LEARNING ROBUST REPRESENTATIONS BY PROJECTING SUPERFICIAL STATISTICS OUT
ICLR 2019 REPRODUCIBILITY CHALLENGE

Marcin Witkowski & Łukasz Klasiński
Institute of Computer Science
University of Wroclaw
Wroclaw, Poland
{288840,290043}@uwr.edu.pl

ABSTRACT

The main goal of reproduced paper is to build a classifier that is not susceptible to covariance shifts, thus making it better in domain generalization. In their work authors introduce new neural building block - NGLCM, and a method to project out textural information learned by CNN - HEX. In this paper we tried to reproduce said method and compare results by training model on PACS (Li et al., 2017), MNIST and FERG-DB (Aneja et al., 2016) datasets.

1 METHOD

1.1 NGLCM

NGLCM is a neural block that mimics *gray-level co-occurrence matrix* (GLCM) (Lam, 1996). Main difference is that NGLCM can be trained like any other NN layer during back-propagation, allowing to tune it's parameters. Thanks to that, this block can extract textural information about image but is not capable of extracting semantic information. This is used later to unlearn classifier layer from associating textual noises to labels.

1.2 HEX

The main idea of HEX is to project predictions based on all information about data onto subspace that is orthogonal to the ones based on textual-only description of input. As a result we should obtain predictions which are more independent to covariance shifts, thus more reliable in domain generalization. It is achieved be using three different outputs:

$$F_A = f([h(X;\theta), g(X;\phi)];\xi)$$
$$F_G = f([\mathbf{0}, g(X;\phi);\xi])$$
$$F_P = f([h(X;\theta), \mathbf{0}];\xi)$$

where F_A, F_G, F_P stands respectively for the results from all representation, only textural representation and raw data.

Projecting F_A onto the subspace that is orthogonal to F_G with

$$F_L = (I - F_G(F_G^T F_G)^{-1} F_G^T) F_A$$

yields F_L for parameter tuning. In testing time F_P is used instead of F_L. More information about rationale of this method can be found in appendix of original paper Wang et al. (2019)

2 IMPLEMENTATION

Everything was implemented using PyTorch, open-source machine learning library for Python. Since no code was provided with paper, we had to write everything from scratch. We have performed all tests using Jupyter Notebooks so they are easy to repeat at will.

We have implemented NGLCM block correspondingly to the paper, although we had to guess what direction was used in GLCM matrix (we used default of 0), as well as we had to guess MLP layer shape. Another assumption was that the output of NGLCM is 16×244 matrix (16×16 pushed through MLP layer), but we somehow had to make it a vector. To accomplish that we 'squashed' it to $16 \cdot 244$ length vector to further concatenate it with AlexNet output and pass to HEX's classifier.

In HEX's case, we calculated F_A, F_G and F_P as in paper and finally computed F_L. Then results from F_L are passed through SoftMax layer.

All of the code is available on our GitHub[1]

3 TESTS

3.1 MNIST TESTS

Firstly, we generated 6 datasets, where each of them has 100 images per label, that is 10000 images in total. Subsequent datasets have images rotated by an angle $\alpha \in \{0°, 15°, 30°, 45°, 60°, 75°\}$, each of them representing one domain. All domains but one was used for training and the remaining for testing.

Hyperparameters

- optimizer: Adam
- learning rate: 10^{-3}
- weight decay: 10^{-3}
- 50 epochs

Table 1: MNIST tests results

Test domain	ADV (Wang et al., 2019)	HEX (Wang et al., 2019)	Our HEX
$\mathcal{M}_{0°}$	91.1	90.1	92.3
$\mathcal{M}_{15°}$	98.2	98.9	98.7
$\mathcal{M}_{30°}$	98.6	98.9	97.1
$\mathcal{M}_{45°}$	98.7	98.8	97.9
$\mathcal{M}_{60°}$	98.4	98.3	96.5
$\mathcal{M}_{75°}$	92.0	90.0	93.1
Avg.	96.2	95.8	95.9

We managed to reproduce results obtained in original paper quite faithfully (Table 3 (Wang et al., 2019)).

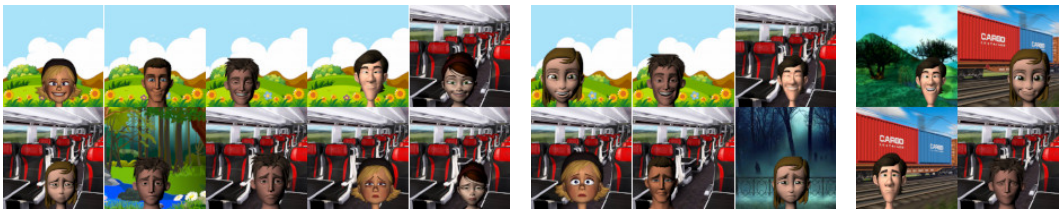
3.2 FERG-DB TESTS

Hyperparameters

- optimizer: Adam
- learning rate: $5 \cdot 10^{-4}$
- 70-100 epochs

[1] https://github.com/MarWit/nn_project

As we did not have access to original dataset used in paper, we generated set of 10 datasets based on clean FERG-DB. Each of them is created with background correlation $\rho \in [0.0, 0.9]$, which describes how one emotion is correlated with it's background (there are different backgrounds for different emotions). While we used the same partitioning of data (50% train, 30% valid, 20% test), each of the datasets had ≈ 10000 images instead of 50000. Sample images from dataset ($\rho = 0.8$) are presented below.



(a) train set (b) validation set (c) test set

We have tested two models on this dataset: normal HEX and modified HEX, where we replaced whole NGLCM block with single 224×16 MLP layer. In paper, tests with this modification are called 'Ablation tests'.

HEX training was converging quickly to 100% accuracy, so we decreased number of epochs to 70 (from 100 in paper) to save time. We also noticed, that during Ablation tests we sometimes had to rerun training, because model over-fitted on test data too fast, therefore model could not reach good results on validation set. Also, we were not sure about number of parameters MLP layer should have, so we picked 224×16 since it is a bit less then number of parameters for NGLCM.

Table 2: FERG-DB tests results

ρ (background correlation)	Our HEX with MLP	Our HEX
0.0	98.9	99.4
0.1	99.2	99.6
0.2	98.8	99.7
0.3	98.2	99.4
0.4	99.0	99.1
0.5	99.0	99.5
0.6	98.0	99.5
0.7	87.2	98.4
0.8	91.4	98.4
0.9	49.2	91.6
Avg.	91.9	98.5

HEX with NGLCM is much more stable than bare CNN, yielding similar results to those from paper (Figure 3. (Wang et al., 2019)). Simple MLP layer isn't capable of removing background noise as well as NGLCM block.

3.3 PACS TESTS

Similar to MNIST, we divided images from PACS into 4 domains representing art, cartoon, photo and sketch styled images. One domain was used for testing and remaining ones for training. We also added random data augmentation (crops and vertical flips) to prevent over-fitting.

First of all, we trained clean AlexNet.

Hyperparameters Alex

- optimizer: Adam
- learning rate: 10^{-3}
- weight decay: 0 or 10^{-2}
- 100 epochs

We wanted do get the best results we could, so we used a two step learning. Firstly we have loaded AlexNet pre-trained on ImageNet dataset (provided by torch library) and trained only classifier. Then, to fine tune the network, we have added weight decay to optimizer and trained all parameters for additional 50 epochs. We saved model that got best results during this two stage training and calculated accuracy.

After getting somehow positive results from AlexNet, we have tested if HEX would perform better or similar using same datasets.

Hyperparameters HEX

- optimizer: Adam
- learning rate: 10^{-5}
- weight decay: 10^{-5}
- 10+100 epochs

Following the paper, we firstly trained classifier layer of pre-trained AlexNet by 10 epochs, then we plugged it into HEX and trained for additional 100 epochs. At the beginning we had problems with really unstable gradient, so we had to turn learning rate drastically down in comparison to the one used in AlexNet. Nonetheless, we didn't really get satisfying results no matter what strategy we used. At the end, we came to conclusion that in original experiment special PACS heuristics, described in this paper (Li et al., 2017) was used. Unfortunately we didn't have enough time to study and replicate such learning method.

Table 3: PACS domain tests results

Domain	AlexNet (Wang et al., 2019)	Our AlexNet	HEX (Wang et al., 2019)	Our HEX
Art	63.3	57.2	66.8	47.6
Cartoon	63.1	61.3	69.7	65.0
Photo	87.7	81.3	87.9	53.3
Sketch	54.0	62.0	56.3	57.8
Avg.	67.0	65.5	70.2	55.9

We have different results in comparison to those from the paper, especially HEX ones. (Table 4 (Wang et al., 2019)). Still we think it is strange that under the same conditions HEX performed far worse than AlexNet, which differ from results of Wang et al. (2019).

4 CONCLUSION

During reproducibility challenge, we have encountered some minor and major problems. Paper did not contain most of information about used hyper-parameters and some aspects of described methods were omitted (ex. used heuristics or how to vectorize output of NGLCM). Overall, we were able to reproduce results of most of the experiments, besides PACS where we used different heuristic for fine-tuning CNN.

REFERENCES

Deepali Aneja, Alex Colburn, Gary Faigin, Linda Shapiro, and Barbara Mones. Modeling stylized character expressions via deep learning. In *Asian Conference on Computer Vision*, pp. 136–153. Springer, 2016.

S.W.-C Lam. Texture feature extraction using gray level gradient based co-occurence matrices. volume 1, pp. 267 – 271 vol.1, 11 1996. ISBN 0-7803-3280-6. doi: 10.1109/ICSMC.1996.569778.

Da Li, Yongxin Yang, Yi-Zhe Song, and Timothy Hospedales. Deeper, broader and artier domain generalization. In *International Conference on Computer Vision*, 2017.

Haohan Wang, Zexue He, and Eric P. Xing. Learning robust representations by projecting superficial statistics out. In *International Conference on Learning Representations*, 2019. URL `https://openreview.net/forum?id=rJEjjoR9K7`.