„Making neural nets uncool again.“

A short introduction to the fastai library

fast.ai & fastai?

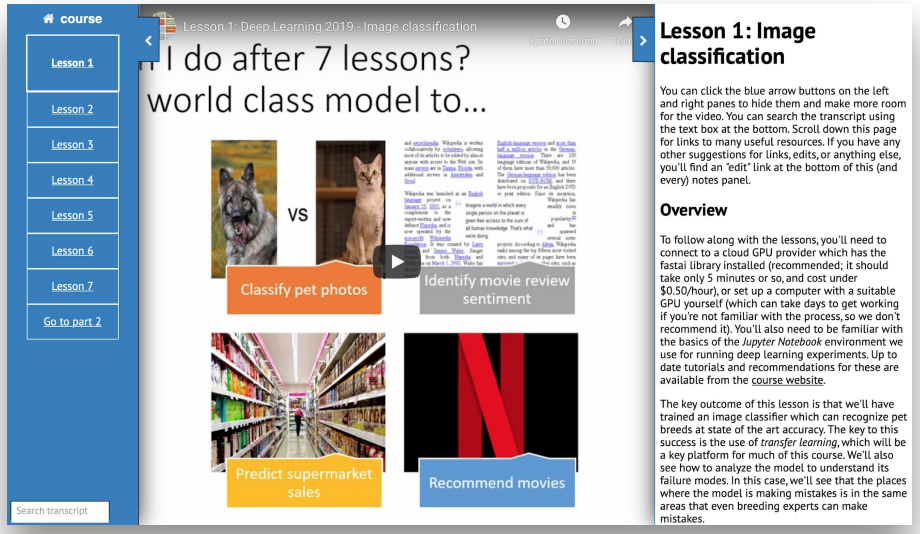
What is fast.ai & fastai?

- fast.ai *courses* [1]
  - Practical Deep Learning for Coders (v3 part 1) [2] *(v4 March 2020)*
  - Deep Learning from the Foundations (v3 part 2) [3]
  - A Code-First Introduction to Natural Language Processing [4]
  - Introduction to Machine Learning for Coders [5]
  - Computational Linear Algebra [6]

- fastai *deep learning library* [7][8]
  - “Keras” for PyTorch

fast.ai course format

- Free “Massive Open Online Course” (MOOC)
- Top down approach:
  1. Play with the code and get your hands dirty,
  2. then look into the details and the theory.
- Very active fastai community forum [9]
- Soon more advanced course material



The fastai deep learning library

fastai library – overview

- First release build on top of Keras based on Theano, then switch to PyTorch.

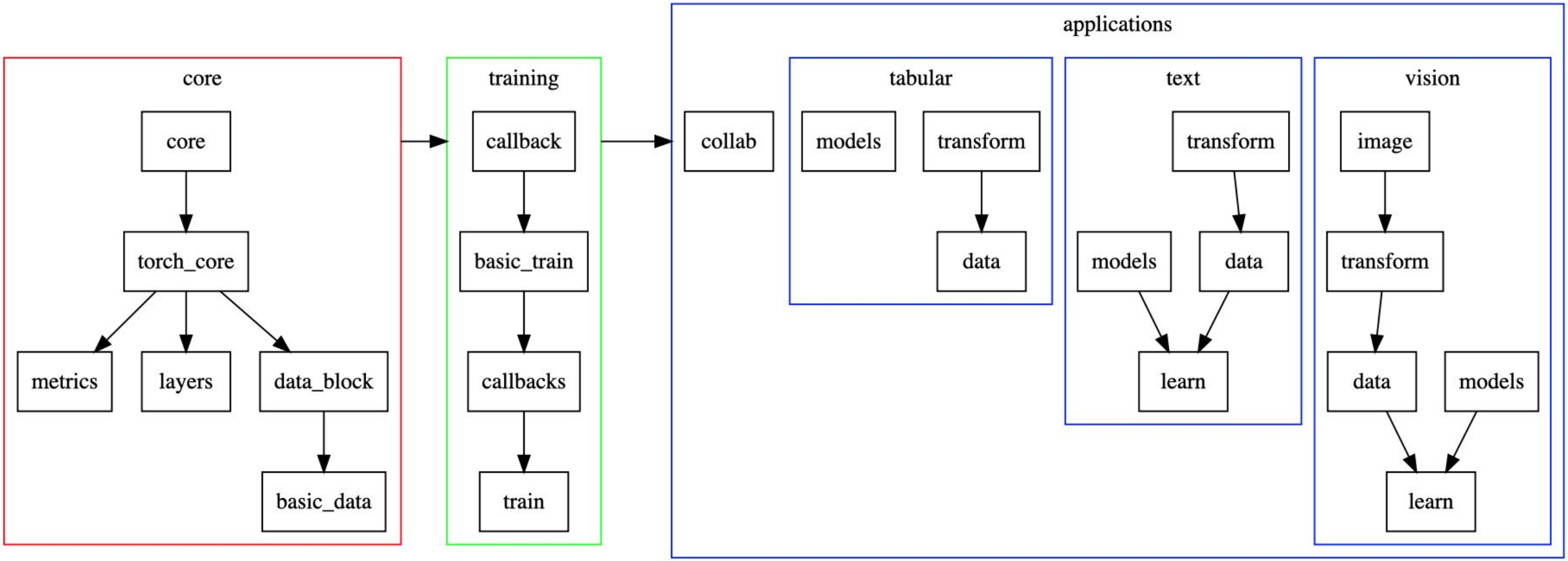
- fastai v1 currently supports Linux (Windows is experimental), and requires PyTorch v1 and Python 3.6 or later.

- Main applications
  - Vision (CV)
  - Text (NLP)
  - Tabular
  - Collaborative filtering

fastai library – main features

- Main features
  - Training loop
  - Data handling
  - Data transformation
  - Callback system
  - Pretrained models incl. tools

fastai library – setup



[8]

fastai library – highlights

- Highlights
  - A lot of best practices!
  - Computer vision data augmentation
  - NLP transfer learning with “Universal Language Model Fine-tuning for Text Classification” (ULMFiT) [10]
  - Tabular data preprocessing (categorify, fill missing data, date time handling, embeddings) [11]
- Very fast prototyping, even for non deep learning experts!

```
path = untar_data(URLs.MNIST_SAMPLE)
data = ImageDataBunch.from_folder(path)
learn = cnn_learner(data, models.resnet18, metrics=accuracy)
learn.fit(1)
```

The fastai workflow:
Data, augmentation, training & Co.

(examples mostly based on CV)

fastai library – data block API

- Where is my raw data?
- How to label the data set?
- How to split my data into a training and validation data set?
- How to add a test data set?
- How to add custom data transformations?

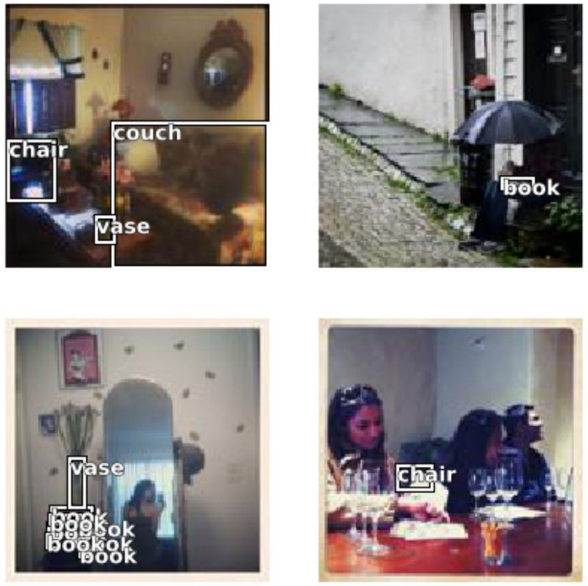
```
data = (ImageList.from_folder(path) #Where to find the data? -> in path and its subfolders
        .split_by_folder()              #How to split in train/valid? -> use the folders
        .label_from_folder()            #How to label? -> depending on the folder of the filenames
        .add_test_folder()              #Optionally add a test set (here default name is test)
        .transform(tfms, size=64)       #Data augmentation? -> use tfms with a size of 64
        .databunch())                   #Finally? -> use the defaults for conversion to ImageDataBunch
```

[8]

fastai library – data block API examples

```
data = (ObjectItemList.from_folder(coco)
        #Where are the images? -> in coco and its subfolders
        .split_by_rand_pct()
        #How to split in train/valid? -> randomly with the default 20% in valid
        .label_from_func(get_y_func)
        #How to find the labels? -> use get_y_func on the file name of the data
        .transform(get_transforms(), tfm_y=True)
        #Data augmentation? -> Standard transforms; also transform the label images
        .databunch(bs=16, collate_fn=bb_pad_collate))
        #Finally we convert to a DataBunch, use a batch size of 16,
        # and we use bb_pad_collate to collate the data into a mini-batch
```

```
data.show_batch(rows=2, ds_type=DatasetType.Valid, figsize=(6,6))
```




```
data_lm = (TextList
           .from_csv(imdb, 'texts.csv', cols='text')
           #Where are the text? Column 'text' of texts.csv
           .split_by_rand_pct()
           #How to split it? Randomly with the default 20% in valid
           .label_for_lm()
           #Label it for a language model
           .databunch())
           #Finally we convert to a DataBunch
```

```
data_lm.show_batch()
```

idx	text
0	! ! ! xxmaj finally this was directed by the guy who did xxmaj big xxmaj xxunk ? xxmaj must be a replay of xxmaj jonestown - hollywood style . xxmaj xxunk ! xxbos xxmaj this is a extremely well - made film . xxmaj the acting , script and camera - work are all first - rate . xxmaj the music is good , too , though it is
1	, co - billed with xxup the xxup xxunk xxup vampire . a xxmaj spanish - xxmaj italian co - production where a series of women in a village are being murdered around the same time a local count named xxmaj yanos xxmaj xxunk is seen on xxunk , riding off with his ' man - eating ' dog behind him . \n \n

```
adult = untar_data(URLs.ADULT_SAMPLE)
df = pd.read_csv(adult/'adult.csv')
dep_var = 'salary'
cat_names = ['workclass', 'education', 'marital-status', 'occupation', 'relationship', 'race', 'sex',
'native-country']
cont_names = ['education-num', 'hours-per-week', 'age', 'capital-loss', 'fnlwgt', 'capital-gain']
procs = [FillMissing, Categorify, Normalize]
```

```
data = (TabularList.from_df(df, path=adult, cat_names=cat_names, cont_names=cont_names, procs=procs)
                           .split_by_idx(valid_idx=range(800,1000))
                           .label_from_df(cols=dep_var)
                           .databunch())
```

[8]

fastai library – vision data augmentation

- A lot of different options
- Very fast/optimized

[8]

fastai library – vision data augm. cont.

- Can be easily setup and adapted [8]

get_transforms

```
get_transforms(do_flip:bool=True, flip_vert:bool=False,
max_rotate:float=10.0, max_zoom:float=1.1, max_lighting:float=0.2,
max_warp:float=0.2, p_affine:float=0.75, p_lighting:float=0.75,
xtra_tfms:Optional[Collection[Transform]]=None) → Collection[Transform]
```

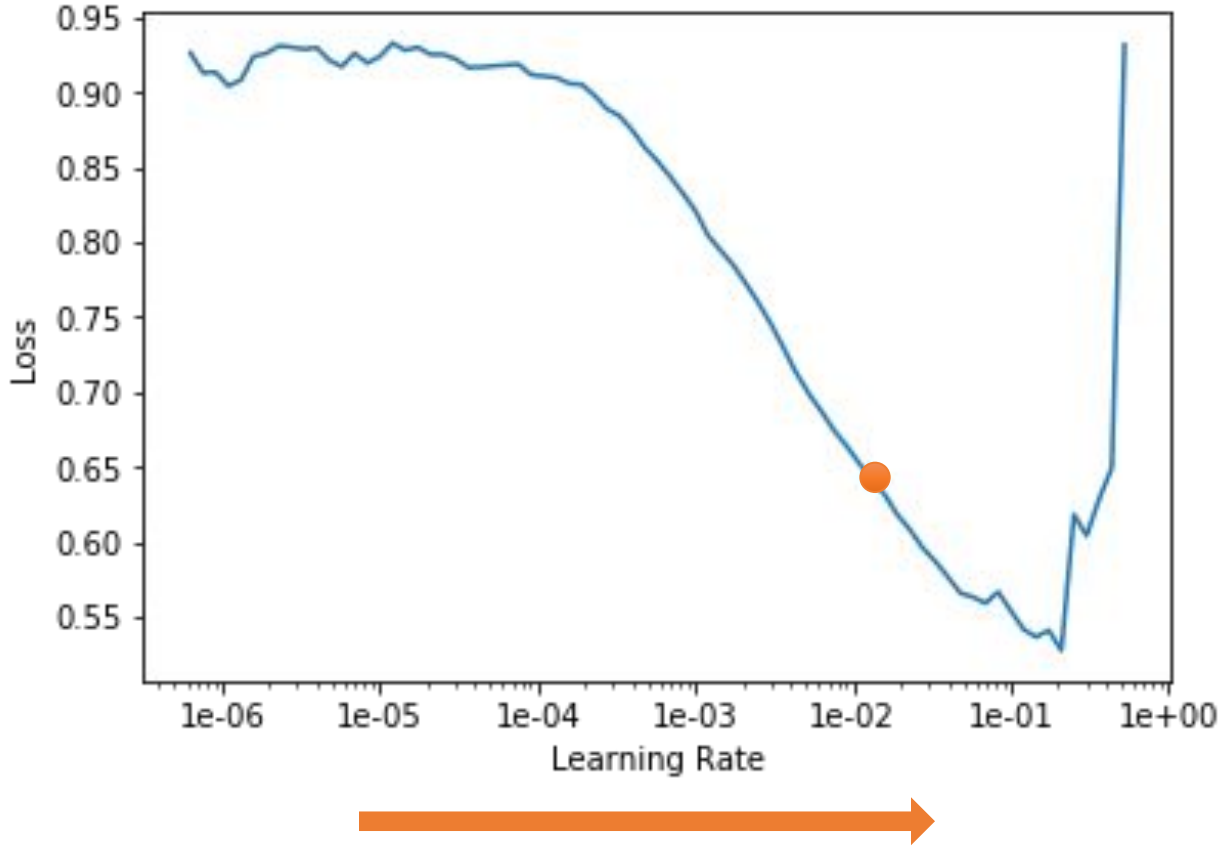
- Example code

```
def _rgb_randomize(x, channel:int=None, thresh:float=0.3):
    "Randomize one of the channels of the input image"
    if channel is None: channel = np.random.randint(0, x.shape[0] - 1)
    x[channel] = torch.rand(x.shape[1:]) * np.random.uniform(0, thresh)
    return x

rgb_randomize = TfmPixel(_rgb_randomize)
```

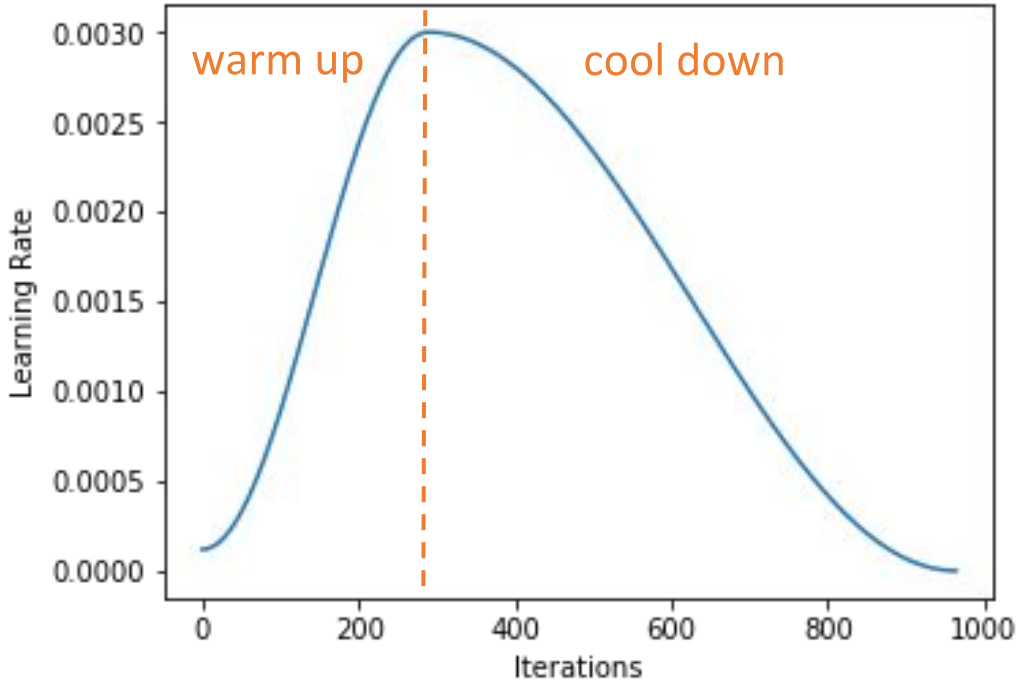
fastai library – training

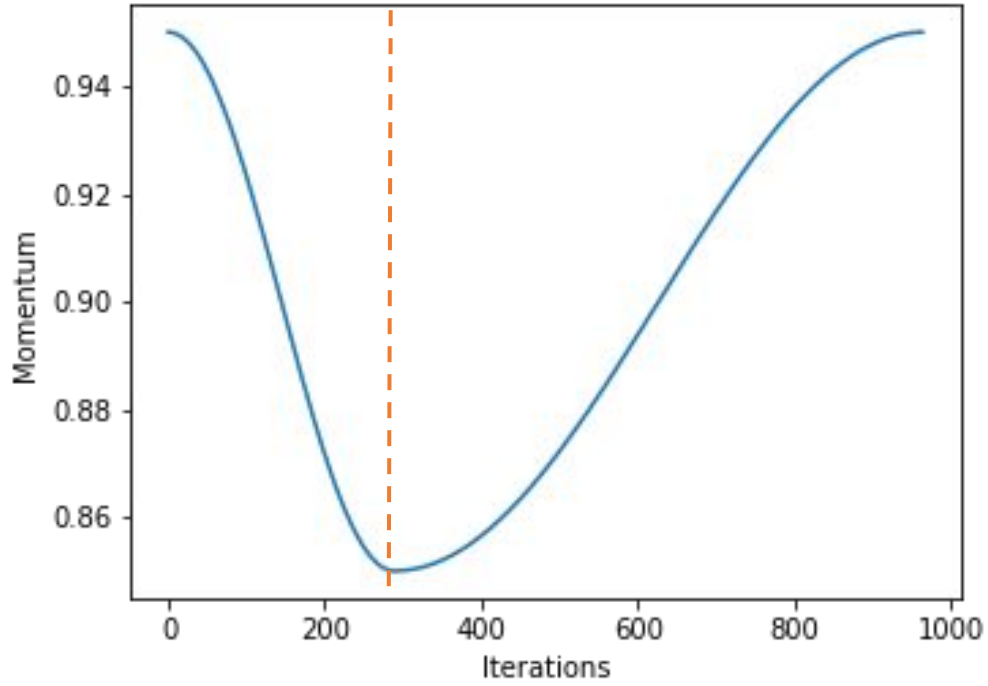
- Learning rate finder [8][12-14]



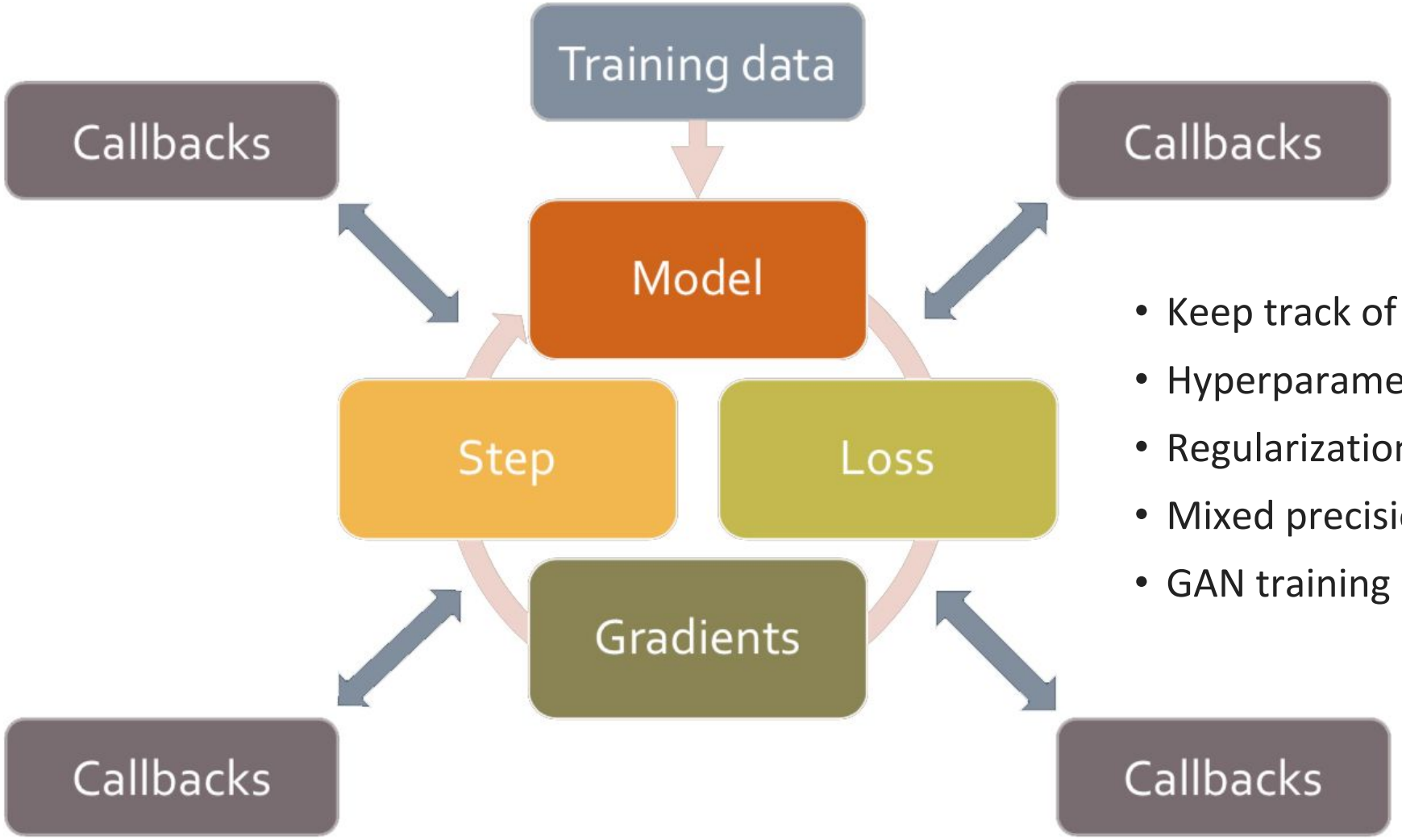
fastai library – training cont.

- Cyclical learning rates “1cycle policy” [8][12-14]





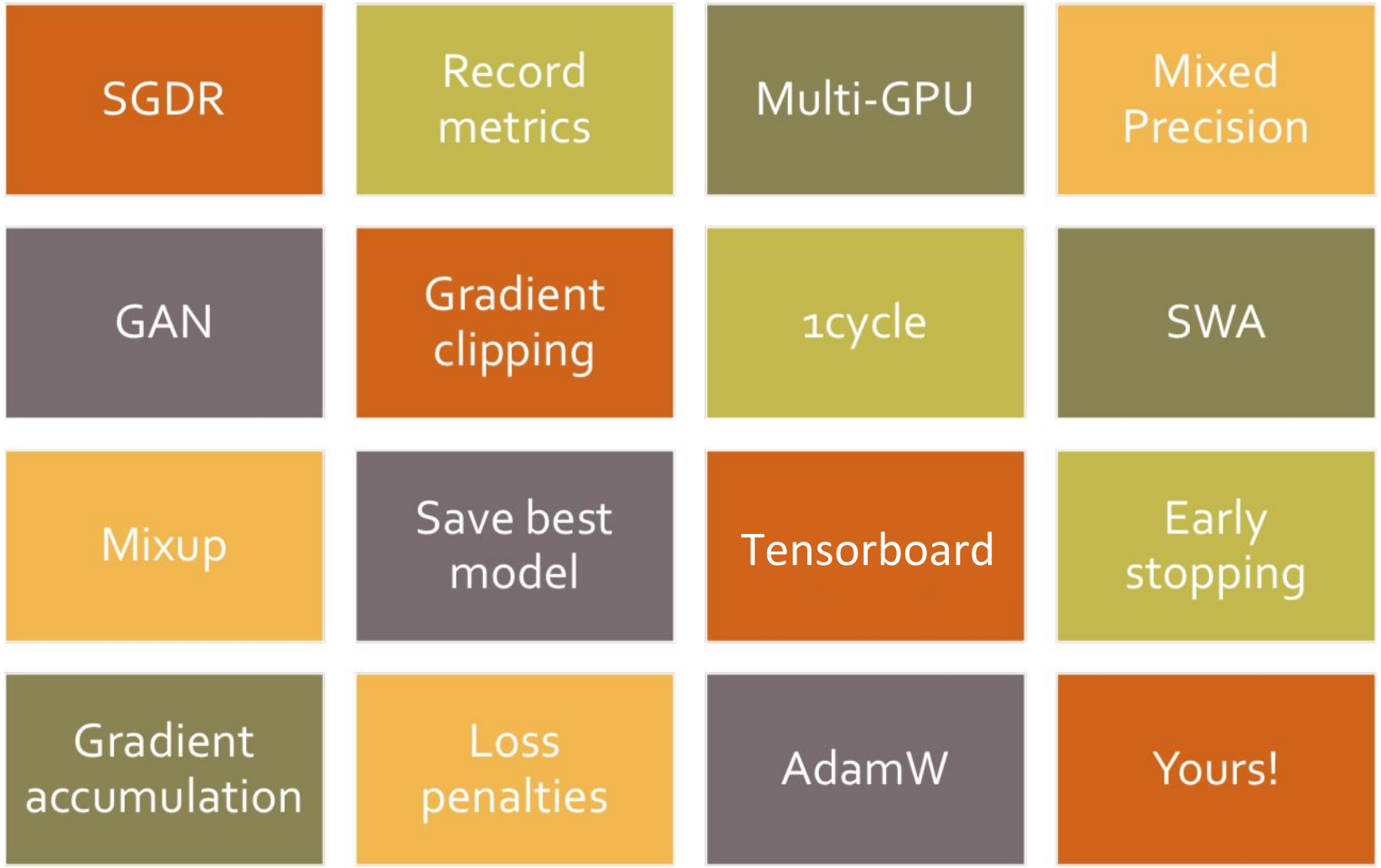
fastai library – callback system



- Keep track of losses & metrics
- Hyperparameters schedules
- Regularization techniques
- Mixed precision training
- GAN training

[15]

fastai library – customizable train. loop



[15]

fastai library – pretrained model zoo

- Vision
  - Torchvision (AlexNet, VGG, ResNet, SqueezeNet, DenseNet, Inception v3, GoogLeNet, ShuffleNet v2, MobileNet v2, ResNeXt, etc.)
  - XResNet (“Bag of tricks”) [16]
  - WideResNet
  - DarkNet (YOLO v3)
  - UNet

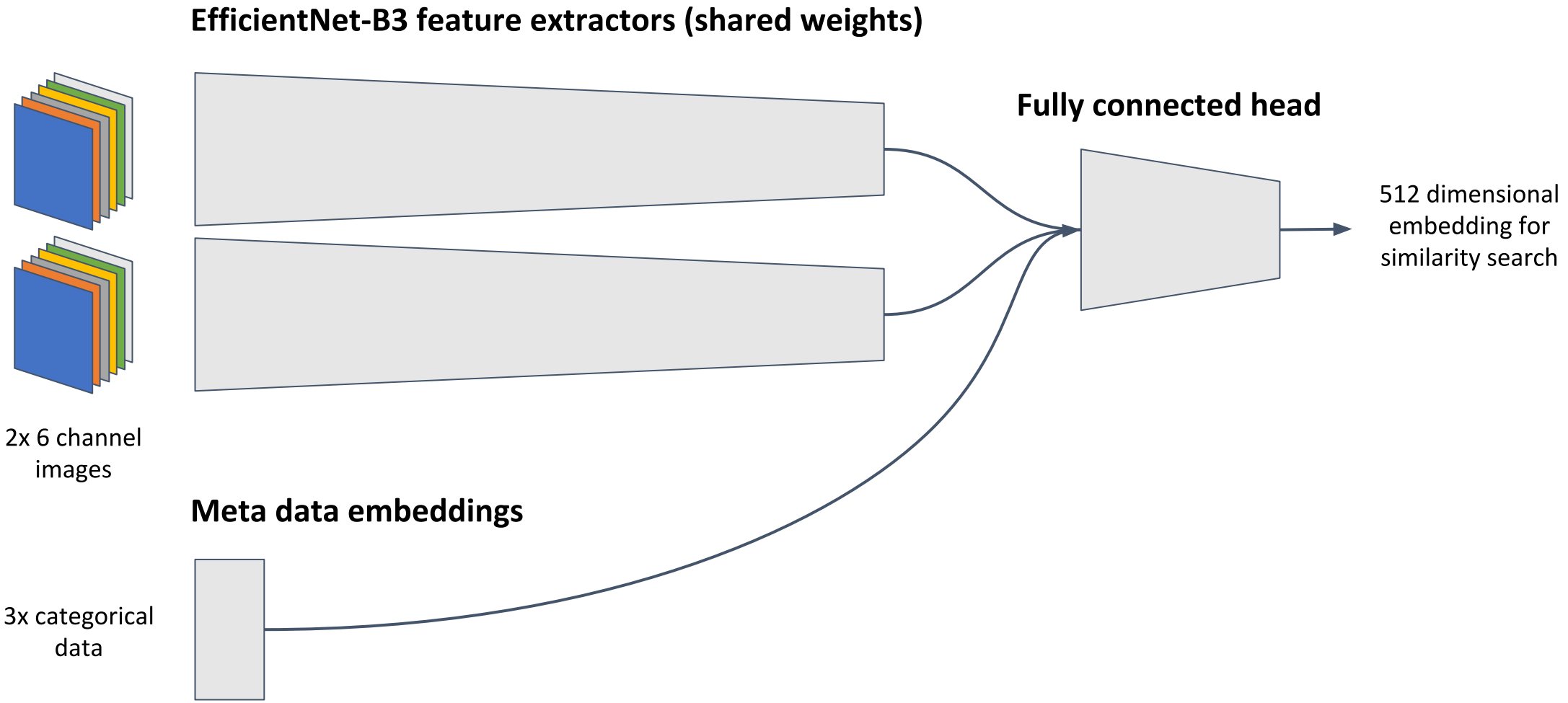
- Text
  - AWD-LSTM [18]
  - Transformer(-XL)
  - (Some fastai-ish BERT implementations.)

- Easy to add custom networks and heads for pretrained models and layer groups for discriminative learning rates.

fastai example – basic vision workflow

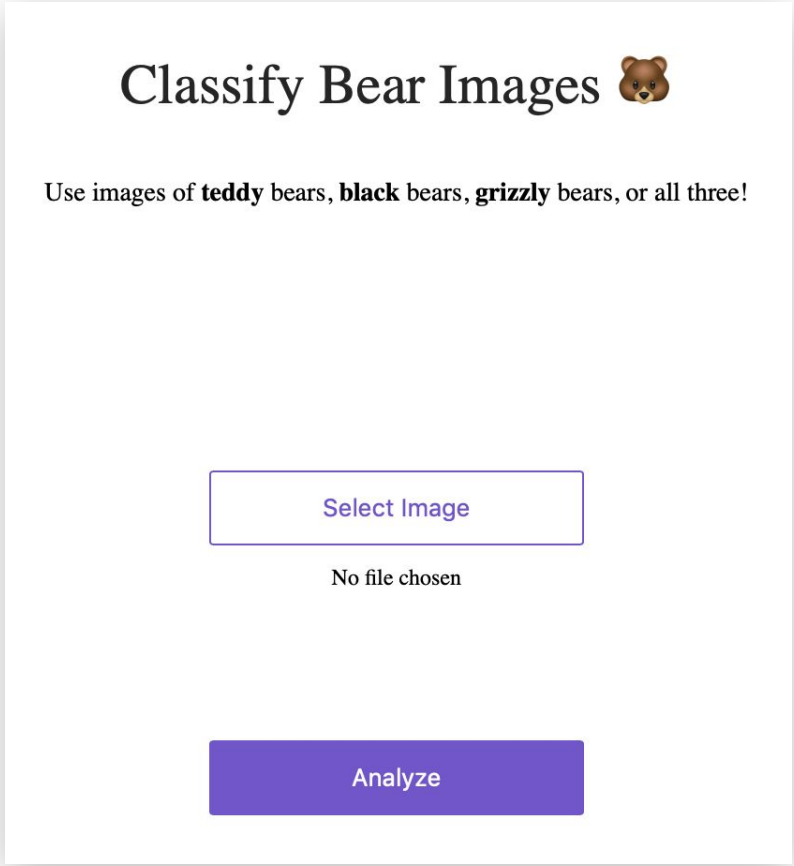
https://nbviewer.jupyter.org/github/fastai/course-v3/blob/master/nbs/dl1/lesson1-pets.ipynb

fastai example – a more complex model



fastai – deployment

- Available guides for deployment [18]
  - Render
  - Google App Engine
  - AWS Lambda
  - Amazon SageMaker
  - AWS BeanStalk
  - Azure Functions



https://fastai-v3.onrender.com

Pros, cons & the future

fastai – main pros & cons

Pros:

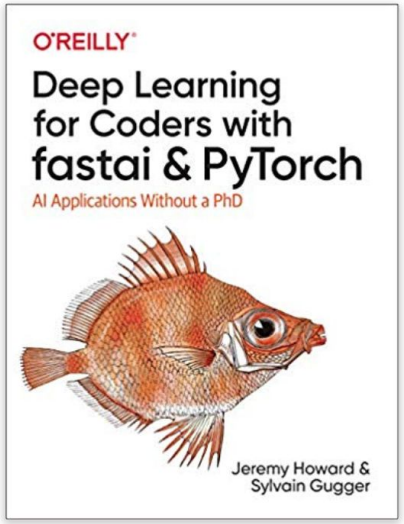
- A lot of integrated best practices!
- Fast results if you use the ready-to-use modules.
- Very hackable!
- Library parts can be (re-)used in custom pipelines.

Cons:

- Mid-level API is complex
- When your setup gets complex you need more knowledge of the inner workings of the library.

Future of fastai

- fastai v2 for PyTorch currently in development [19]
  - Will be extremely hackable
  - Data pipelines (GPU data augmentation, Rapids.ai for tabular)
  - Optimized callback system



[20]

- swiftai based on Swift for TensorFlow (S4TF) [21]

- Other PyTorch-based deep learning libraries:
  - PyTorch ignite [22]
  - PyTorch Lightning [23]

Sources

(1) fast.ai general site, https://www.fast.ai

(2) DL course v3 part 1, https://course.fast.ai

(3) DL course v3 part 2, https://course.fast.ai/part2

(4) NLP course, https://www.fast.ai/2019/07/08/fastai-nlp

(5) ML course, https://www.fast.ai/2018/09/26/ml-launch/

(6) Computational linear algebra, https://www.fast.ai/2017/07/17/num-lin-alg/

(7) fastai repo, https://github.com/fastai/fastai

(8) fastai documentation, https://docs.fast.ai

(9) fast.ai forums, https://forums.fast.ai

(10) ULMFiT, http://nlp.fast.ai/classification/2018/05/15/introducting-ulmfit.html

(11) fastai tabular, https://docs.fast.ai/tabular.html

(12) 1cycle policy, https://sgugger.github.io/the-1cycle-policy.html#the-1cycle-policy

(13) Super convergence, https://arxiv.org/abs/1708.07120

(14) Cyclical Learning Rates, https://arxiv.org/pdf/1506.01186

(15) “An infinitely customizable training loop” by Sylvain Gugger, https://www.youtube.com/watch?v=roc-dOSeehM

(16) Bag of tricks for CNNs, https://arxiv.org/abs/1812.01187

(17) fastai deployment, https://course.fast.ai/deployment_render.htm

(18) AWD-LSTM, https://arxiv.org/abs/1708.02182

(19) fastai v2, https://forums.fast.ai/t/fastai-v2-roadmap/46661 & http://dev.fast.ai/

(20) Deep Learning for Coders with Fastai and Pytorch, https://www.amazon.de/Deep-Learning-Fastai-Pytorch-Applications/dp/1492045527

(21) swiftai, https://github.com/fastai/swiftai

(22) PyTorch ignite, https://pytorch.org/ignite/

(23) PyTorch Lightning, https://github.com/williamFalcon/pytorch-lightning

Compiled by Michael M. Pieler (https://MichaelPieler.com)

Thank you for your attention!

Questions?

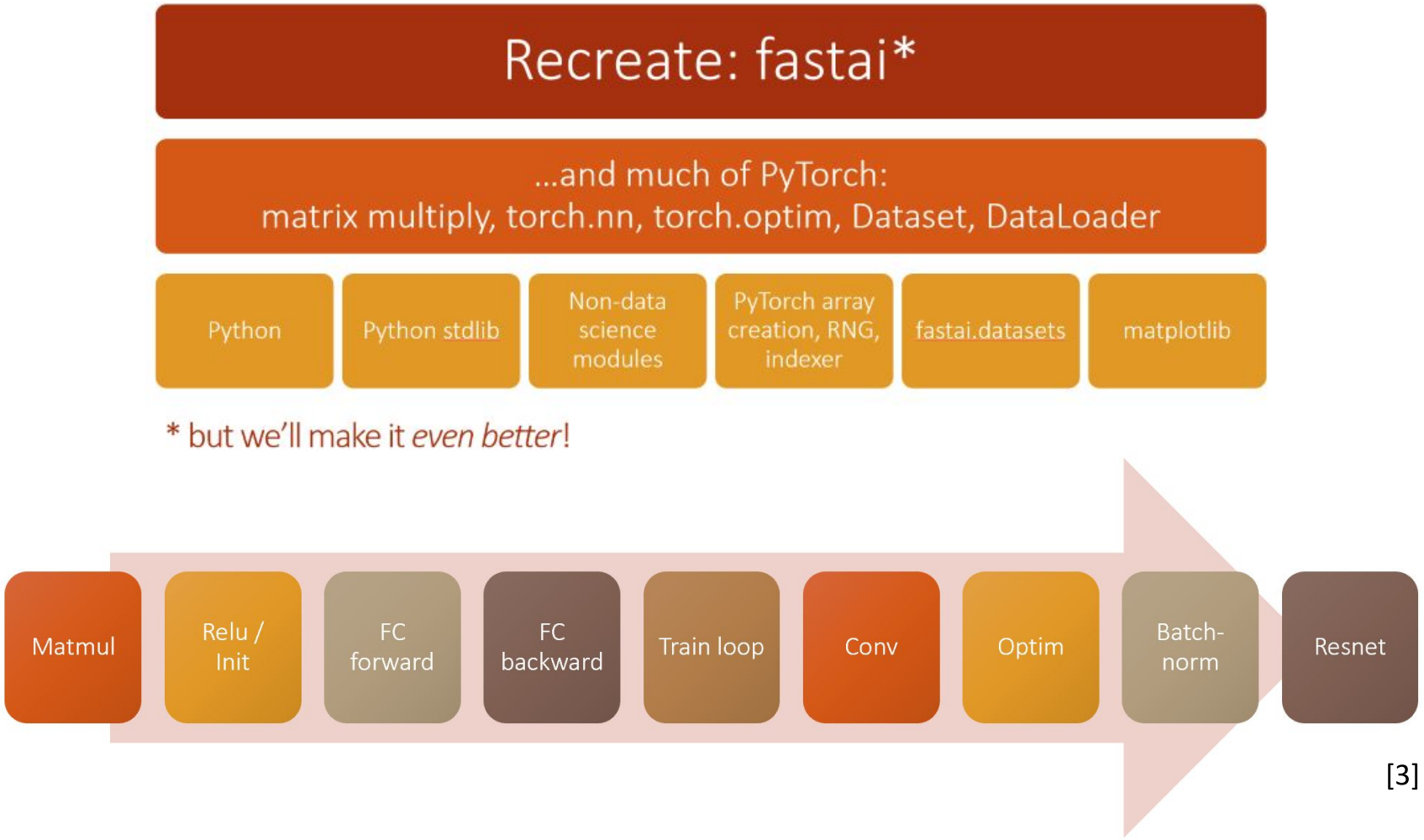
One more thing...

PyTorch fastai learning group in Vienna

- fast.ai v3 part 1 learning group meetups finished before summer.
  https://github.com/MicPie/fastai-pytorch-course-vienna

- **fast.ai v3 part 2 and fastai v2 learning group starts in October!**
  - How to setup a deep learning library from matrix multiplication to SOTA!
  - For people with coding and deep learning experience (fast.ai v3 part 1).
  - More information and sign up at the GitHub repo:
    https://github.com/MicPie/fastai-pytorch-course-vienna-v2

fast.ai v3 part 2 learning group outline



[3]