Pattern Recognition Lab

FRIEDRICH-ALEXANDER UNIVERSITÄT ERLANGEN-NÜRNBERG
FACULTY OF ENGINEERING

FastAi

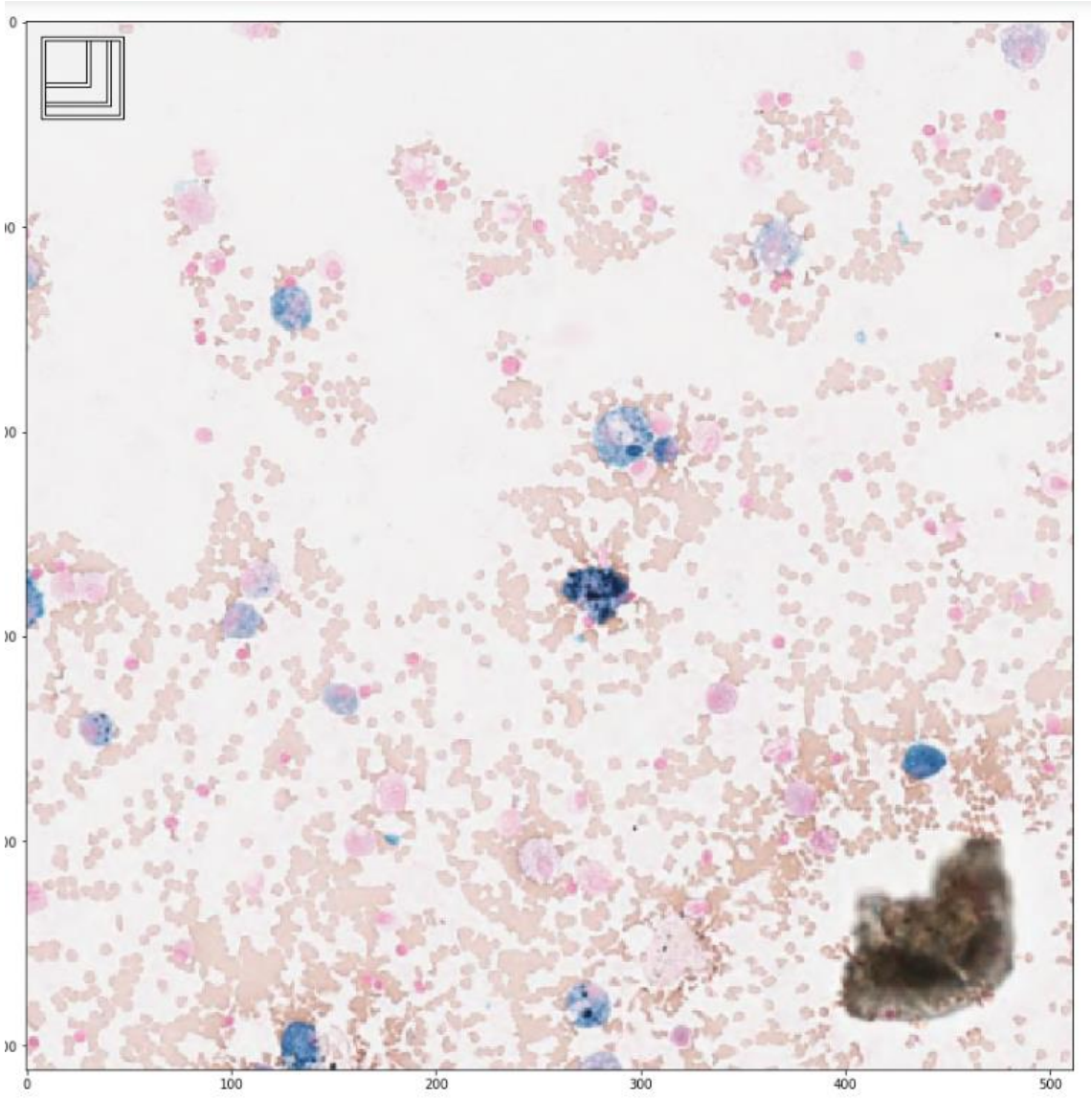
Computer Vision colloqium

Christian Marzahl

EUROIMMUN
a PerkinElmer company

Agenda

- Introduction
  - Motivation
  - Material
- Classification
  - Custom Head
  - Custom Augmentation
- Segmentation
- Key Concepts
  - Callbacks
    - Learning Rate Finder
    - Accumulate Scheduler
- Object Detection



1. Introduction:

My background:

- **Professional DL since Nov. 2015**
- **2016 Keras, 2017 FastAi**
- **https://github.com/ChristianMarzahl**

FastAi Features:

- **Easy to set up your experiment**
- **State of the Art methods (MixUp, FP16, Accumulate Scheduler)**
- **Support for deep customisations**

FastAi Documentation:

- **Videos: https://course.fast.ai/videos/?lesson=1**
- **Docs: https://docs.fast.ai/**

2. Key Features:

One Pipeline to rule them all

	Image Classification | Segmentation | ObjectDetection			Text	Tabluar
Data	Data = ImageList .from_folder(mnist) .split_by_folder() .label_from_folder() .transform(tfms) .databunch(bs=16) .normalize()	Data = SegmentationItemList .from_folder(path) .split_by_rand_pct() .label_from_func(y_fn) .transform(tfms, tfm_y=True) .databunch(bs=16) .normalize(imagenet_stats)	Data = ObjectItemList .from_folder(coco) .split_by_rand_pct() .label_from_func(y_fn) .transform(tfms, tfm_y=True) .databunch(bs=16, collate_fn=bb_pad_collate) .normalize()	Data = TextList .from_csv(imdb, 'texts.csv', cols='text') .split_by_rand_pct() .label_from_df(cols='label') .databunch(bs=42)	Data = TabularList .from_df(df, cont_names =[,age', 'hours-per-week']) .split_by_idx(valid_idx) .label_from_df(cols=,Salary‘) .databunch()
Vis	data.show_batch()	data.show_batch()	data.show_batch()	data.show_batch()	data.show_batch()
Training	learn = cnn_learner(data, models.resnet18) learn.fit_one_cycle(2, 1e-2)	learn = unet_learner(data, models.resnet34) learn.fit_one_cycle(2, 1e-2)	learn = Learner(data, models.resnet34) learn.fit_one_cycle(2, 1e-2)	learn = text_classifier_learner(data, AWD_LSTM) learn.fit_one_cycle(2, 1e-2)	learn = tabular_learner(data, layers=[200,100], metrics=accuracy) learn.fit_one_cycle(2, 1e-2)
Results	learn.show_results()	learn.show_results()	learn.show_results()	learn.show_results()	learn.show_results()

3. Classification

Custom Head or lets start a pool party with ten classes:

1. **Flatten:**

   Flatten(7x7x512) -> 25088 -> 10

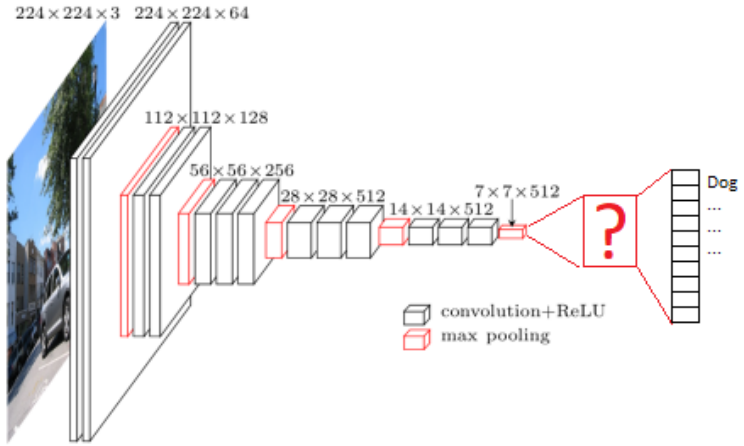
2. **Global Pool**

   GlobalAvg(7x7x512) -> 512 -> 10

   GlobalAvgMax(7x7x512) -> 1024 -> 10

3. **Custom Pool**

   https://git5.cs.fau.de/christlein/dgmp



4. Classification

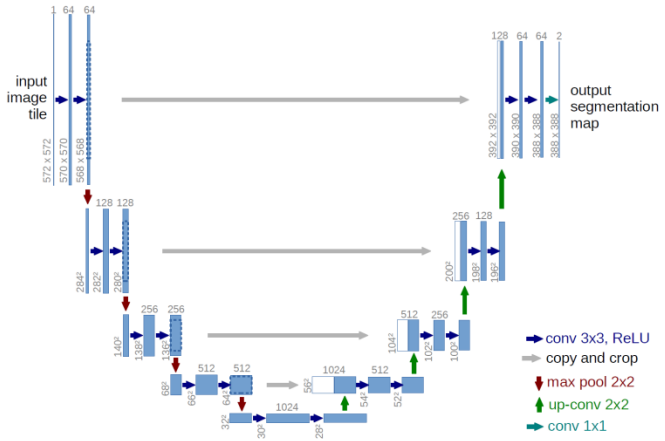
Custom Augmentation

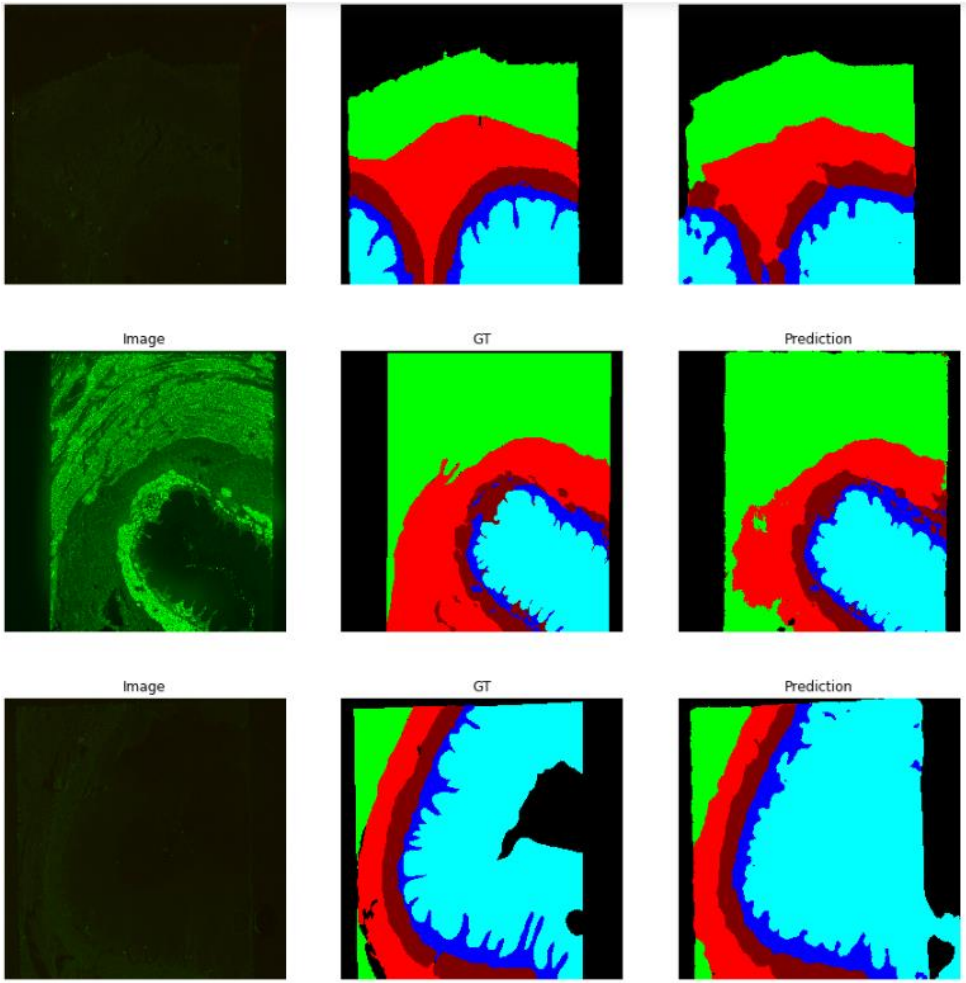



5. Segmentation & Demo

Key Feature:

- **unet_learner**
  - Data
  - Model
  - Metric (opt)
  - Loss (opt)



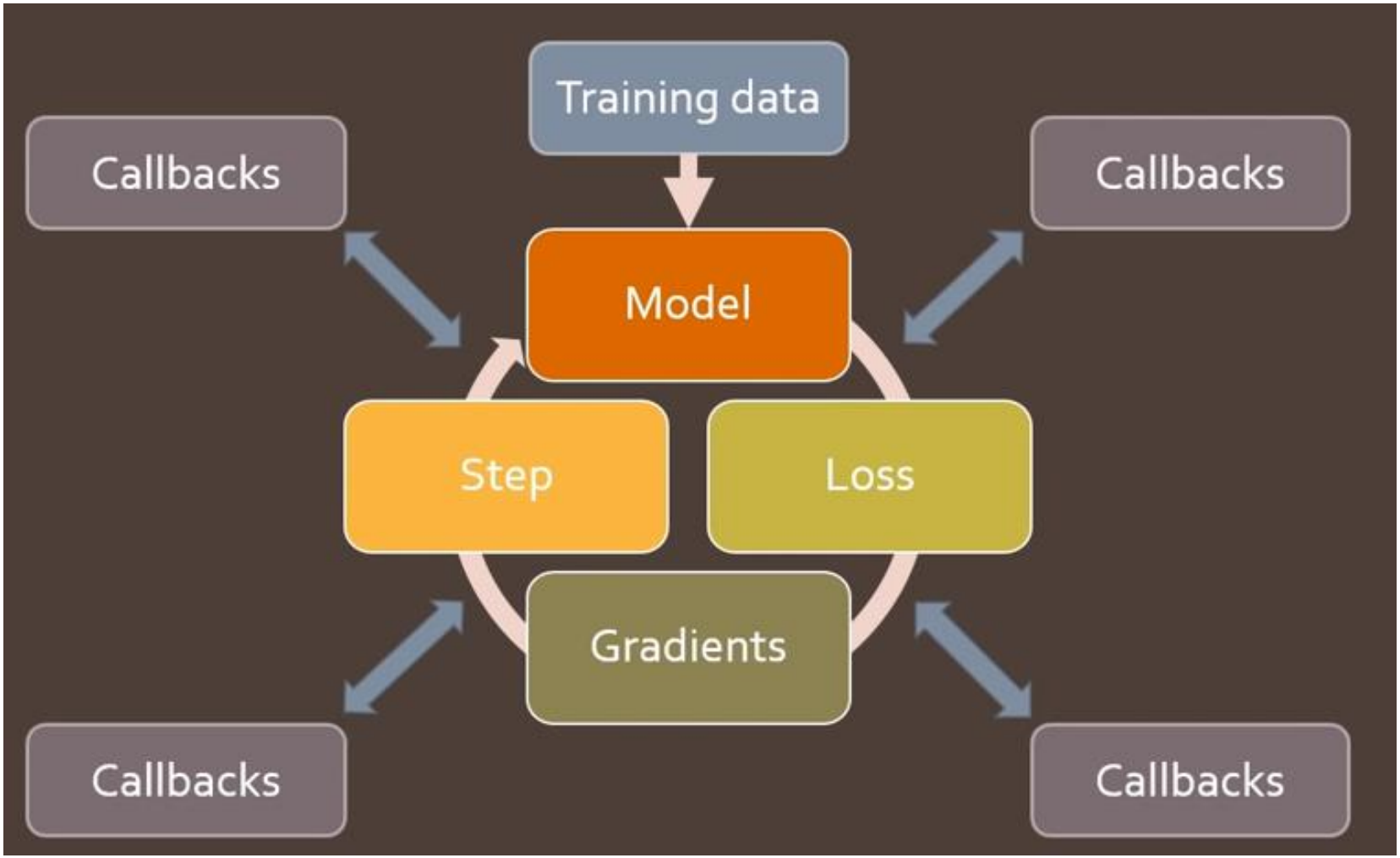


6. Callback System

```
def fit(epochs, learn):
    for epoch in range(epochs):
        learn.model.train()
        for xb,yb in learn.data.train_dl:
            loss = learn.loss_func(learn.model(xb), yb)
            loss.backward()
            learn.opt.step()
            learn.opt.zero_grad()

        learn.model.eval()
        with torch.no_grad():
            tot_loss,tot_acc = 0.,0.
            for xb,yb in learn.data.valid_dl:
                pred = learn.model(xb)
                tot_loss += learn.loss_func(pred, yb)
                tot_acc  += accuracy (pred,yb)
        nv = len(learn.data.valid_dl)
        print(epoch, tot_loss/nv, tot_acc/nv)
    return tot_loss/nv, tot_acc/nv
```

7. Callback System



8. Callback System

class Callback

Methods your subclass can implement

- on_train_begin
- on_epoch_begin
- on_batch_begin
- on_loss_begin
- on_backward_begin
- on_backward_end
- on_step_end
- on_batch_end
- on_epoch_end
- on_train_end
- get_state

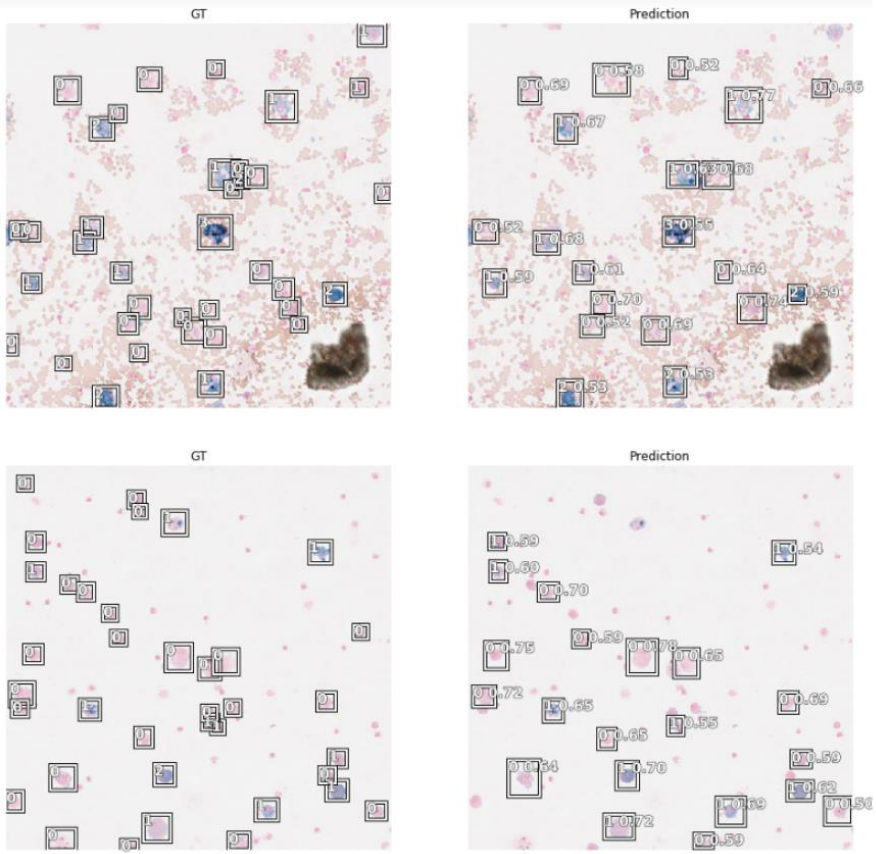
Callback

- LRFinder
- OneCycleScheduler
- MixUpCallback
- CSVLogger
- GeneralScheduler
- MixedPrecision
- HookCallback
- RNNTrainer
- TerminateOnNaNCallback
- EarlyStoppingCallback
- SaveModelCallback
- ReduceLROnPlateauCallback
- PeakMemMetric
- StopAfterNBatches

train and basic_train

- Recorder
- ShowGraph
- BnFreeze
- GradientClipping

9. Object Detection



Quellen

FastAi: https://www.fast.ai/