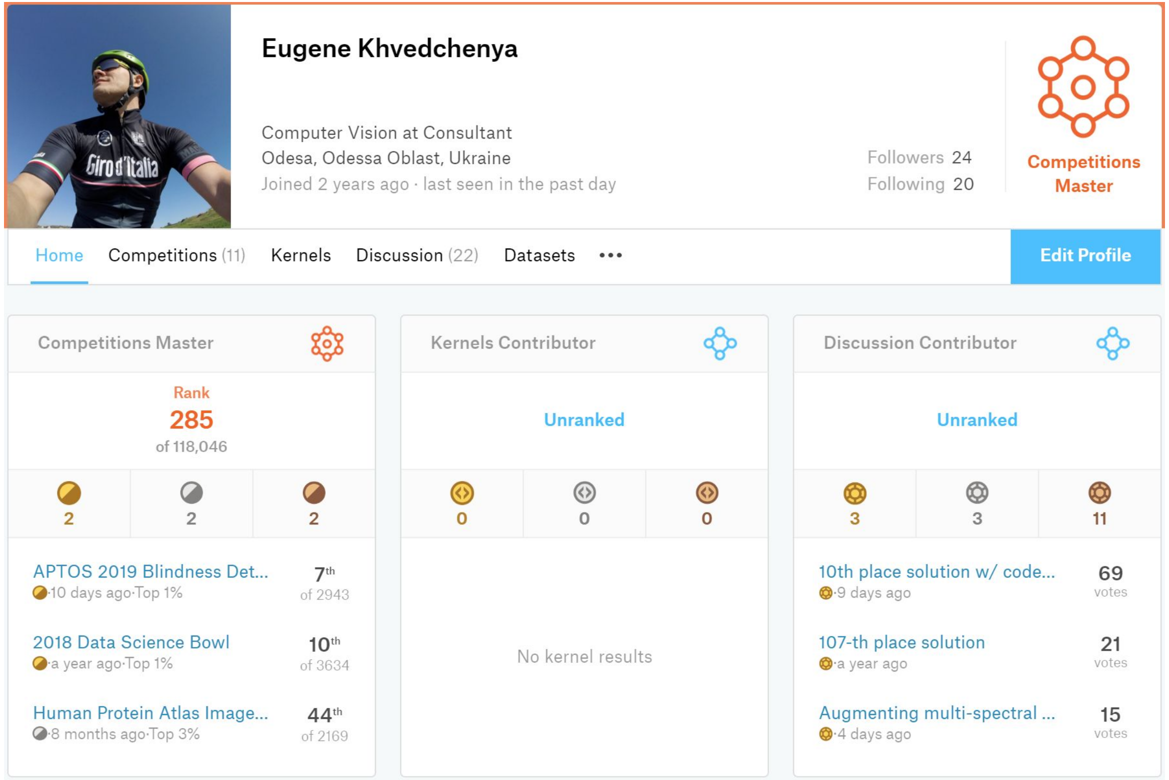
Bag of freebies

from competitive machine learning

Eugene Khvedchenya
ekhvedchenya@gmail.com
linkedin.com/in/cvtalks http://github.com/BloodAxe



Eugene Khvedchenya
ekhvedchenya@gmail.com
linkedin.com/in/cvtalks http://github.com/BloodAxe

What is “competitive machine learning”?

- In: Task, Data, Target Metric
- Out: Build a best model

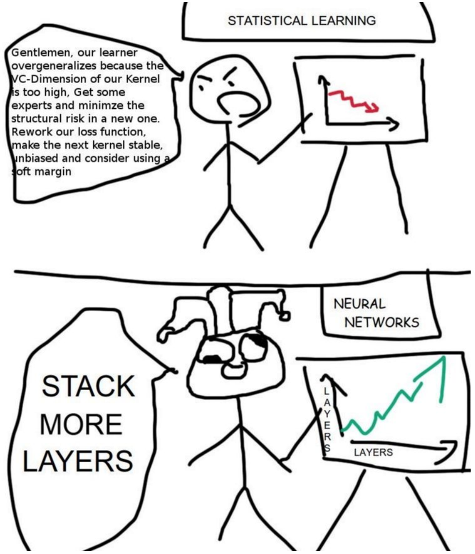
What is “competitive machine learning”?

- In: Task, Data, Target Metric
- Out: Build a best model

kaggle

[topcoder]™

DRIVENDATA SIGNATE



Menu

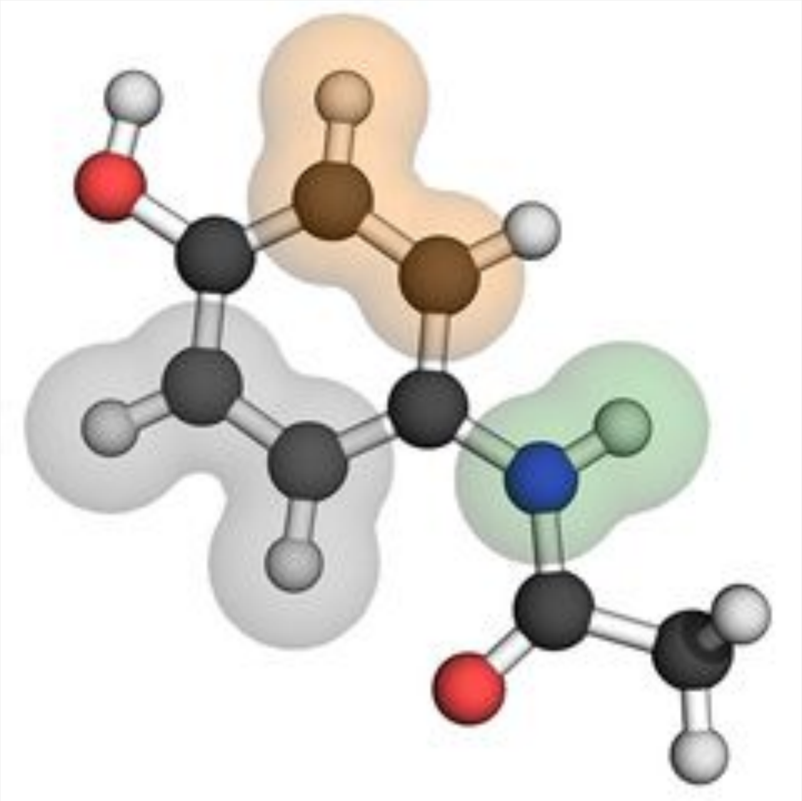
- Introduction
- Loss functions tricks
- Dataset tricks
- Training tricks

Recent Challenges

Fine-grained segmentation task for fashion and apparel

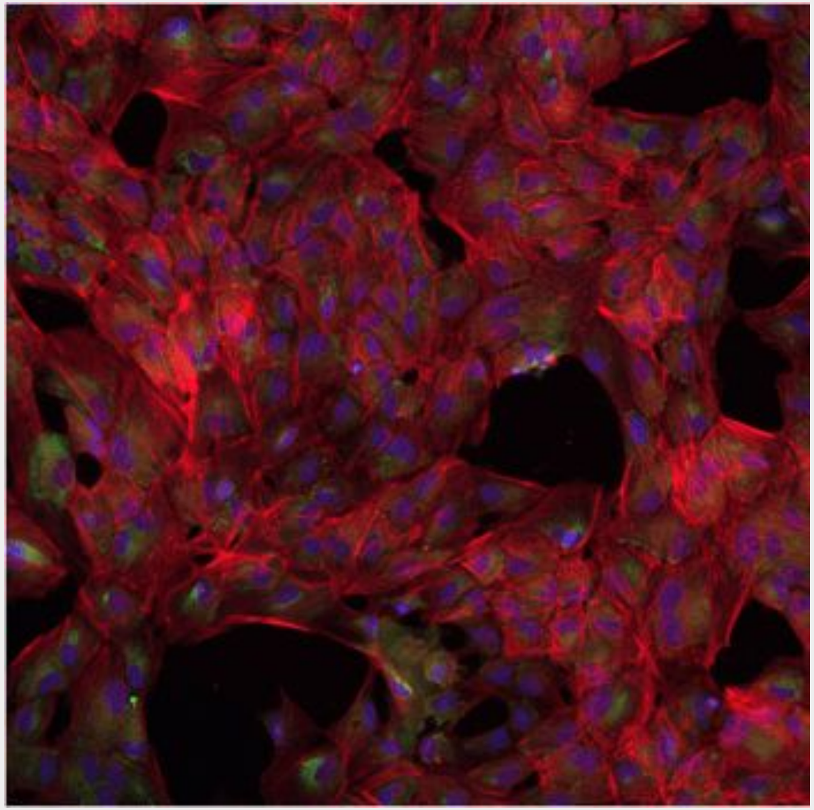
Recent Challenges

Predicting magnetic interactions between pair of atoms



Recent Challenges

Disentangling biological signal from experimental noise in cellular images



Recent Challenges

Build a model to transcribe ancient Kuzushiji into contemporary Japanese characters

Loss function tricks

- Focal Loss / Reduced FL
- Soft Jaccard / Soft Dice loss
- Wing Loss
- Lovazh loss
- Combining losses

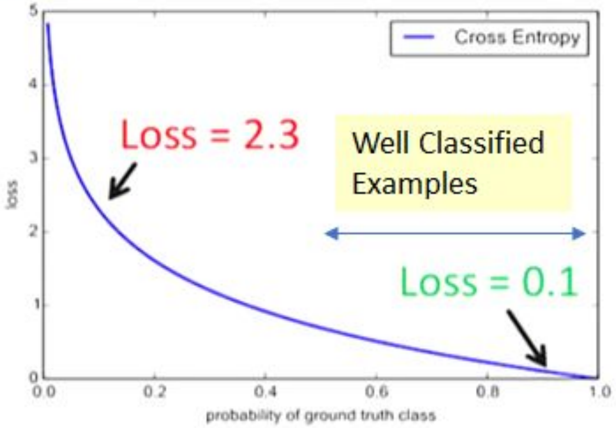
Loss function tricks - Focal Loss

Problem: Huge class imbalance (1:10000)

Solution: Dampen weight of easy examples to total loss, focusing model on hard examples

Outcome: RetinaNet reached SOTA on COCO for single-stage objection detection in 2018

- 100000 easy : 100 hard examples
- **40x bigger loss** from easy examples



Arxiv: https://arxiv.org/abs/1708.02002

Loss function tricks - Focal Loss

Problem: Huge class imbalance (1:10000)

Solution: Dampen weight of easy examples to total loss, focusing model on hard examples

Outcome: RetinaNet reached SOTA on COCO for single-stage objection detection in 2018

$$\mathrm{CE}(p, y) = \begin{cases} -\log(p) & \text{if } y = 1 \\ -\log(1-p) & \text{otherwise.} \end{cases}$$

$$\mathrm{FL}(p_{\mathrm{t}}) = -\alpha_{\mathrm{t}}(1-p_{\mathrm{t}})^{\gamma}\log(p_{\mathrm{t}}).$$

$$p_{\mathrm{t}} = \begin{cases} p & \text{if } y = 1 \\ 1-p & \text{otherwise,} \end{cases}$$

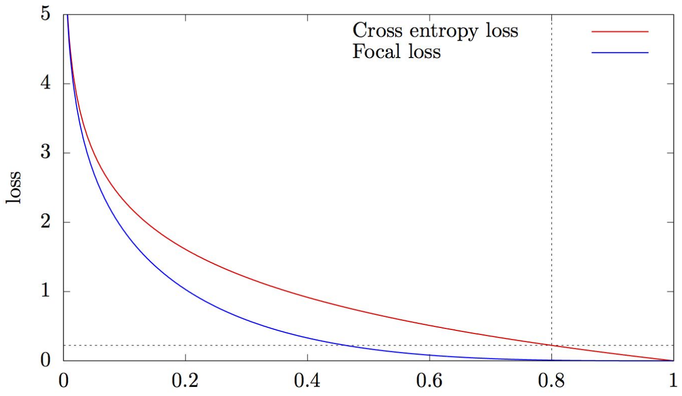
Arxiv: https://arxiv.org/abs/1708.02002

Loss function tricks - Focal Loss

Problem: Huge class imbalance (1:10000)

Solution: Dampen weight of easy examples to total loss, focusing model on hard examples

Outcome: RetinaNet reached SOTA on COCO for single-stage objection detection in 2018



$$\mathrm{FL}(p_t) = -\alpha_t (1 - p_t)^{\gamma} \log(p_t).$$

$$p_t = \begin{cases} p & \text{if } y = 1 \\ 1 - p & \text{otherwise,} \end{cases}$$

Arxiv: https://arxiv.org/abs/1708.02002

Loss function tricks - Reduced Focal Loss

Problem: Low recall for rare classes in object detection

Idea: Quickly dampen loss for ‘good enough’ predictions

Outcome: Recall increased by +11% for rare classes.

Winning solution on “xView object detection in satellite imagery”

Arxiv: https://arxiv.org/abs/1903.01347

Loss function tricks - Reduced Focal Loss

$$RFL(pt) = -fr(pt, th)\log pt$$

$$f(x) = \begin{cases} 1 & : pt < th \\ \dfrac{(1-pt)^{\gamma}}{th^{\gamma}} & : pt \geq th \end{cases}$$

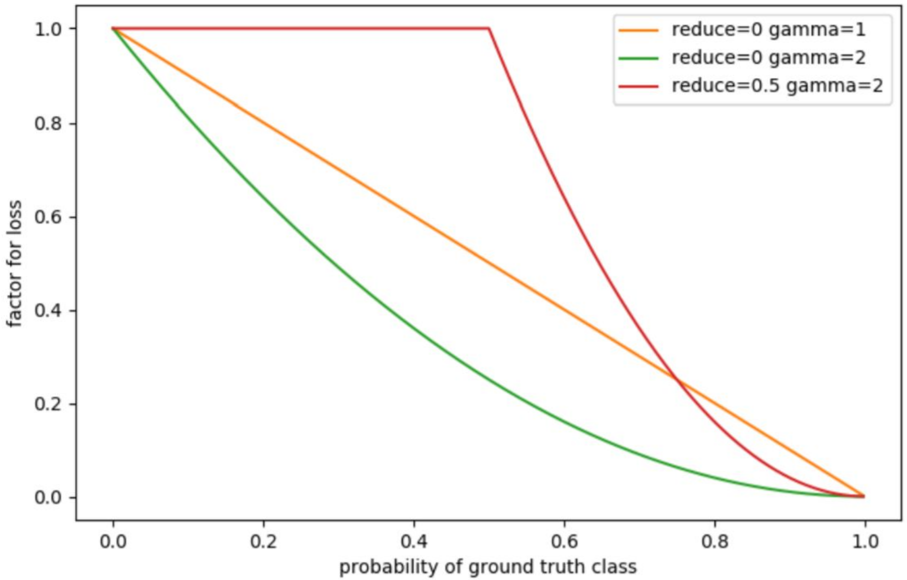


Figure 2: Reduced Focal Loss - Cut-off Factor.

Arxiv: https://arxiv.org/abs/1903.01347

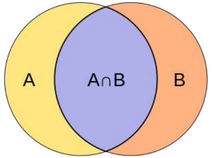
Loss function tricks - Soft Dice / Jaccard

Problem: Binary Cross Entropy is a proxy for target metric (IoU, Dice) for image segmentation

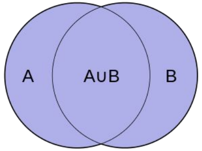
Idea: Optimize target metric directly

Outcome: Target metric optimized directly, model makes more “sharp” masks

$$Dice = \frac{2\,|A \cap B|}{|A| + |B|}$$



$$J(A,B) = \frac{|A \cap B|}{|A \cup B|}$$



Arxiv: https://arxiv.org/pdf/1801.05746.pdf

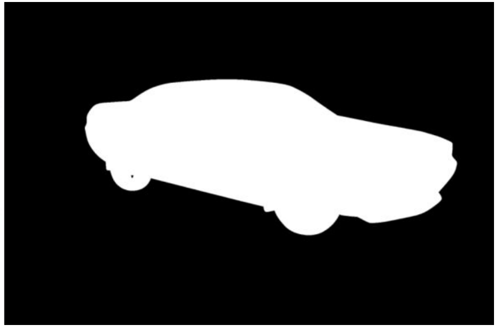
Loss function tricks - Soft Dice / Jaccard

$$Dice = \frac{2\,|A \cap B|}{|A| + |B|}$$

$$|A \cap B| = \underbrace{\begin{bmatrix} 0.01 & 0.03 & 0.02 & 0.02 \\ 0.05 & 0.12 & 0.09 & 0.07 \\ 0.89 & 0.85 & 0.88 & 0.91 \\ 0.99 & 0.97 & 0.95 & 0.97 \end{bmatrix}}_{\text{prediction}} * \underbrace{\begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{bmatrix}}_{\text{target}} \xrightarrow{\text{element-wise multiply}} \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0.89 & 0.85 & 0.88 & 0.91 \\ 0.99 & 0.97 & 0.95 & 0.97 \end{bmatrix} \xrightarrow{\text{sum}} 7.41$$

$$|A| = \begin{bmatrix} 0.01 & 0.03 & 0.02 & 0.02 \\ 0.05 & 0.12 & 0.09 & 0.07 \\ 0.89 & 0.85 & 0.88 & 0.91 \\ 0.99 & 0.97 & 0.95 & 0.97 \end{bmatrix}^{2\ (optional)} \xrightarrow{\text{sum}} 7.82$$

$$|B| = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{bmatrix}^{2\ (optional)} \xrightarrow{\text{sum}} 8$$



Dice = 2 * 7.41 / (7.82+8) = 0.93

Arxiv: https://arxiv.org/pdf/1801.05746.pdf

Loss function tricks - Soft Dice / Jaccard

Used by top-performing teams in

- Carvana Image Masking Channels (1st place)
- 2018 Data Science Bowl (1st place)
- DeepGlobe Building Extraction Challenge (2nd place)
- Airbus Ship Detection Challenge (2nd place)
- TGS Salt Identification Challenge (1st place)

Arxiv: https://arxiv.org/pdf/1801.05746.pdf

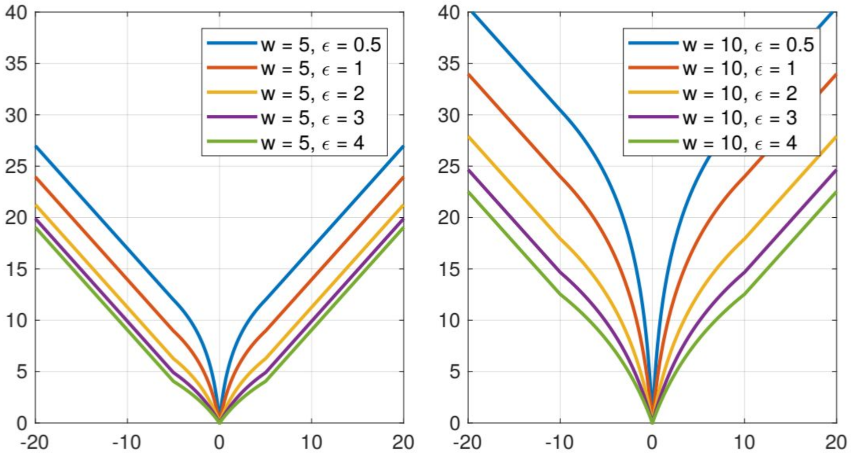
Loss function tricks - Wing Loss

Problem: Mean squared error is sensitive to noise

Solution: Use loss function less sensitive to noise

Outcome: SOTA на facial landmark localization.

Alternatives: Smooth L1, Cauchy



$$wing(x) = \begin{cases} w \ln(1 + |x|/\epsilon) & \text{if } |x| < w \\ |x| - C & \text{otherwise} \end{cases}$$

$$C = w - w \ln(1 + w/\epsilon)$$

Arxiv: https://arxiv.org/abs/1711.06753

Loss function tricks - combining losses

Problem: Low model accuracy

Solution: Use multiple loss functions

Outcome: Changes loss landscape, makes model concentrate on multiple aspects

Example:

- BCE + 0.5 * SoftDice
- 2.0 * Focal + Lovasz
- BCE + Lovasz

Used by top-teams:

- Carvana Image Masking Challenge
- Data Science Bowl 2018
- TGS Salt Identification
- Human protein atlas classification

Arxiv: https://arxiv.org/pdf/1806.00844.pdf

Dataset tricks

- Image augmentation
- Test-time augmentation
- Adversarial validation
- Label smoothing
- Pseudolabeling

Data tricks - Image Augmentation

Problem: Model tends to overfitting, limited amount of data

Solution: Apply image transformation, but keep semantic information

Outcome: Free way to extend dataset and increase its variability

GitHub: https://github.com/albu/albumentations

Data tricks - Image Augmentation

Albumentations library used by

- 3rd place: Dstl Satellite Imagery Feature Detection
- 2nd place: Safe passage: Detecting and classifying vehicles in aerial imagery
- 7th place: Kaggle: Planet: Understanding the Amazon from Space
- 1st place: MICCAI 2017: Gastrointestinal Image ANAlysis (GIANA)
- 1st place: MICCAI 2017: Robotic Instrument Segmentation
- 1st place: Kaggle: Carvana Image Masking Challenge
- 1st place: Topcoder: Urban 3D Challenge
- 1st place: Topcode: SpaceNet Roads Extraction and Routing Challenge
- 9th place: Kaggle: IEEE's Signal Processing Society - Camera Model Identification
- 1st and 10th place: Kaggle: 2018 Data Science Bowl
- 2nd place: CVPR 2018 Deepglobe. Road Extraction.
- 2nd place: CVPR 2018 Deepglobe. Building Detection.
- 3rd place: CVPR 2018 Deepglobe. Land Cover Classification.
- 1st place: TGS Salt Identification.
- 7th place: APTOS2019 Blindness detection.

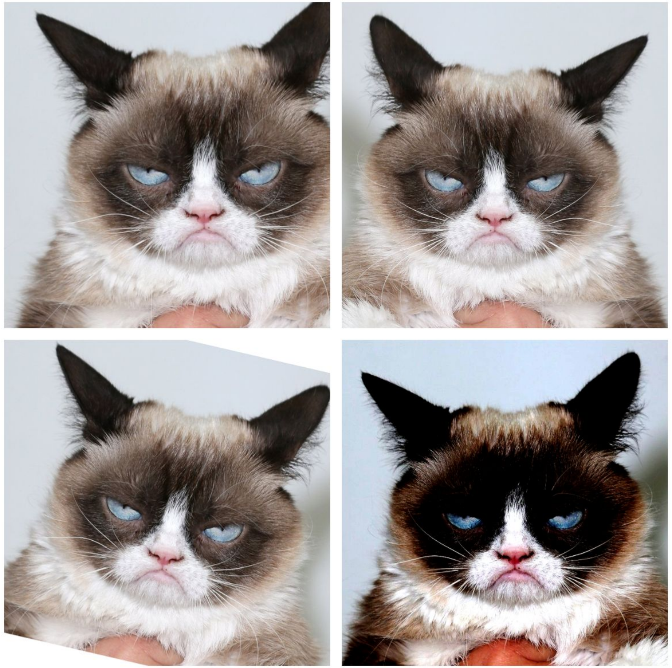
GitHub: https://github.com/albu/albumentations

Data tricks - Test Time Augmentation

Problem: We want to increase model accuracy without re-training it

Solution: Make predictions on fixed set of image transformations and average them

Outcome: Decrease predictions variance



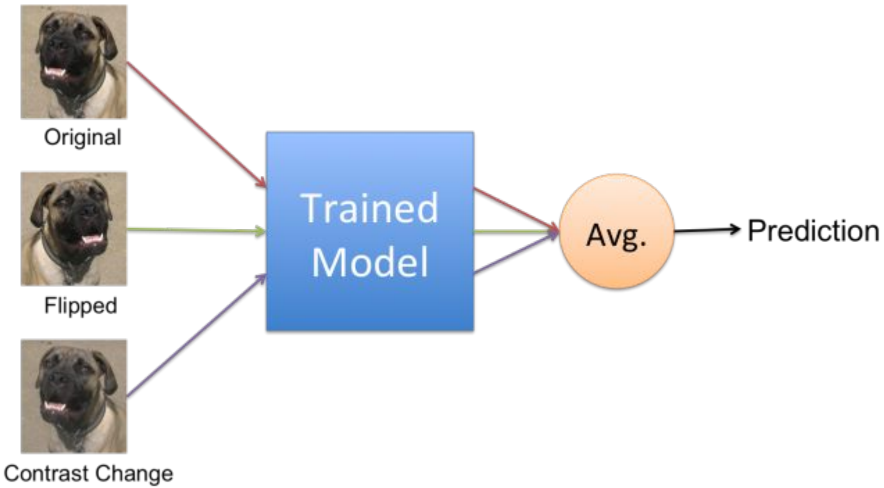
https://arxiv.org/pdf/1807.07356.pdf, https://github.com/BloodAxe/pytorch-toolbelt/blob/develop/pytorch_toolbelt/inference/tta.py

Data tricks - Test Time Augmentation

Problem: We want to increase model accuracy without re-training it

Solution: Make predictions on fixed set of image transformations and average them

Outcome: Decrease predictions variance



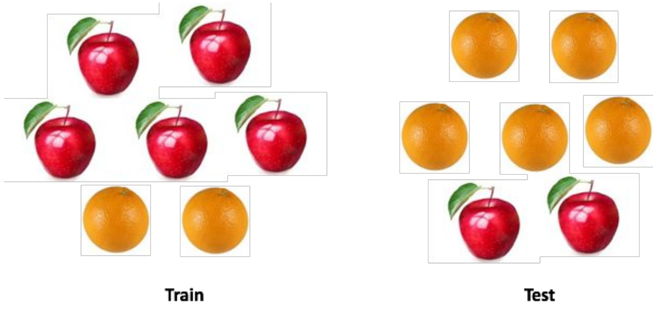
https://arxiv.org/pdf/1807.07356.pdf, https://github.com/BloodAxe/pytorch-toolbelt/blob/develop/pytorch_toolbelt/inference/tta.py

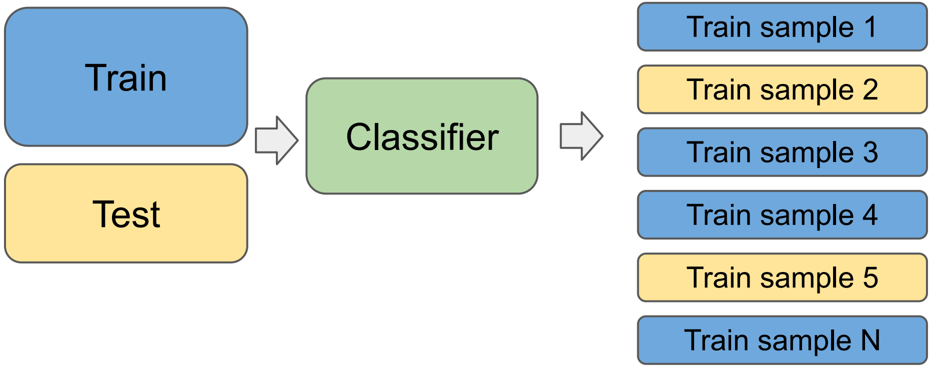
Data tricks - Adversarial validation

Problem: Different data distribution in train and test

Solution: Build a classifier to label train data into train/test classes.

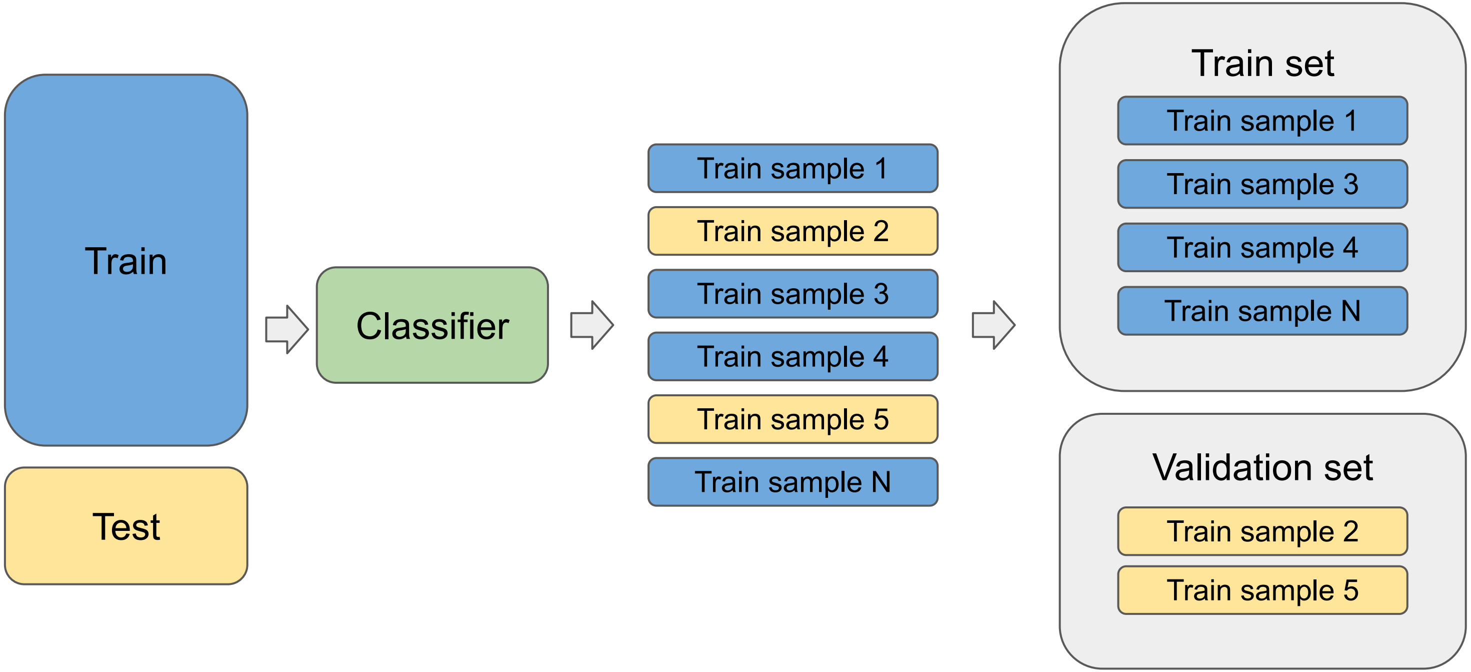
Outcome: Distribution of validation dataset becomes closer to test





https://www.kaggle.com/konradb/adversarial-validation-and-other-scary-terms

Data tricks - Adversarial validation



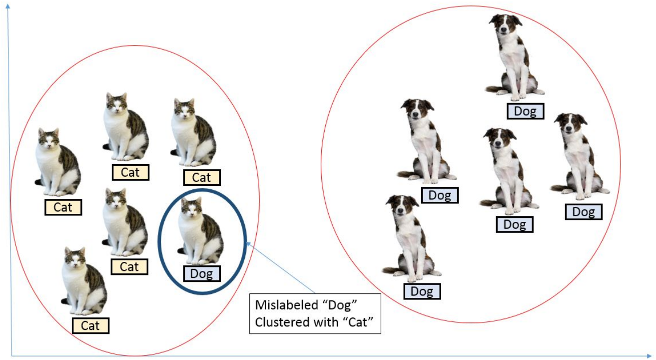
https://www.kaggle.com/konradb/adversarial-validation-and-other-scary-terms

Data tricks - Label smoothing

Problem: Mislabeled samples in trainset

Solution: Instead **{0,1}** targets use smoothed targets: **{eps, 1-eps}**

Outcome: Discourages model from making overconfident predictions



	Cat	Dog
Target	0	1
Probs	0.93	0.07
BCE Loss	1.154	

Smooth 0.2	Cat	Dog
Target	0.2	0.8
Probs	0.93	0.07
BCE Loss	0.923	

Arxiv: https://arxiv.org/abs/1906.02629

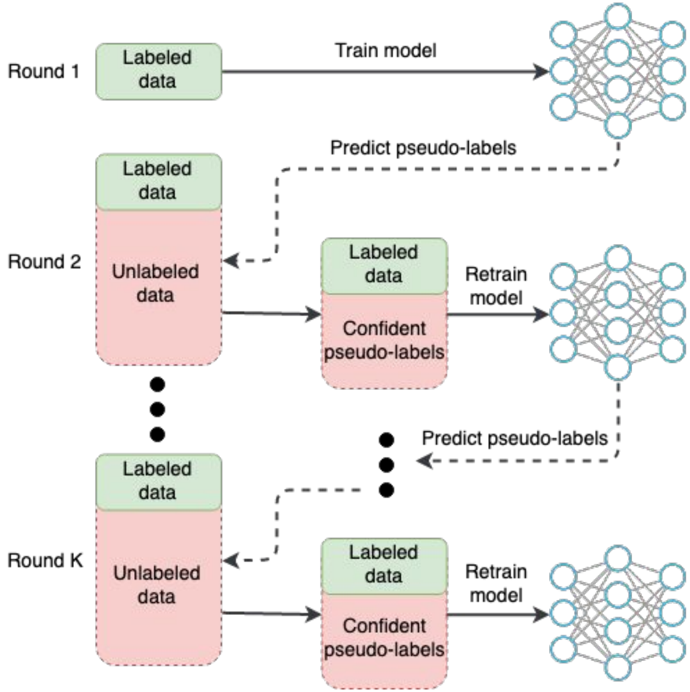
Data tricks - Pseudo-labeling

Problem: There are a lot of unlabeled data

Solution: Use confident model predictions as pseudo-labels and re-train model using it

Outcome: More accurate model

Used by winners of TGS Salt Identification Challenge



Arxiv: https://arxiv.org/abs/1904.04445

Training tricks

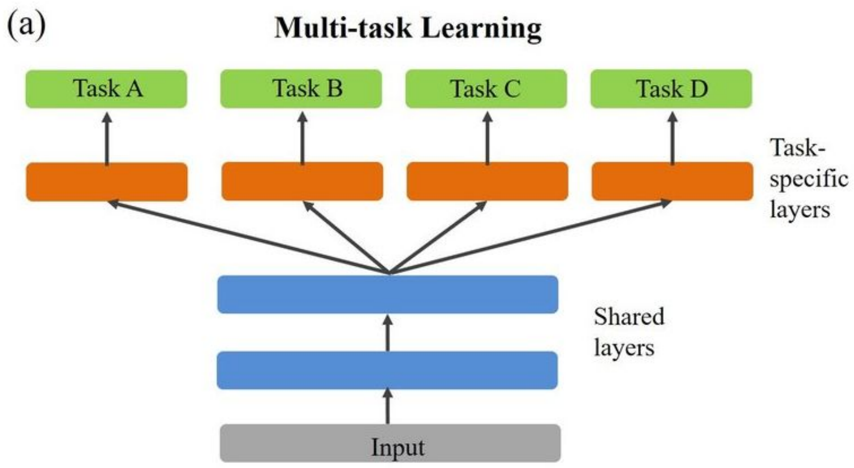
- Multi-task learning
- Cyclic annealing
- Gradient accumulation
- Mixed-precision training
- Metric learning

Training tricks - Multi-task learning

Problem: Model prone to overfitting to irrelevant signals

Solution: Add supplementary tasks to the model

Outcome: Less prone to overfitting. Additional outputs can be used for post-processing



$$\min_{w} \boxed{\sum_{i=1}^{n} V(\hat{x}_i \cdot w, \hat{y}_i)} + \boxed{\lambda \|w\|_2^2}$$

Main optimization criteria

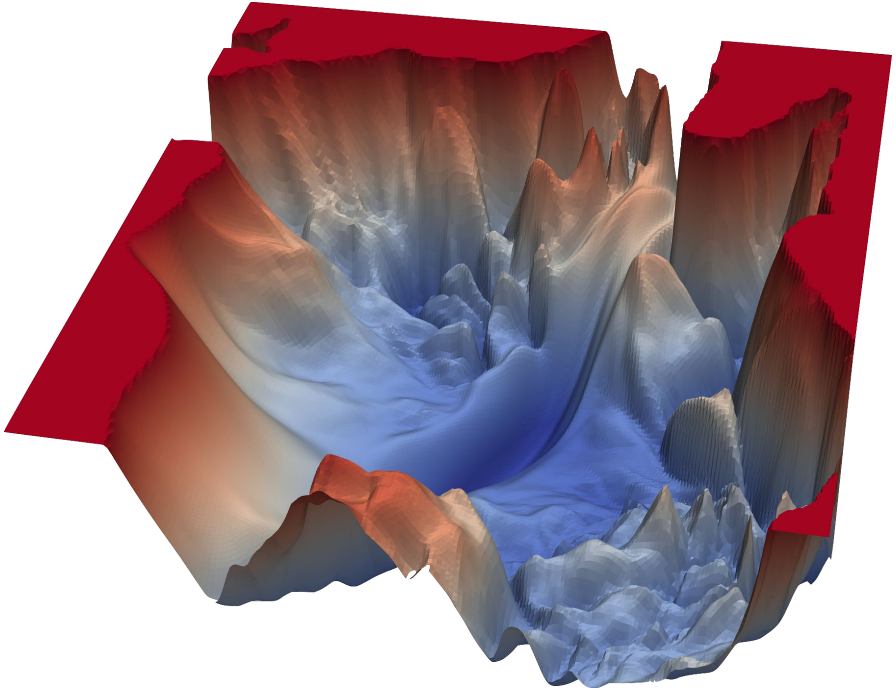
Training tricks - Cyclic annealing

Problem: SGD is slow, Adam introduces large variance

Solution: Repeat training steps:

1) Train model with Adam
2) Train model with SGD

Outcome: Faster model convergence



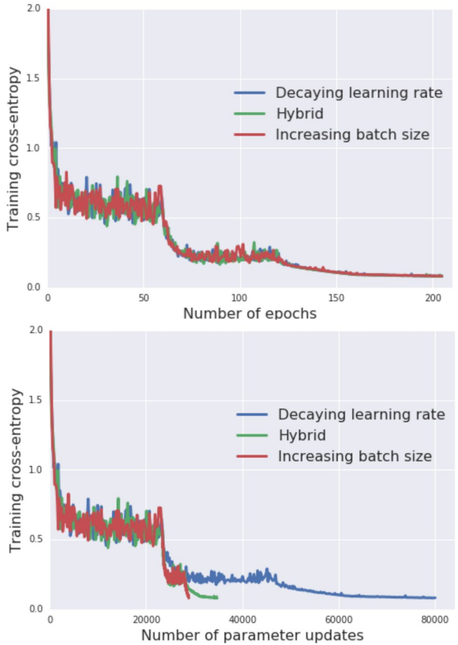
https://github.com/catalyst-team/catalyst

Training tricks - Gradient accumulation

Problem: Small batch size

Solution: Accumulate gradients after each batch and make optimizer step after N batches

Outcome: Simulates an effect of increased batch size for SGD, which is known to converge faster with bigger batch



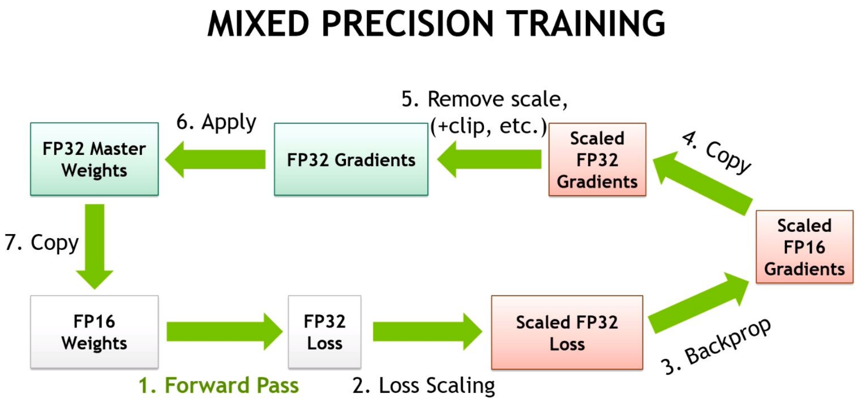
https://arxiv.org/abs/1711.00489

Training tricks - Mixed precision training

Problem: Small batch size

Solution: Use fp16 instead fp32

Outcome: 2x less GPU memory used




```
# Initialization
opt_level = 'O1'
model, optimizer = amp.initialize(model, optimizer, opt_level=opt_level)

# Train your model
...
```

https://arxiv.org/abs/1711.00489

Training tricks - Metric learning

Problem: CrossEntropy + Softmax does not enforce clear decision boundary

Solution: Train model to learn concept of distance.

Outcome: Image embeddings can be directly compared with L2 or cosine distance

http://openaccess.thecvf.com/content_CVPR_2019/papers/Sanakoyeu_Divide_and_Conquer_the_Embedding_Space_for_Metric_Learning_CVPR_2019_paper.pdf

Training tricks - Metric learning

There are many approaches

- Contrastive loss
- Triplet loss
- Center loss
- CosFace
- ArcFace
- Divide&Conquer

http://openaccess.thecvf.com/content_CVPR_2019/papers/Sanakoyeu_Divide_and_Conquer_the_Embedding_Space_for_Metric_Learning_CVPR_2019_paper.pdf

Training tricks - Metric learning

Problem: CrossEntropy + Softmax does not enforce clear decision boundary

Solution: Train model to learn concept of distance.

Outcome: Image embedding's can be directly compared with L2 or cosine distance

Used by top-teams in

- Kaggle Whale identification
- HPA protein classification challenge
- Google landmark retrieval 2019
- Google landmark retrieval 2018

http://openaccess.thecvf.com/content_CVPR_2019/papers/Sanakoyeu_Divide_and_Conquer_the_Embedding_Space_for_Metric_Learning_CVPR_2019_paper.pdf

Thank you

Questions?

Eugene Khvedchenya
ekhvedchenya@gmail.com
linkedin.com/in/cvtalks http://github.com/BloodAxe