine Learning. Springer.