To-Do's before Monday the 18th of May, 19:00

To be full prepared for Monday's online session with a mentor, you should do:

1. **[MUST]** Learn all important concepts and run all lines of code in Chapter 2, Production - Alternative notebook (USE OPTION 2 in Gathering data section!). *Pay special attention to the Questionnaire at the end of the Notebooks. Those questions are designed to help you with understanding of the material, and answering them confidently is proof that you have learned well. We will also bring them up during the Tuesday or Saturday sessions to make sure everyone is on the same page.*

   1.1 **[OPTIONAL]** If you struggle with the content, you can utilize the video materials from fast.ai course

2. **[MUST]** Write down an article with not less than 500 words and publish it on Medium:

   Topics for Medium articles. **Choose one by adding your name next to the title**. Only one person per topic, so do not take too much time to decide.

   This week, the topics are tagged as *General* or *Technical*. Technical topics are related to the fastai library and may require a little more reading of the documentation or experiments with the jupyter notebooks. We encourage you to try these as soon as you feel comfortable with the coding and even before that as it will help you become comfortable.

   1. **DataBlock and Dataloaders.** (Technical) [Add name]
      The notebook quickly introduces the concepts of `DataBlock, Dataloader` and `Dataloaders.` You do not need to understand all aspects of these, but just that these are Python Objects that group many elements related to the data for the model: training set, validation set, classes … It also gives access to methods (functions) that are useful to handle data.

      Experiment with the DataBlock and DataLoaders you have created (`bears` and `dls`) and discover what it allows you to see when you call different parts of the objects.
      `DataBlock`: the method summary gives a step by step account of what the computer is doing when you create a DataLoaders. Run the command below and see what this means to you.

      ```
      bears.summary(path)
      ```

      `DataLoaders`: the code below shows you some of things you can retrieve from the data:

      ```
      dls.train.dataset, dls.valid.dataset
      dls.train.items[0:5], dls.valid.items[0:5],
      dls.n, dls.train.n, dls.valid.n
      dls.vocab
      dls.train.show_batch(max_n=12, ncols=4)
      ```

```
dls.valid.show_batch(max_n=9, ncols=3)
```

Experiment more with the DataBlock (bears) and the DataLoaders (dls) and think of what these may mean. Share your experiments with your fellow students.

To go deeper (optional) see the following article on Pytorch dataloaders.

2. **Preprocessing**. (General) [Add name]
   Before we can use the images with our model, we need to preprocess them, in this case, we resize them. Why is preprocessing required or advised, and what possible preprocessing beyond resizing (and their purpose). Do not handle augmentation which is for another topic.
   The following articles may be a good start: [1], [2], [3] (focus on data preparation in this article, augmentation is for another topic)

3. **Data augmentation ideas**. (General) [Add name]
   The notebook explains what data augmentation is and some options available in fastai. Explain the underlying idea/intuition for data augmentation and the methods used by practitioners and researchers. Do not handle data preparation.preprocessing which is for another topic.
   These articles may help to start: [1], [2] (focus on augmentation in this article, data preparation is for another topic), [3] (these are research topics but explain the underlying idea of augmentation)

4. **Data Augmentation for images (fastai).** (Technical) [Add name]
   Explain in your own words the concept of transforms (tfsm) and how it is used for data augmentation. What is the difference between `item_tfms` and `batch_tfms`. The documentation and experimentation in the notebook are the best ways to learn here. Experiment with different transforms and see how they show when you apply `dls.valid.show_batch(...)` to them. Share this with your fellow students.

5. **Metrics for classification and Confusion Matrix**. (General) [Add name]
   In the original fastbook notebook. Jeremy creates a learner with the following command:
   ```
   learn = cnn_learner(dls, resnet18, metrics=error_rate)
   ```
   In effect, he uses a single metric, namely the classification error rate. There are other metrics available for classification problems. In particu``lar Accuracy, Precision, Recall. These are also summarised in the Confusion Matrix. Summarize what the confusion matrix is, how these metrics mean, how they are calculated and when one is preferred to others.
   The following article is a good start.

6. **Metrics for classification in fastai**. (Technical) [Add name]
   Similar to the previous topic, but with a focus on experimenting how fastai works with these. In the notebook, Jeremy only uses a single metric:

epoch	train_loss	valid_loss	error_rate	time
0	1.235733	0.212541	0.087302	00:05

   It is also possible to show several metrics like below:

epoch	train_loss	valid_loss	error_rate	accuracy	precision_score	recall_score	time
0	1.376304	0.232178	0.115385	0.884615	0.918919	0.850000	00:16

   Experiment with several metrics using the fastai library. Have a look at the document on fast.ai and the underlying sklearn documentation to understand the difference between the metrics. Show to your fellow student what you have seen in your notebook. We will discuss how to interpret these.

   Documentation is here. Focus on the single-label classification metrics here, as this is the type of problem we have here. Play with several metrics and see what type of result they give.

   Hint 1:
   Pass a list to `metrics` when you create your learner

```
metric_list = [....]
cnn_learner(dls, resnet18, metrics=metric_list)
```

   Hint 2:
   When the documentation uses "CamelFormat" (e.g. `Recall` or `CohenKappa`), it is a metric class and not a function (e.g. `accuracy` or `error_rate`).
   For classes, you need to first "instantiate" the class by passing it a few parameters. In particular in our case, we need to pass the parameter `average`:
   `average='macro'` or `'weighted'` instead of the default value `'binary'`.

   In summary:

```
metric_list = [Recall(average='Macro'), …]
```

7. **Cleaning the data** (General) [Add name]
   In the notebook, we have an example of how data can be cleaned up or improved, namely, identifying the biggest errors, correcting the class or deleting irrelevant data. A key question is what are the criteria to do this.

   In the case of this notebook, share which images you corrected for each class and how (change class, delete) and provide the rationale.

Can you find some general approaches of criteria to do this in general for a classification problem? Can you think of how you would modify this for other CV problems?

8. **Inference in fastai.** (Technical) [Add name]
   In the notebook, inference is made by loading one new picture and using `learn.predict`. The function returns the predicted class (text and number) and a vector of "probabilities" for each of the three classes:

   ```
   ('grizzly', tensor(1), tensor([0.2541, 0.6065, 0.1394]))
   ```

   Experiment with other images and see what it gives. Try with images that were not in the training set, and also a few that were in the training set to see the result. Experiments with images that have nothing to do with bears to see how the probabilities evolve.

   In fastai, there is another way to infer, not for a single image, but for a full set of images: `get_preds`. Use this method to make a prediction for the full validation set in one step:
   ```
   learn.get_preds(dl=dls.valid)
   ```
   See what it returns and how do you interpret this. How could you use these?

   Share these with your fellow students, as well as your conclusion after this exercise.

   Fastai documentation on inference.

9. **Projects - DriveTrain Approach**. (General) [Add name]:
   Jeremy proposes the DriveTrain approach in his article here. Read the article and summarize the most important points to share with the others.

10. **Labeling Images**. (General) [Add name]
    In the notebook, we use a straightforward method to label data, the search key word, that is we trust Bing to label the data. Then we correct manually for the largest errors. There are many approaches to labeling images. Here is a laundry list: manual labeling, using annotation tools, outsourcing labeling to third parties.
    There are also innovative approaches, such as active learning or cooperative learning. Provide a high level review on these approaches. These links may help: annotation tools, third parties, active learning (video), active learning (article)

11. **Deploy your app** (Technical) [Add name]
    The notebook refers to a way to deploy test web apps using Voila. Unfortunately this is not possible with Colab. If you are familiar with web app, you may want to

go into this to prepare for the practical exercise and share with your fellow students. References:
https://voila.readthedocs.io/en/stable/deploy.html
https://forums.fast.ai/t/chapter-2-voila/77990

3. **[MUST]** PPT of key concepts from Medium article. **You'll have 5 minutes to present it followed by 5 min Q&A with your classmates and a mentor.**

How to publish under the unpackAI publication:

- If you do not have a Medium Account, please register and send us your Medium username as can be seen in the picture.
- Write your own article and add it to the unpackAI Medium publication (https://medium.com/unpackai). How do that is here : https://help.medium.com/hc/en-us/articles/213904978-Add-a-draft-or-post-to-publication

Remember, we are learning together as a community so make sure to voice out your educated questions. While most confusions can be solved the quickest by simply Googling, do not hold back any questions if you have spent a fair amount researching.

We look forward to our next Saturday Q&A session and main Tuesday session!