Graded assignment 2 : Do advanced keras based NN models and autodiff based NN model

Due May 18 by 11:59pm **Points** 20 **Submitting** a website url
Available Mar 19 at 12am - May 18 at 11:59pm 2 months

This assignment was locked May 18 at 11:59pm.

I) Do write autodiff python library (we discussed in class) and based on it , do mnist classifier in colab (similar to last exercise but using autodiff library that you will write instead of direct numpy) - DO NOT COPY PASTE FROM BOOK. WRITE CLEAN CUSTOM CODE WITH YOUR OWN CLASS NAMES AND VARIABLE NAMES FROM SCRATCH.

II) do keras models for four problems (4 different data sets)

a) boolean cross entropy

b) sparse categorical crossentropy (softmax) - try to get images of 7 classes with images.google.com google search and classify

c) logistic regression

d) multi head classification (multi class classification)

Do not copy from medium or existing colabs. do it from scratch (we will run turnitin).

Write proper colabs (points based on how cleanly and well documented your colabs are. checkout the tensorflow.org tutorial colabs as a template (i shared in announcement). Proper integration with tensorboard and metrics visualization. Proper visualize the input samples, results, losses, confusion matrices etc.. (just like fastai kind of colabs).

Pick problems from either kaggle or from paperwithcode.com for this exercise.

Try to use as many advanced features of keras as possible

Ensure you do dropout, early stopping and l2 regularization techniques
Ensure you use proper network initialization and optimizer (see keras api) and proper activation functions in each layer.

Ensure you use proper number of neurons and proper layers

Use mini batching and write your custom callbacks for printing custom metrics

Explore and use both functional api and sequential api.

Use https://github.com/keras-team/keras-tuner (https://github.com/keras-team/keras-tuner) for hyperparam tuning

Submit each of the model colab seperately (also onrender example web app link) in single directory (directory name is assignment 2).

Also checkin github the modelcards showcasing your work for each model - only evaluation section is mandatory - fairness stuff is not.

https://modelcards.withgoogle.com/about (https://modelcards.withgoogle.com/about)

https://research.google/pubs/pub48120/ (https://research.google/pubs/pub48120/)

https://arxiv.org/pdf/1810.03993.pdf (https://arxiv.org/pdf/1810.03993.pdf)

https://medium.com/the-false-positive/increasing-transparency-in-machine-learning-models-311ee08ca58a (https://medium.com/the-false-positive/increasing-transparency-in-machine-learning-models-311ee08ca58a)

Use nbdev for notebook development in structured way. seperate model/code files and tests using tools of nbdev.

https://www.youtube.com/watch?v=Hrs7iEYmRmg (https://www.youtube.com/watch?v=Hrs7iEYmRmg)

(https://www.youtube.com/watch?v=Hrs7iEYmRmg)

Ensure you give full proper access to your github - and submit top level url of the directory.

Follow keras coding conventions listed here : **https://keras.io/contributing/ (https://keras.io/contributing/)**

1. use PEP8 syntax conventions, but we aren't dogmatic when it comes to line length. Make sure your lines stay reasonably sized, though. To make your life easier, we recommend running a PEP8 linter:
   - Install PEP8 packages: `pip install pep8 pytest-pep8 autopep8`
   - Run a standalone PEP8 check: `py.test --pep8 -m pep8`
   - You can automatically fix some PEP8 error by running: `autopep8 -i --select <errors> <FILENAME>` for example: `autopep8 -i --select E128 tests/keras/backend/test_backends.py`
2. When committing, use appropriate, descriptive commit messages.
3. Update the documentation. If introducing new functionality, make sure you include code snippets demonstrating the usage of your new feature.
4. Submit your PR. If your changes have been approved in a previous discussion, and if you have complete (and passing) unit tests as well as proper docstrings/documentation, your PR is likely to be merged promptly.