UiT
THE ARCTIC
UNIVERSITY
OF NORWAY

Home Exam

Home Exam in	**FYS-3033 - Deep Learning**
Hand-out:	**Monday March 28, 2022, 09:00**
Hand-in:	**Thursday April 28, 2022, 13:00**

The Home Exam contains 6 pages including this cover page

Contact person:	**Michael Kampffmeyer**
Email:	**michael.c.kampffmeyer@uit.no**

UiT THE ARCTIC UNIVERSITY OF NORWAY

Before You Start

Portfolio instructions

Your code should be submitted together with your report (see instructions below). **Further, please include a discussion of the results obtained and a discussion of the implementation in your report, which show that you understand what you are doing.**

The code should be commented in such a way that any person with programming knowledge should be able to understand how the program works. Like your report, the code must be your own individual work.

You are permitted to use deep learning frameworks such as Pytorch and Tensorflow. As there is a lot of code available online, please make sure that your report and code clearly show that you understand what you are doing.

Hand-in format

Please submit your report (in pdf format) to WISEflow and attach *one* single `.zip` file that contains two folders, one called `doc` that contains your report, and another one called `src` containing the code. The file name of the `.zip` file should follow the format `homeexam_candidate`*`XX`*`.pdf` (replace *`XX`* with your candidate number obtained from WISEflow) for anonymity.

Please include your candidate number and the course name on the frontpage of your report.

Follow the hand-in instruction in Wiseflow.

UNIVERSITY OF TROMSØ

Problem 1

In this problem you will derive and discuss some of the theoretical results that are found in deep learning.

A binary neural classifier is often trained by minimizing the cross-entropy (CE) between its output $\mathbf{z} \in \mathbb{R}^2$ and a label $\mathbf{y} \in \mathbb{R}^2$, where $\mathbf{z}$ is a stochastic vector and $\mathbf{y}$ is a one-hot vector. Here, we suppose that the dataset is made of a single point $\mathbf{z} = (z_0, z_1)$ associated to the label $\mathbf{y} = (1, 0)$.

(1a) After stating the definition of the cross-entropy between $\mathbf{z}$ and $\mathbf{y}$, show that the formula simplifies into the logarithm of a single term.

(1b) What can happen if $\mathbf{z}$ is not stochastic?

(1c) In this question only, we consider $N > 0$ samples. Give a pseudo-code to compute the cross-entropy for that dataset in $2N$ operations[1]. Bonus: Give a Python one-line of code.

(1d) In order to study the consequences of a regularization, compute:

$$\text{argmin}_{\mathbf{z}} \text{CE}(\mathbf{z}; \mathbf{y}) \qquad \text{(No regularization)}$$

$$\text{argmin}_{\mathbf{z}} \text{CE}(\mathbf{z}; \mathbf{y}) + ||\mathbf{z}||_2^2 \qquad (\ell_2 \text{ regularization})$$

$$\text{argmin}_{\mathbf{z}} \text{CE}(\mathbf{z}; \mathbf{y}) + ||\mathbf{z}||_1 \qquad (\ell_1 \text{ regularization})$$

(1e) Discuss the results of the previous question.

[1] Accessing a coordinate/indexing is not an operation whereas a test is.

Problem 2

In this problem you will use a pretrained image classifier trained on the ImageNet dataset (1) and explore several approaches to explain the models predictions and quantify its uncertainty. The data can be found on Canvas in `Canvas/Files/homeexam/problem2.zip`.

(2a) Explain one approach that can be used to interpret/explain a specific prediction of a deep learning model and one that can be used to interpret/explain the model as a whole.

(2b) Implement a VGG16 (2) and load the weights for the ImageNet trained model (PyTorch:https://pytorch.org/vision/stable/_modules/torchvision/models/vgg.html, TensorFlow:https://www.tensorflow.org/api_docs/python/tf/keras/applications/vgg16/VGG16). Describe the VGG16 network and your implementation and report the accuracy for the validation data. Describe improvements that could be done to improve results further and argue why they will help.

(2c) Produce Class Model Visualisations as proposed in (3) (Section 2) for 3 random classes of your choice. To reduce the high frequency components in the generated maps due to gradient ascent, one effective method of regularization is to convolve the output class model maps with a Guassian kernel after every k steps (i.e. k=5). See Section 3 in (4) for more details. Visualize the obtained maps with and without applying the Gaussian kernel and discuss the results.

(2d) For each of the 5 images of the test set, report the top five predictions of the model and discuss the results.

(2e) For the 5 images of the test set, produce Class Saliency Maps as described in (3) (Section 3.1) for the predicted class. Visualize and discuss the results.

(2f) Repeat the process for the same images, now using Guided Backpropagation (5) to obtain the visualization maps. Visualize and discuss the results.

(2g) Explain Dropout (6) and how it can be used to model uncertainties via Monte Carlo Dropout.

(2h) Take the 5 images of the test set and compute their uncertainties. Discuss the results.

Problem 3

You are working for the company RocketAI and working on the task of Image Classification. Your co-worker approaches you one day with a dataset (`Canvas/Files/homeexam/problem3.zip (DataShuffled.npz)`) and admits to you that he has by accident shuffled the data such that each image is divided into four quarters that have a random permutation. Your co-worker managed to keep the permutation vectors (indicating, which quarter belongs into which position) for some of the images, while the rest and the original data was permanently deleted before he realized this mistake. He is asking for your advice and you mention that you can try to design a deep learning model that can take the shuffled images as input and predict the correct permutation vector, thereby hopefully fixing his mistake. A function for loading the data and de-shuffling it based on a permutation vector is provided on Canvas (`Canvas/Files/homeexam/problem3.zip`).

(3a) Design a Deep Learning model that can be used for this task and describe how you would solve this problem. As a suggestion, a VGG-11 with Batch Normalization after the convolutional layers should be a good start. Explain why.

(3b) Implement your proposed solution. Your boss will not be happy unless you are able to get 60% or more of the images correctly de-shuffled. Report the accuracy on an image-level (based on the number of images that are correctly de-shuffled) and the accuracy for assigning a quarter to the correct position in an image (quarter-level accuracy).

As you complete your implementation and achieve sufficient results, your co-worker approaches you in an agitated mood. His boss has asked him to deliver the classification results for another dataset `Canvas/Files/homeexam/problem3.zip (DataNormal.npz)` in the next 5 minutes, giving you not enough time to retrain the model on the new de-shuffled data. You mention to your colleague that all you can do is to train a nearest neighbor classifier on top of the already trained deep learning model in order to at least get some results relatively fast.

(3c) Take the dataset and send it through the feature extractor of your de-shuffling network (remove the permutation prediction head) to extract a feature representation. Report what accuracy you get for the nearest neighbor classifier on top of these features and compare it to: 1) a nearest neighbor classifier trained directly on the input space and 2) a nearest neighbor classifier trained on top of your proposed Deep Learning model with randomly initialized weights. Feel free to use `sklearn.neighbors.KNeighborsClassifier` to implement the nearest neighbor classifier with $k = 3$.

(3d) Discuss the results and argue if the de-shuffling approach can be a useful tool also in other settings.

References

[1] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255, 2009. doi: 10.1109/CVPR.2009.5206848.

[2] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In Yoshua Bengio and Yann LeCun, editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015. URL http://arxiv.org/abs/1409.1556.

[3] Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Deep inside convolutional networks: Visualising image classification models and saliency maps. *arXiv preprint arXiv:1312.6034*, 2013.

[4] Jason Yosinski, Jeff Clune, Anh Mai Nguyen, Thomas J. Fuchs, and Hod Lipson. Understanding neural networks through deep visualization. *CoRR*, abs/1506.06579, 2015. URL http://arxiv.org/abs/1506.06579.

[5] Jost Tobias Springenberg, Alexey Dosovitskiy, Thomas Brox, and Martin Riedmiller. Striving for simplicity: The all convolutional net. *arXiv preprint arXiv:1412.6806*, 2014.

[6] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 15(1):1929–1958, 2014.