Project Name: *Semi-Supervision (Unlabeled Data) & Self-Supervision Improve Class-Imbalanced / Long-Tailed Learning using real time analytics with spark.*

Problem Statement:

To study and explore the imbalanced learning, value of labels in the class-imbalanced learning, whether it is useful or not, how it impacts the performance of the models, negatively and positively and to prove the theoretical hypothesis that imbalance labels are valuable but not always.

Dependencies: (to be installed)

1. PyTorch (version >=1.2)
2. yaml
3. scikit-learn
4. TensorBoardX

All the output files, checkpoints, data and repo files can be found in the Drive Link.

Implementation:

Read the description of the github repo for clear understanding of important files in the repo and for main arguments.

- Download/Clone the repository. It is best to upload it to the drive folder “BDAProject” for easy access while implementing it in colab, so we uploaded it in drive.
- Download the datasets mentioned in the description, following the links given there and upload it to the new folder “data” inside of BDAProject folder.
- Now, For semi-supervised imbalance learning (\rho = 50 and \rho_U = 50).

    1. Implement the following code for creating pseudo labels.
       To perform pseudo-labeling (self-training), first a base classifier is trained on the original imbalanced dataset. With the trained base classifier, pseudo-labels can be generated using

       “python gen_pseudolabels.py --resume <ckpt-path> --data_dir <data_path> --output_dir <output_path> --output_filename <save_name>”

Where,
ckpt-path = checkpoint directory(initially we won't have one, so can remove that argument)
data_path = './data'
output_path = the path where you want to store the pseudo label files
Save_name = output file name

Or

you can directly use the pseudo label files that were provided after training base models with standard cross-entropy (CE) loss.

2. To Train with the unlabeled data.

   "python train_semi.py --dataset cifar10 --imb_factor 0.02 --imb_factor_unlabel 0.02"

3. To Test a pre-trained checkpoint. (checkpoint is created in the second step).

   "python train_semi.py --dataset cifar10 --resume <ckpt-path> -e"

   where, ckpt-path = give the best path from above created checkpoint

The above 3 steps are implemented on the **cifar10 dataset,** just replace cifar10 with '**svhn**' and the pseudo label file for '**svhn**'.

- For self-supervised Imbalanced Learning, (\rho = 100)

  1. First perform Rotation SSP, pretrain the model on the dataset without considering labels.

     "python pretrain_rot.py --dataset cifar10 --imb_factor 0.01"

  2. Network Training with SSP Models.

     "python train.py --dataset cifar10 --imb_factor 0.01 --pretrained_model <path_to_ssp_model>"

     where, path_to_ssp_model = best checkpoint path created in the step-1(pre-training).

3. Test a pre-trained checkpoint.

   "python train.py --dataset cifar10 --resume <ckpt-path> -e"

   where, ckpt-path = best checkpoint path created in the 2nd step.

For the cifar100 dataset, replace **cifar10** with **cifar100** in the above 3 steps.

Observations:

- Used Semi-Supervised and Self-Supervised methods for the imbalanced Learning.
- If the models are supervised, like they are trained on the labeled data, in almost all cases the data in real life is not fair, not distributed properly among all labels, so models suffer from the label bias. Here imbalanced labels are valuable, have a positive impact. With more unlabeled data, the original labels can be leveraged with the extra data, by employing a simple pseudo-labelling strategy to reduce label bias in a semi-supervised manner, which greatly improves the final classifier.
- However if models are not trained on the labeled dataset initially and are pre-trained in a self supervised manner based on the imbalance factors and probabilities of the distributions(here gaussian distribution is used), then introducing the more unlabelled data (imbalanced data) has negative impact on the final result. Using the above idea, Classifiers that are pre-trained in a self-supervised way are found to be outperforming their corresponding baselines. (without using any extra data and abandoning the labels at the initial pre-trained stage)
- **Conclusion**: As discussed earlier, we can conclude that imbalanced unlabeled data has positive impact and negative impact, both can be utilized properly and explored further for improvement(better) of performance, and are promising directions for improving class-imbalanced learning.

Challenges Faced:

1. Finding and Downloading the large datasets, Limited resources are available for the implementation(ex: GPU for processing the large amount of data).
2. Tried using kaggle for the ImageNet dataset but unable to do so because of OS errors(read-only), as we can't change the files in the input folder in kaggle, which we want to do. We are not able to implement on the ImageNet and iNaturalist datasets in self-supervised imbalanced learning due to the lack of resources.
3. We used Google Colab to implement the code and Not Enough GPU to implement the code for all the datasets, did the execution of code in parts, in different accounts for each dataset and uploaded the outputs to drive.

4. Change **“view”** to **“reshape”** in the utils.py file in line 203 as their functionalities are changed and also reduced one pooling layer in “resnet” model as input size and output we want are not matching(as it is becoming to reduced due to the extra avg pooling layer).
5. Should be careful while giving directory paths while implementing the code.

Thank you

Team Members: (Group 12)

Jayanth Dasari, 2019BCS-016,
Satya Pavan Kalyan Vemula, 2019BCS-069,
Mahesh Bommisetty, 2019BCS-013,
Anunay Nalam, 2019 BCS-034.