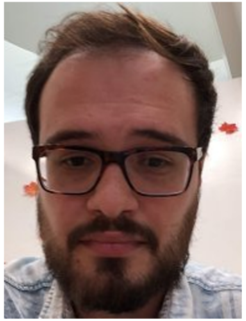
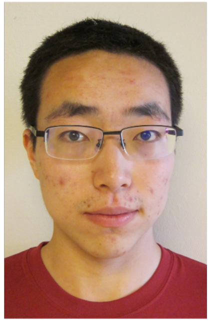
XYZ studies

a paradigm for research in data science

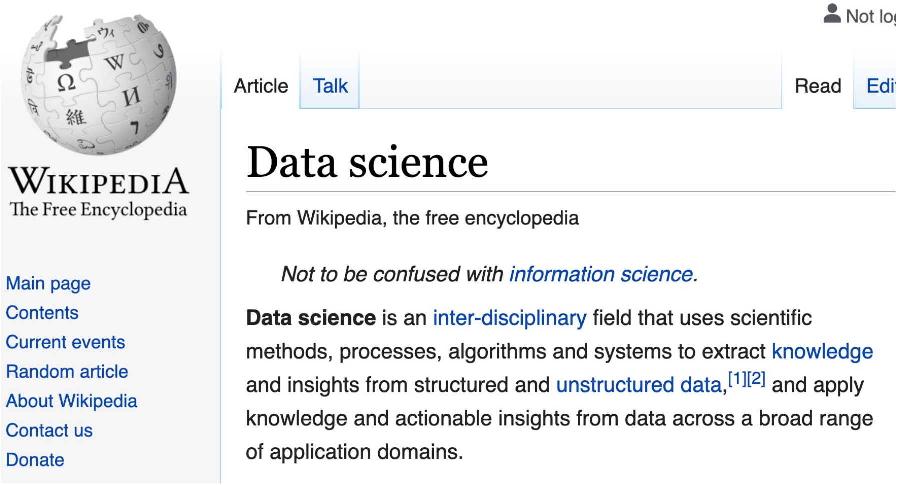
X.Y. Han





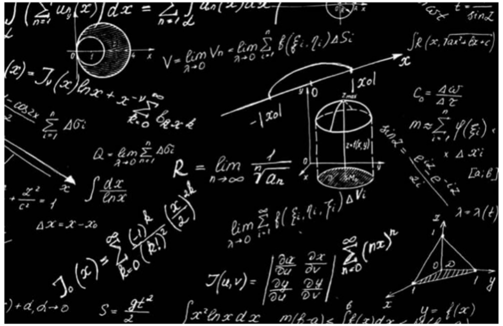

What is data science?

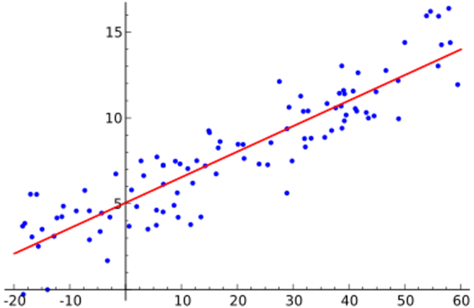
- Data collection
- Database management
- Data-cleaning (wrangling)
- **Research** and **analysis** of data.
  (The "science" part)



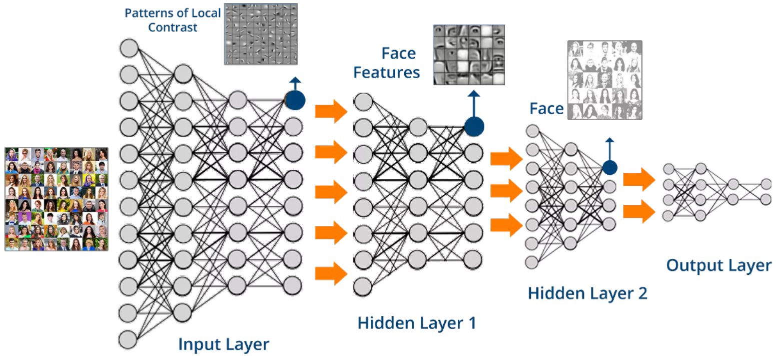
What is data science research?

Statistics:





Empirical Machine Learning:



Drawbacks: Statistics and ML Paradigms

Statistics:

- Reliance on generative models
- Reliance on asymptotic theory
- Focus on mathematical deliverable

Stats Alignment Problem: Deliverables may not be relevant to truth

Empirical Machine learning:

- Reliance on predictive accuracy alone
- Reliance on what works on one dataset
- Conference papers promote “narratives” without solidarity

ML Alignment Problem: Uncertain relationships between poetic deliverables and broader lessons.

XYZ studies

… an important Data Science Paradigm responding to the Statistics/ML Alignment Problems

Y — datasets considered canonical for certain task

X — all relevant methods

Z — control parameters

W — observables of interest

Algorithm 1: Description of XYZ experiment

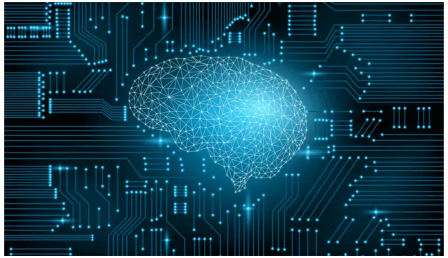
Input : methods X, datasets Y, control parameters Z

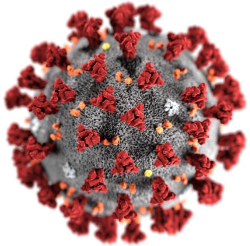
Output: observables W

```
foreach method x ∈ X do
    foreach dataset y ∈ Y do
        foreach control parameter z ∈ Z do
            /* run experiment and collect observables */
            W(x, y, z) = Experiment(x, y, z)
        end
    end
end
```

XYZ in...

- Medical Research (Meta-clinical)
- Empirical ML Research
- COVID-19 Simulation





An Example in Meta-Clinical research

Comparative Meta-analysis of Prognostic Gene Signatures for Late-Stage Ovarian Cancer

Levi Waldron, Benjamin Haibe-Kains, Aedín C. Culhane, Markus Riester, Jie Ding, Xin Victoria Wang, Mahnaz Ahmadifar, Svitlana Tyekucheva, Christoph Bernau, Thomas Risch, Benjamin Frederick Ganzfried, Curtis Huttenhower, Michael Birrer, Giovanni Parmigiani

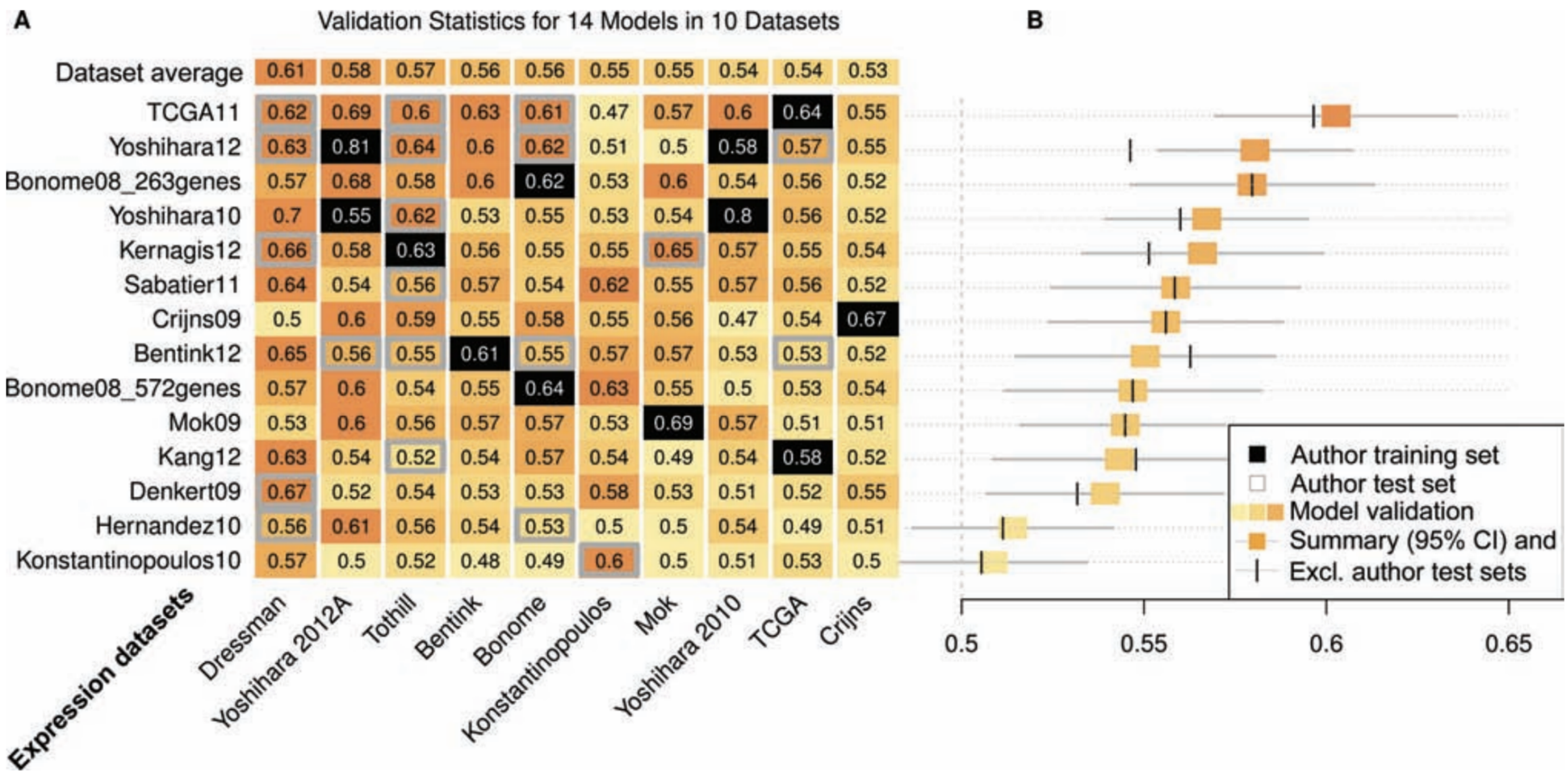
Manuscript received February 24, 2013; revised January 13, 2014; accepted January 29, 2014.

Correspondence to: Giovanni Parmigiani, PhD, Department of Biostatistics and Computational Biology, Dana-Farber Cancer Institute, 450 Brookline Ave, Boston, MA 02115 (e-mail: gp@jimmy.harvard.edu).

Background Ovarian cancer is the fifth most common cause of cancer deaths in women in the United States. Numerous gene signatures of patient prognosis have been proposed, but diverse data and methods make these difficult to compare or use in a clinically meaningful way. We sought to identify successful published prognostic gene signatures through systematic validation using public data.

Methods A systematic review identified 14 prognostic models for late-stage ovarian cancer. For each, we evaluated its 1) reimplementation as described by the original study, 2) performance for prognosis of overall survival in independent data, and 3) performance compared with random gene signatures. We compared and ranked models by validation in 10 published datasets comprising 1251 primarily high-grade, late-stage serous ovarian cancer patients. All tests of statistical significance were two-sided.

Results Twelve published models had 95% confidence intervals of the C-index that did not include the null value of 0.5; eight outperformed 97.5% of signatures including the same number of randomly selected genes and trained on the same data. The four top-ranked models achieved overall validation C-indices of 0.56 to 0.60 and shared anti-correlation with expression of immune response pathways. Most models demonstrated lower accuracy in new datasets than in validation sets presented in their publication.


A
Validation Statistics for 14 Models in 10 Datasets
B
Dataset average 0.61 0.58 0.57 0.56 0.56 0.55 0.55 0.54 0.54 0.53
TCGA11 0.62 0.69 0.6 0.63 0.61 0.47 0.57 0.6 0.64 0.55
Yoshihara12 0.63 0.81 0.64 0.6 0.62 0.51 0.5 0.58 0.57 0.55
Bonome08_263genes 0.57 0.68 0.58 0.6 0.62 0.53 0.6 0.54 0.56 0.52
Yoshihara10 0.7 0.55 0.62 0.53 0.55 0.53 0.54 0.8 0.56 0.52
Kernagis12 0.66 0.58 0.63 0.56 0.55 0.55 0.65 0.57 0.55 0.54
Sabatier11 0.64 0.54 0.56 0.57 0.54 0.62 0.55 0.57 0.56 0.52
Crijns09 0.5 0.6 0.59 0.55 0.58 0.55 0.56 0.47 0.54 0.67
Bentink12 0.65 0.56 0.55 0.61 0.55 0.57 0.57 0.53 0.53 0.52
Bonome08_572genes 0.57 0.6 0.54 0.55 0.64 0.63 0.55 0.5 0.53 0.54
Mok09 0.53 0.6 0.56 0.57 0.57 0.53 0.69 0.57 0.51 0.51
Kang12 0.63 0.54 0.52 0.54 0.57 0.54 0.49 0.54 0.58 0.52
Denkert09 0.67 0.52 0.54 0.53 0.53 0.58 0.53 0.51 0.52 0.55
Hernandez10 0.56 0.61 0.56 0.54 0.53 0.5 0.5 0.54 0.49 0.51
Konstantinopoulos10 0.57 0.5 0.52 0.48 0.49 0.6 0.5 0.51 0.53 0.5
Expression datasets
Dressman
Yoshihara 2012A
Tothill
Bentink
Bonome
Konstantinopoulos
Mok
Yoshihara 2010
TCGA
Crijns
Author training set
Author test set
Model validation
Summary (95% CI) and
Excl. author test sets
0.5
0.55
0.6
0.65

Examples in Empirical ML

https://arxiv.org › cs

Understanding deep learning requires rethinking generalization

by C Zhang · 2016 · Cited by 2642

Perfect score on the ICLR reviews

ICLR 2017 best paper award

OCT 13, 2017 @ 01:23 PM 7,420

What You Need To Know About One Of The Most Talked-About Papers On Deep Learning To Date

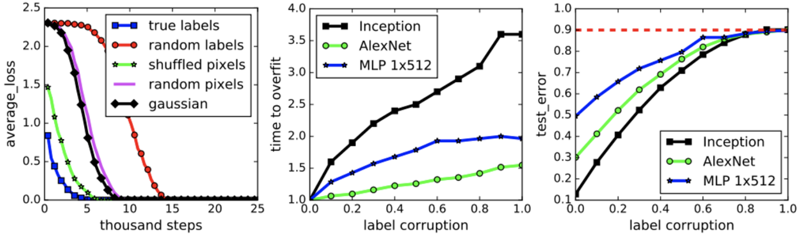
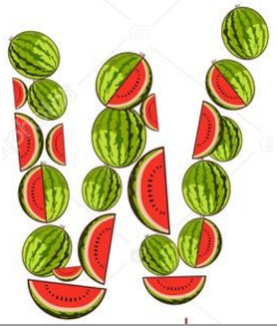


Figure 1: Fitting random labels and random pixels on CIFAR10. (a) shows the training loss of various experiment settings decaying with the training steps. (b) shows the relative convergence time with different label corruption ratio. (c) shows the test error (also the generalization error since training error is 0) under different label corruptions.

Table 2: The top-1 and top-5 accuracy (in percentage) of the Inception v3 model on the ImageNet dataset. We compare the training and test accuracy with various regularization turned on and off, for both true labels and random labels. The original reported top-5 accuracy of the Alexnet on ILSVRC 2012 is also listed for reference. The numbers in parentheses are the best test accuracy during training, as a reference for potential performance gain of early stopping.

data aug	dropout	weight decay	top-1 train	top-5 train	top-1 test	top-5 test
ImageNet 1000 classes with the original labels						
yes	yes	yes	92.18	99.21	77.84	93.92
yes	no	no	92.33	99.17	72.95	90.43
no	no	yes	90.60	100.0	67.18 (72.57)	86.44 (91.31)
no	no	no	99.53	100.0	59.80 (63.16)	80.38 (84.49)
Alexnet (Krizhevsky et al., 2012)			-	-	-	83.6
ImageNet 1000 classes with random labels						
no	yes	yes	91.18	97.95	0.09	0.49
no	no	yes	87.81	96.15	0.12	0.50
no	no	no	95.20	99.14	0.11	0.56

					
Rethinking Generalization	CIFAR10, ImageNet	MLP, AlexNet, Inception	% randomized labels	number of epochs until perfect fit, test error at epoch of perfect fit	Could be done on more datasets and methods

Examples in Empirical ML

https://arxiv.org › stat

Are GANs Created Equal? A Large-Scale Study

by M Lucic · 2017 · Cited by 548 ·

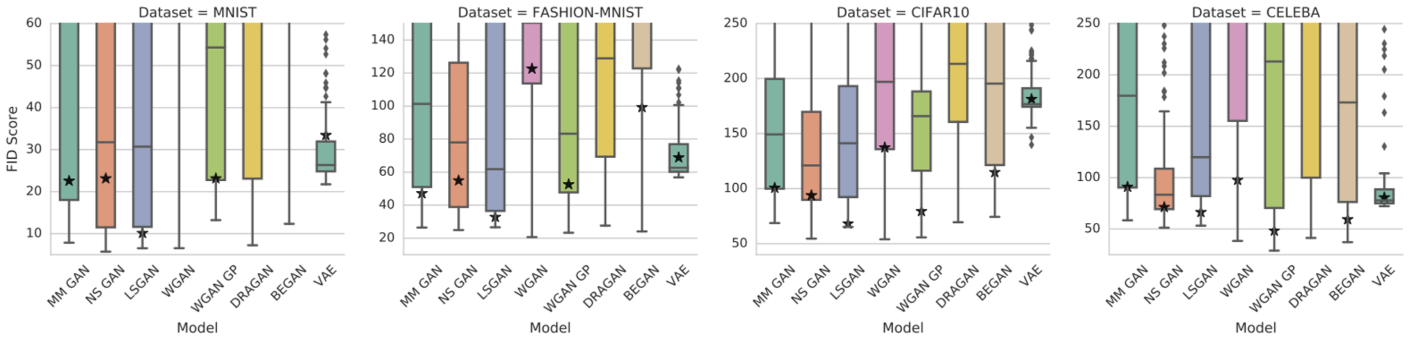
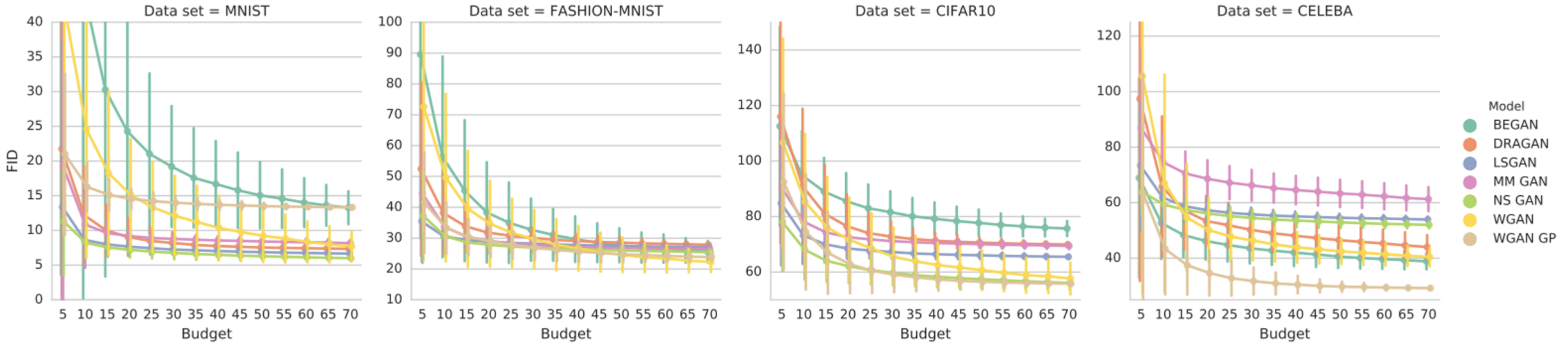
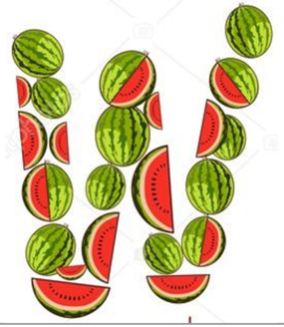


Figure 4: A *wide range* hyperparameter search (100 hyperparameter samples per model). Black stars indicate the performance of suggested hyperparameter settings. We observe that GAN training is extremely sensitive to hyperparameter settings and there is no model which is significantly more stable than others.



					
Are GANs Created Equal? Lucic et. al	MNIST, FASHION - MNIST, CIFAR10,	MM GAN, NS GAN, LSGAN, WGAN, WGAN GP, DRAGAN, BEGAN, VAE	seed, computational budget	precision, recall, F1, FID	Great example!

Decision Making and COVID-19

REOPENING SCHOOLS

Stanford University Inviting Juniors and Seniors Back to Campus for Spring Classes

The University noted that most undergraduate instruction would continue to be remote.

By Bay City News • Published March 14, 2021 • Updated on March 15, 2021 at 8:34 am

COVID-19 and Reactivation Planning

NEWS

Cornell University To Require COVID-19 Vaccine For On-Campus Students

BY SYDNEY PEREIRA

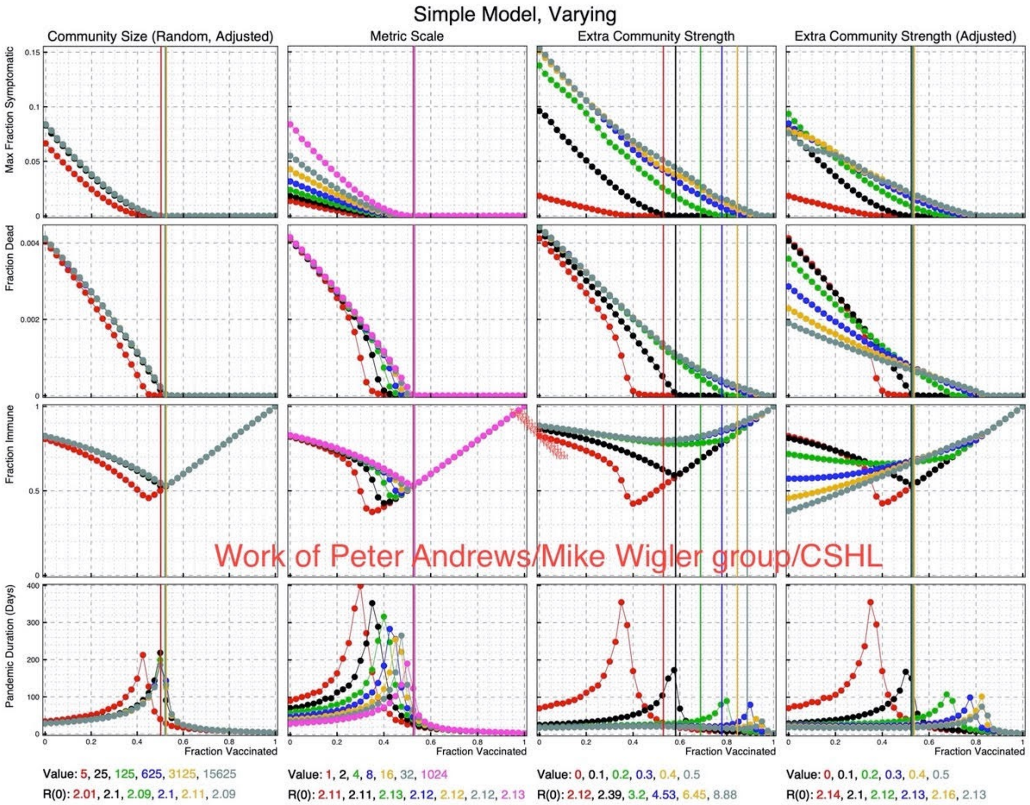
APRIL 4, 2021 12:16 P.M. • 23 COMMENTS

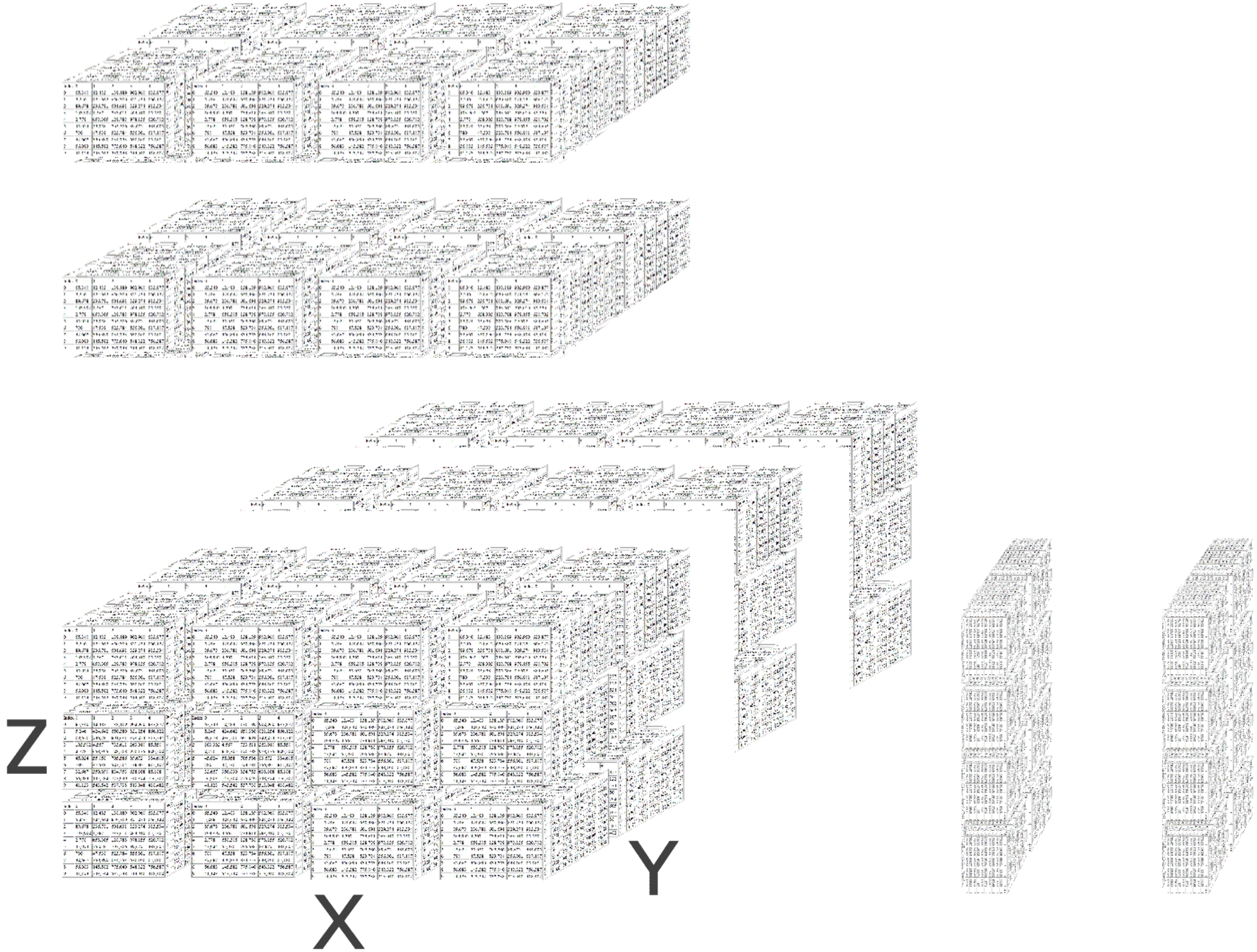
Epidemiological Modeling

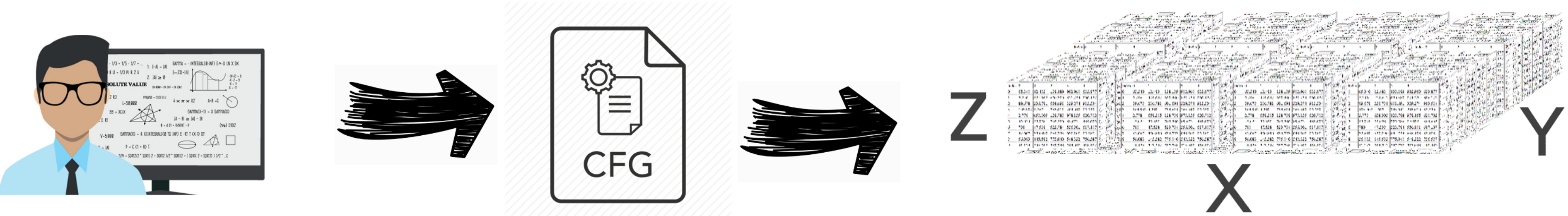
The health of our campus community and the greater Ithaca area were key considerations in Cornell's plan to invite students to campus for instruction. To guide this decision-making, the university relied on numerous evidence-based sources, including the findings of epidemiological modeling by experts on our faculty.

How much can we trust simulated models?

Examples in COVID-19 Simulations




Z
X
Y



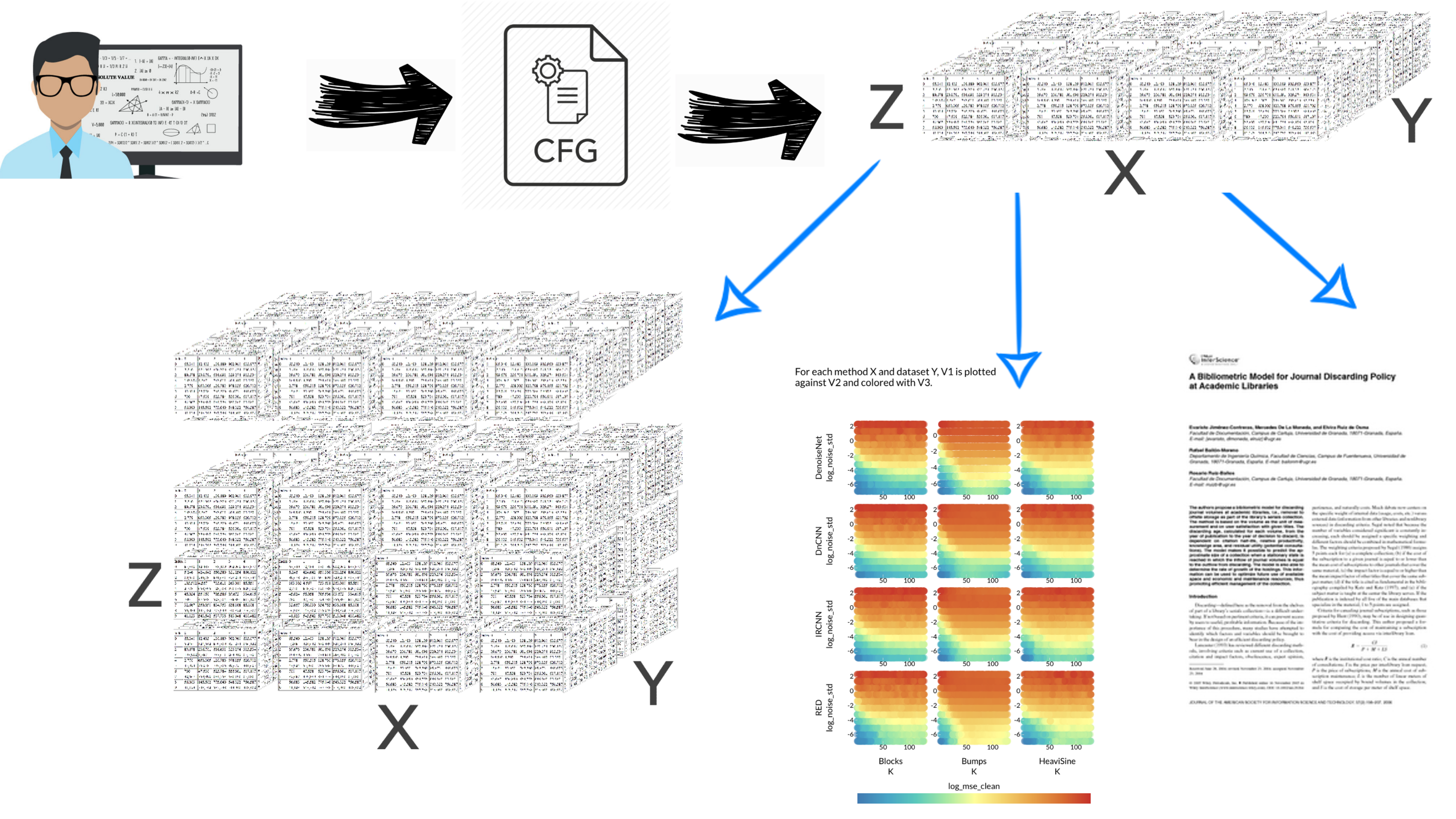
hadoop

CodaLab

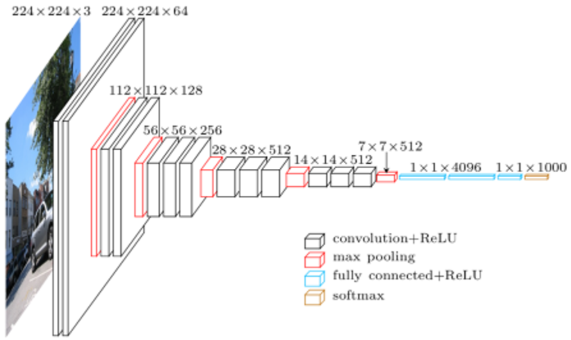
APACHE Spark

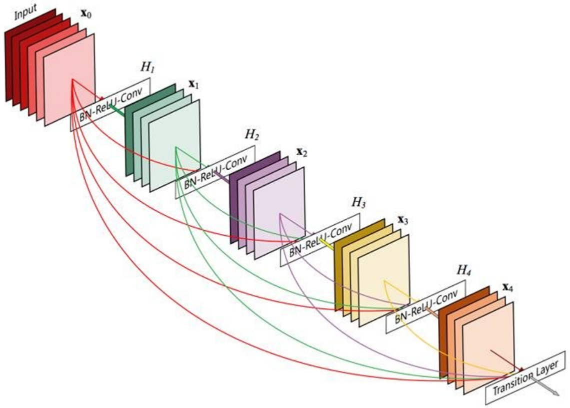
ElastiCluster

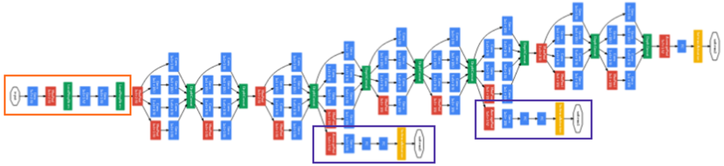
Caffe
Chainer
DL4J Deeplearning4j
KERAS
Microsoft CNTK
MatConvNet
MINERVA
mxnet
Purine
TensorFlow
theano
torch


CFG
Z
X
Y
For each method X and dataset Y, V1 is plotted against V2 and colored with V3.
DenoiseNet log_noise_std
DnCNN log_noise_std
IRCNN log_noise_std
RED log_noise_std
Blocks K
Bumps K
HeaviSine K
log_mse_clean
A Bibliometric Model for Journal Discarding Policy at Academic Libraries
Z
X
Y

```
net_list          = [
                    'CNN',
                    'AlexNet',
                    'VGG11_bn',
                    'VGG13_bn',
                    'VGG16_bn',
                    'VGG19_bn',
                    'ResNet18',
                    'ResNet34',
                    'ResNet50',
                    'ResNet101',
                    'ResNet152',
                    'SqueezeNet_1_0',
                    'SqueezeNet_1_1',
                    'DenseNet121',
                    'DenseNet161',
                    'DenseNet169',
                    'DenseNet201',
                    'Inception3'
                   ]
```







```
dataset_list    = [
                    'MNIST',
                    'FashionMNIST',
                    'EMNIST_byclass',
                    'EMNIST_bymerge',
                    'EMNIST_balanced',
                    'EMNIST_letters',
                    'EMNIST_digits',
                    'CIFAR10',
                    'CIFAR100',
                    'STL10',
                    'SVHN',
                  ]
```

```
lr_list             = [
                        0.5,
                        0.25,
                        0.1,
                        0.075,
                        0.05,
                        0.025,
                        0.01,
                        0.0075,
                        0.0050,
                        0.0025,
                        0.001,
                        0.00075,
                        0.0005,
                        0.00025,
                        0.0001,
                        ]
```

XYZ experiment

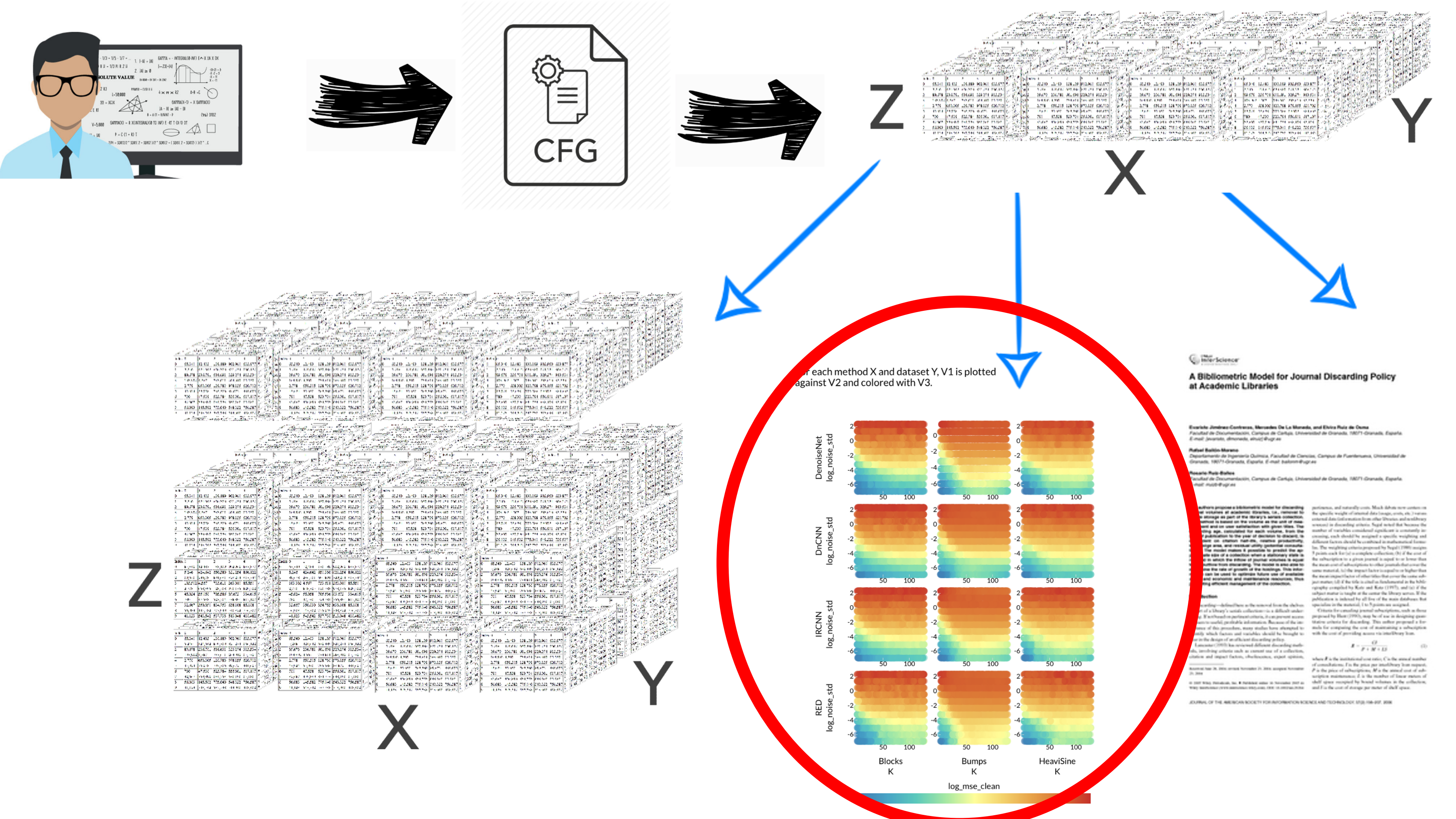
```
for model_name in [...]:
    for dataset_name in [...]:
        for learning_rate in [...]:

            network = create_model(model_name)
            dataset = create_dataset(dataset_name)

            for epoch in range(num_epochs):
                for image, target in dataset:

                    # forward pass
                    output = network(image)

                    # backward pass
                    loss(output, target).backward()

                    # update model
                    optimizer.step(learning_rate)

                    # compute accuracy
                    acc = compute_accuracy()

                    # save to csv
                    save_results(acc)
```

save **EVERYTHING** about the experiment in the CSV

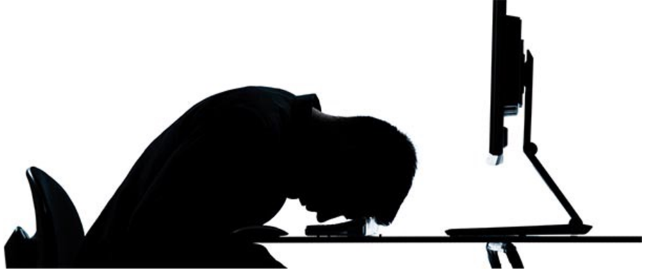
XYZ experiment in practice

```
loader_opts  = {'train_dataset'      : str(row['train_dataset']),
                'test_dataset'       : row['test_dataset'],
                'phase'              : None,
                'loader_type'        : str(row['loader_type']),
                'pytorch_dataset'    : bool(row['pytorch_dataset']),
                'dataset_path'       : '../../data',
                'dataset_path'       : '/scratch/users/papyan/datasets',
                'dataset_kwargs'     : {},
                'im_size'            : int(row['im_size']),
                'padded_im_size'     : int(row['padded_im_size']),
                'num_classes'        : int(row['num_classes']),
                'input_ch'           : int(row['input_ch']),
                'threads'            : 0,
                'limited_dataset'    : bool(row['limited_dataset']),
                'examples_per_class': int(row['examples_per_class']),
                'epc_seed'           : epc_seed_idx,
                'train_seed'         : train_seed_idx,
                'size_list'          : str(row['size_list']),
                'pretrained'         : bool(row['pretrained']),
                'multilabel'         : bool(row['multilabel']),
                'corrupt_prob'       : 0,
                'test_trans_only'    : True,
                'concat_loader'      : False,
                'loader_constructor': Constructor,
                'drop_last'          : False,
               }

train_opts   = {'crit'               : str(row['crit']),
                'net'                : str(row['net']),
                'optim'              : str(row['optim']),
                'epochs'             : int(row['epochs']),
                'lr'                 : float(row['lr']),
                'milestones_perc'    : str(row['milestones_perc']),
                'gamma'              : float(row['gamma']),
                'train_batch_size'   : 128,
                'test_batch_size'    : 128,
                'cuda'               : torch.cuda.is_available(),
                'seed'               : int(row['seed']),
                'epsi'               : float(row['seed']),
               }
```

```
results_opts = {'training_results_path': training_results_path,
                'train_dump_file'    : str(row['train_dump_file']),
                'save_init_epoch'    : bool(row['save_init_epoch']),
                'garbage_collect'    : bool(row['garbage_collect']),
                'save_middle'        : bool(row['save_middle']),
               }

cpu_opts     = {'one_batch'          : bool(row['one_batch'])}

anals_opts   = {'k'                  : float('inf'),
                'project_last'       : False,
                'anals_results_path': analysis_results_path,
                'do_visual'          : False,
                'embedded_max_examples': 512,
                'stats_max_examples': float('inf'),
                'save_Sigma_wc'      : True,
                'vgg_remove_last_dropout' : True,
                'reset_classifier'   : True,
                'analyze_last_only' : True,
                'l_analysis'         : l,
                'layers_func'        : 'get_imp_layers',
                'hook_type'          : 'output',
                'activations_per_example': 10,
                'distribution'       : 'norm',
                'coeff_max_examples': 1000,
                'single_coeff_model': True,
                'record_activation' : False,
                'compute_norm_mean' : False,
                'compute_Sigma_b_w' : False,
                'compute_w_norm_mean': True,
                'compute_t_norm_mean': True,
                'power'              : 0.75,
                'seed'               : False,
               }

spectral_opts = {'hessian_type'      : hessian_type_list[hessian_type_i
                 'init_poly_deg'     : 64,
                 'poly_deg'          : 256, # paper suggests M=100
                 'mat_vec_iters'     : float('inf'),
                 'poly_points'       : 2**9,
                 'spectrum_margin'   : 0.05,
                 'log_hessian'       : False,
                 'start_eig_range'   : -float('inf'),
                 'stop_eig_range'    : float('inf'),
                 'power_method_iters': 256,
                 'repeat_idx'        : repeat_idx,
                }
```


CFG
Z
X
Y
each method X and dataset Y, V1 is plotted against V2 and colored with V3.
DenoiseNet
DnCNN
IRCNN
RED
log_noise_std
Blocks
Bumps
HeaviSine
K
log_mse_clean
A Bibliometric Model for Journal Discarding Policy at Academic Libraries

Me coding plots on python:



```python
import pandas as pd
import matplotlib.pyplot as plt

df = get_data_frame(path_to_csv)

colors = cm.rainbow(np.linspace(0, 1, num_learning_rates))

for dataset in [...]:
    for net in [...]:
        for learning_rate in [...]:

            df = df[(df['dataset'] == dataset)
                  & (df['net'] == net)
                  & (df['learning_rate'] == learning_rate)]

            plt.plot(df.epoch, df.accuracy, color=colors[learning_rate])
            plt.title('dataset: {}, net: {}, learning_rate: {}',
                      dataset,
                      net,
                      learning rate)
```

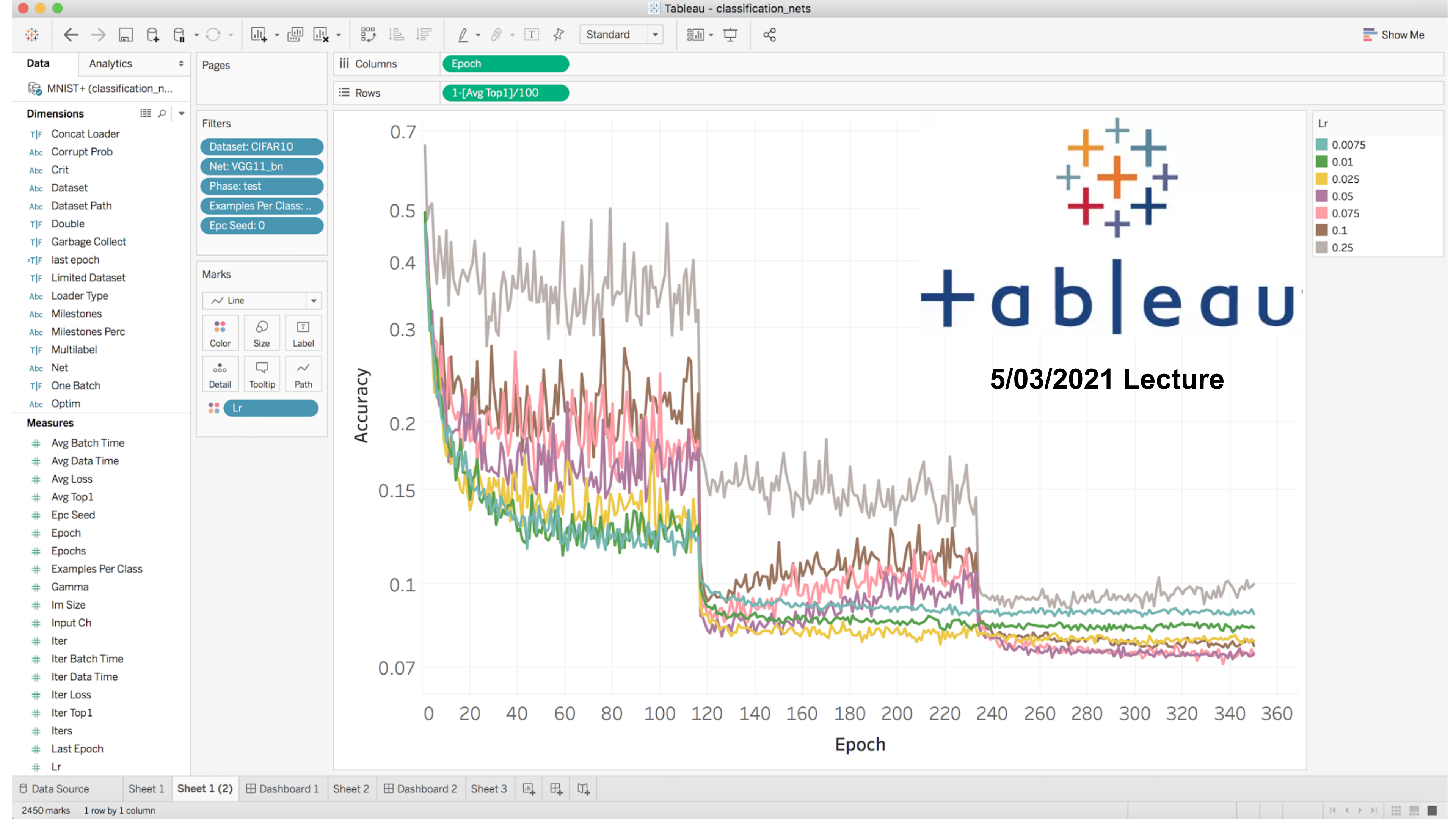


Tableau - classification_nets
Standard
Show Me
Data
Analytics
MNIST+ (classification_n...
Dimensions
Concat Loader
Corrupt Prob
Crit
Dataset
Dataset Path
Double
Garbage Collect
last epoch
Limited Dataset
Loader Type
Milestones
Milestones Perc
Multilabel
Net
One Batch
Optim
Measures
Avg Batch Time
Avg Data Time
Avg Loss
Avg Top1
Epc Seed
Epoch
Epochs
Examples Per Class
Gamma
Im Size
Input Ch
Iter
Iter Batch Time
Iter Data Time
Iter Loss
Iter Top1
Iters
Last Epoch
Lr
Pages
Columns
Epoch
Rows
1-[Avg Top1]/100
Filters
Dataset: CIFAR10
Net: VGG11_bn
Phase: test
Examples Per Class: ..
Epc Seed: 0
Marks
Line
Color
Size
Label
Detail
Tooltip
Path
Lr
Accuracy
0.7
0.5
0.4
0.3
0.2
0.15
0.1
0.07
0
20
40
60
80
100
120
140
160
180
200
220
240
260
280
300
320
340
360
Epoch
tableau
5/03/2021 Lecture
Lr
0.0075
0.01
0.025
0.05
0.075
0.1
0.25
Data Source
Sheet 1
Sheet 1 (2)
Dashboard 1
Sheet 2
Dashboard 2
Sheet 3
2450 marks
1 row by 1 column

Tableau is...

- **P**owerful: can compute mathematical expressions
- **E**fficient: can handle tens of GB easily
- **R**: you write R scripts (can do regression!)
- **F**ast: few clicks to create plot
- **E**asy: drag and drop
- **C**loud: data sits on cloud
- **T**ime: spent on more useful things

Tableau-Generated Plot:

Papyan, Vardan, X. Y. Han, and David L. Donoho. "Prevalence of Neural Collapse during the terminal phase of deep learning training." *Proceedings of the National Academy of Sciences* 117, no. 40 (2020): 24652-24663.

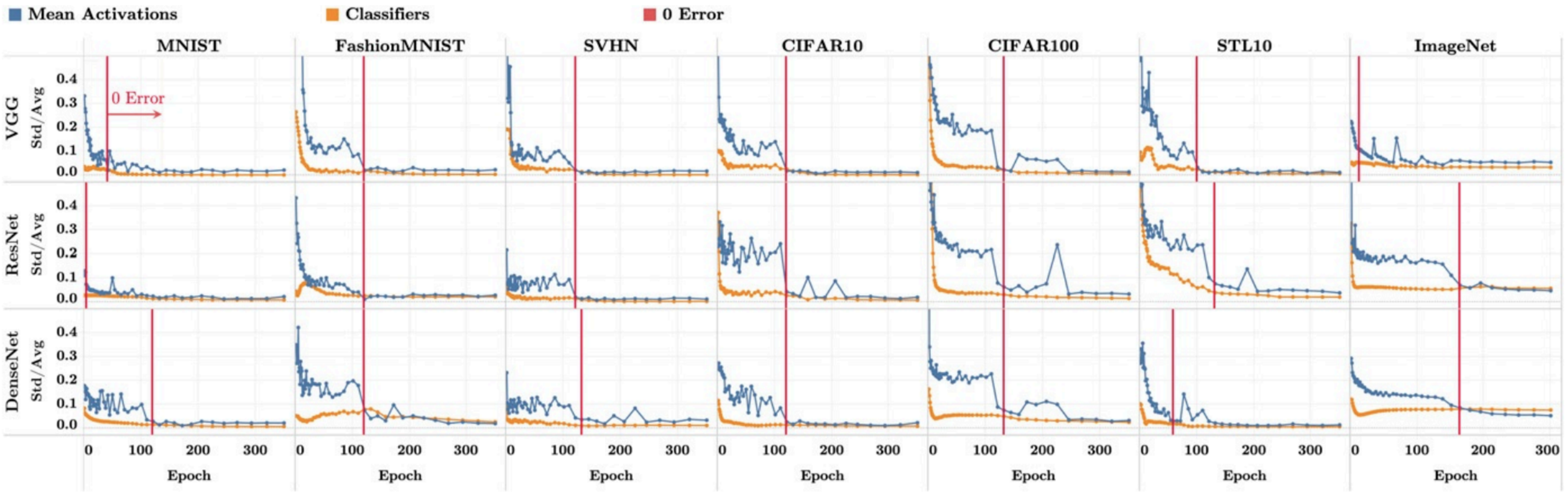
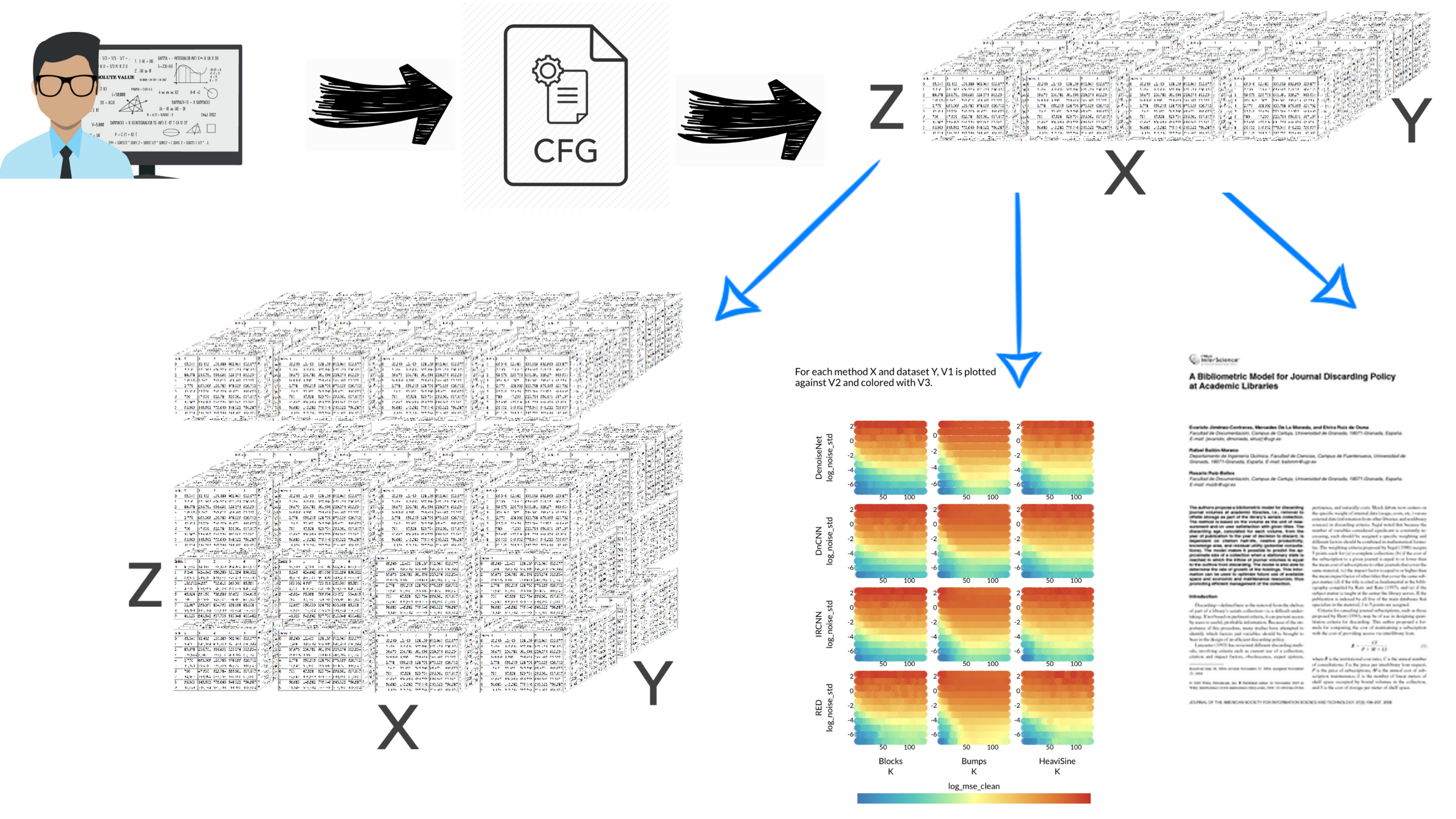
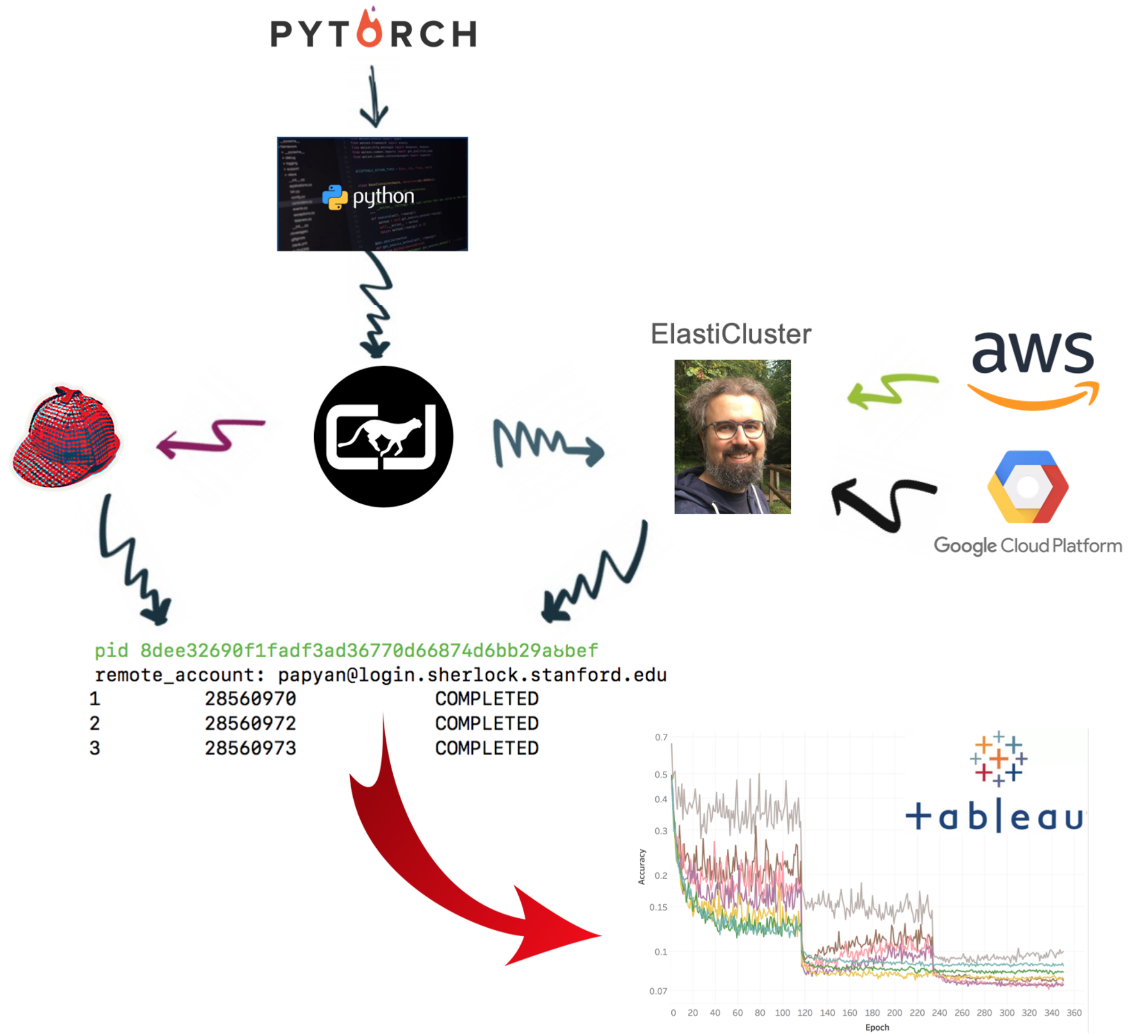
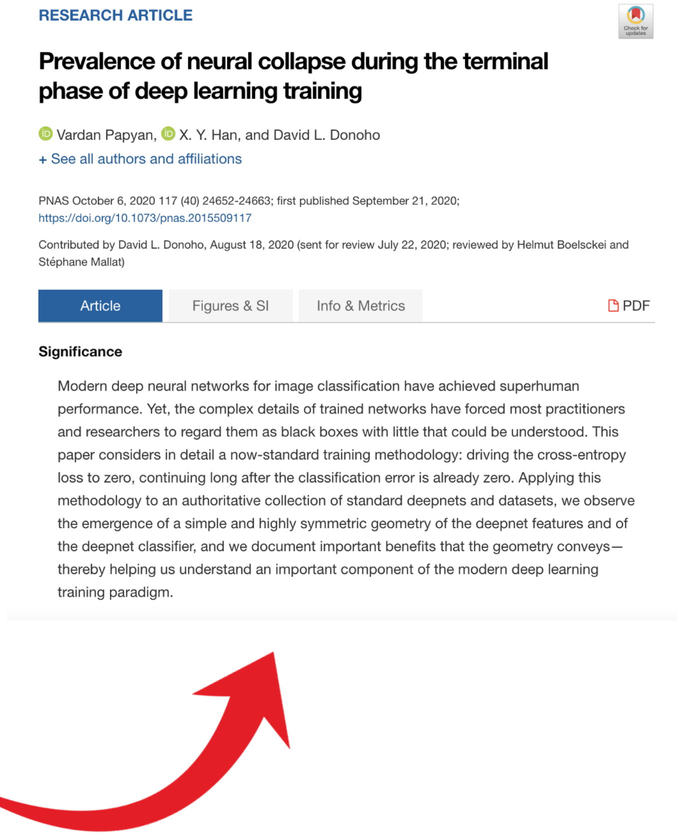


Fig. 2. Train class means become equinorm. The formatting and technical details are as described in Section 3. In each array cell, the vertical axis shows the coefficient of variation of the centered class-mean norms as well as the network classifiers norms. In particular, the blue lines show $\text{Std}_c(\|\boldsymbol{\mu}_c - \boldsymbol{\mu}_G\|_2)/\text{Avg}_c(\|\boldsymbol{\mu}_c - \boldsymbol{\mu}_G\|_2)$ where $\{\boldsymbol{\mu}_c\}$ are the class means of the last-layer activations of the training data and $\boldsymbol{\mu}_G$ is the corresponding train global mean; the orange lines show $\text{Std}_c(\|\boldsymbol{w}_c\|_2)/\text{Avg}_c(\|\boldsymbol{w}_c\|_2)$ where $\boldsymbol{w}_c$ is the last-layer classifier of the cth class. As training progresses, the coefficients of variation of both class means and classifiers decrease.


CFG
Z
X
Y
Z
X
Y
For each method X and dataset Y, V1 is plotted against V2 and colored with V3.
DenoiseNet log_noise_std
DnCNN log_noise_std
IRCNN log_noise_std
RED log_noise_std
Blocks K
Bumps K
HeaviSine K
log_mse_clean
A Bibliometric Model for Journal Discarding Policy at Academic Libraries





RESEARCH ARTICLE

Prevalence of neural collapse during the terminal phase of deep learning training

Vardan Papyan, X. Y. Han, and David L. Donoho

+ See all authors and affiliations

PNAS October 6, 2020 117 (40) 24652-24663; first published September 21, 2020; https://doi.org/10.1073/pnas.2015509117

Contributed by David L. Donoho, August 18, 2020 (sent for review July 22, 2020; reviewed by Helmut Boelsckei and Stéphane Mallat)

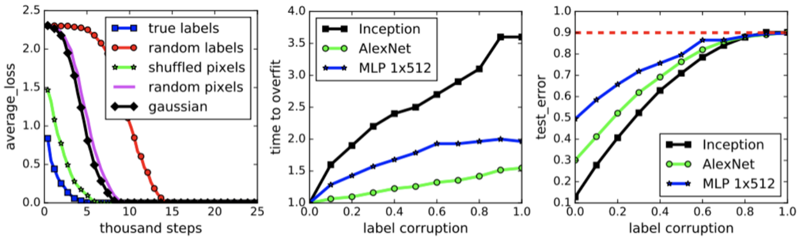
Article | Figures & SI | Info & Metrics | PDF

Significance

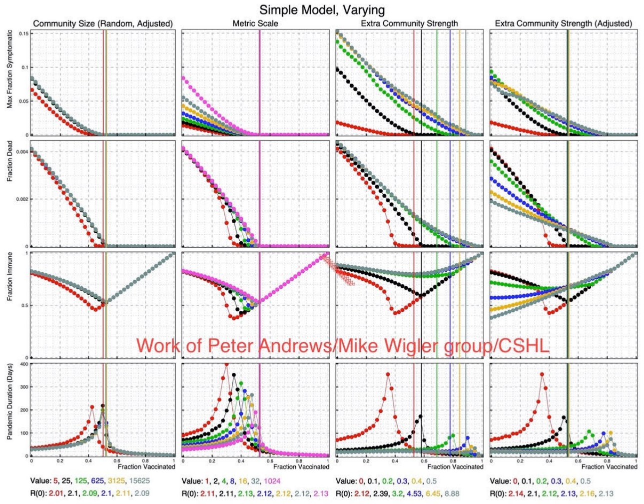
Modern deep neural networks for image classification have achieved superhuman performance. Yet, the complex details of trained networks have forced most practitioners and researchers to regard them as black boxes with little that could be understood. This paper considers in detail a now-standard training methodology: driving the cross-entropy loss to zero, continuing long after the classification error is already zero. Applying this methodology to an authoritative collection of standard deepnets and datasets, we observe the emergence of a simple and highly symmetric geometry of the deepnet features and of the deepnet classifier, and we document important benefits that the geometry conveys—thereby helping us understand an important component of the modern deep learning training paradigm.

XYZ Paradigm for Data Science Research

- Clear insights seen immediately from XYZ grid.
- Real phenomena rather than generative models.
- One massive experiment making a convincing point rather than multiple small ones.
- Data Science Research: *Productively*.



(a) learning curves (b) convergence slowdown (c) generalization error growth



Comments?

Questions?

Epilogue: An XYZ Story

RESEARCH ARTICLE

Check for updates

Prevalence of neural collapse during the terminal phase of deep learning training

Vardan Papyan, X. Y. Han, and David L. Donoho
+ See all authors and affiliations

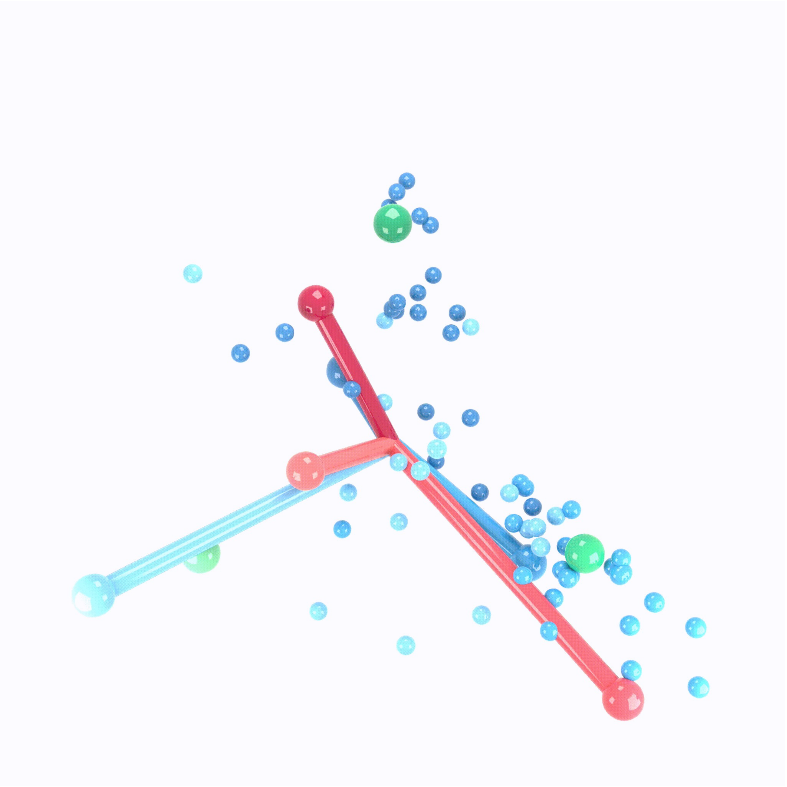
PNAS October 6, 2020 117 (40) 24652-24663; first published September 21, 2020;
https://doi.org/10.1073/pnas.2015509117

Contributed by David L. Donoho, August 18, 2020 (sent for review July 22, 2020; reviewed by Helmut Boelsckei and Stéphane Mallat)

Article | Figures & SI | Info & Metrics | PDF

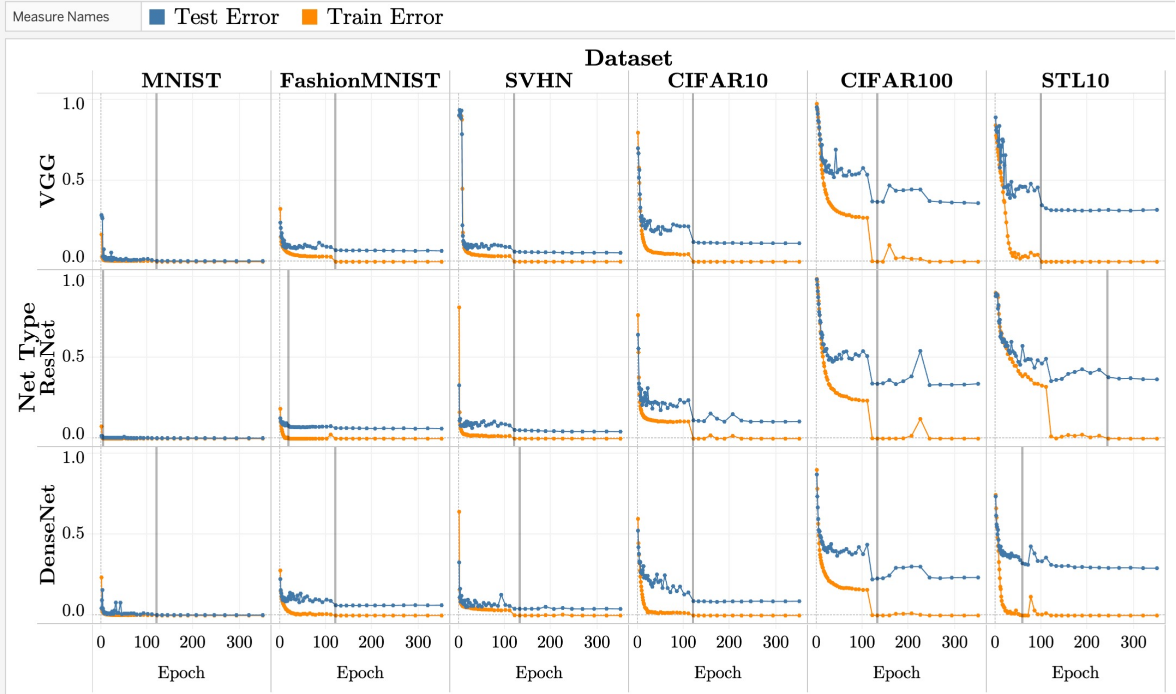
Significance

Modern deep neural networks for image classification have achieved superhuman performance. Yet, the complex details of trained networks have forced most practitioners and researchers to regard them as black boxes with little that could be understood. This paper considers in detail a now-standard training methodology: driving the cross-entropy loss to zero, continuing long after the classification error is already zero. Applying this methodology to an authoritative collection of standard deepnets and datasets, we observe the emergence of a simple and highly symmetric geometry of the deepnet features and of the deepnet classifier, and we document important benefits that the geometry conveys—thereby helping us understand an important component of the modern deep learning training paradigm.



Neural Collapse: An XYZ Story

- Original Goal: Can deep net performance be predicted?

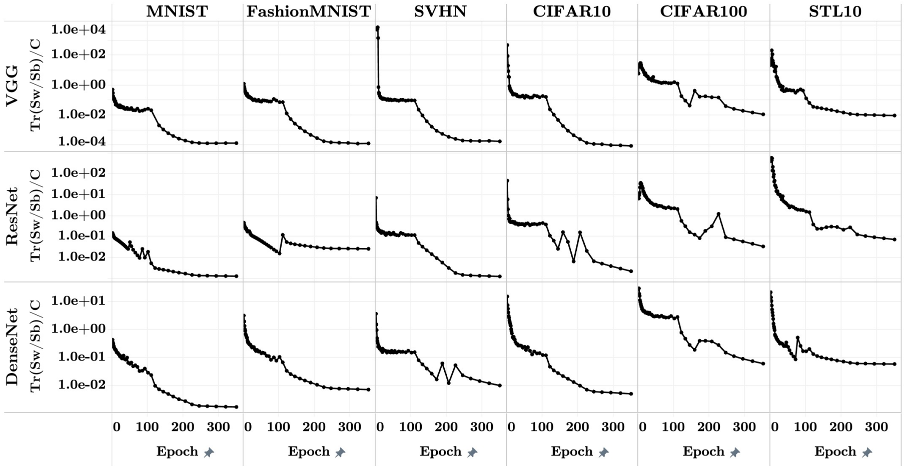


Neural Collapse: An XYZ Story

- Statistician’s Intuition: Bias-variance
    - Bias: How the class-means behave.
    - Variance: How spread out the data is around the class mean.

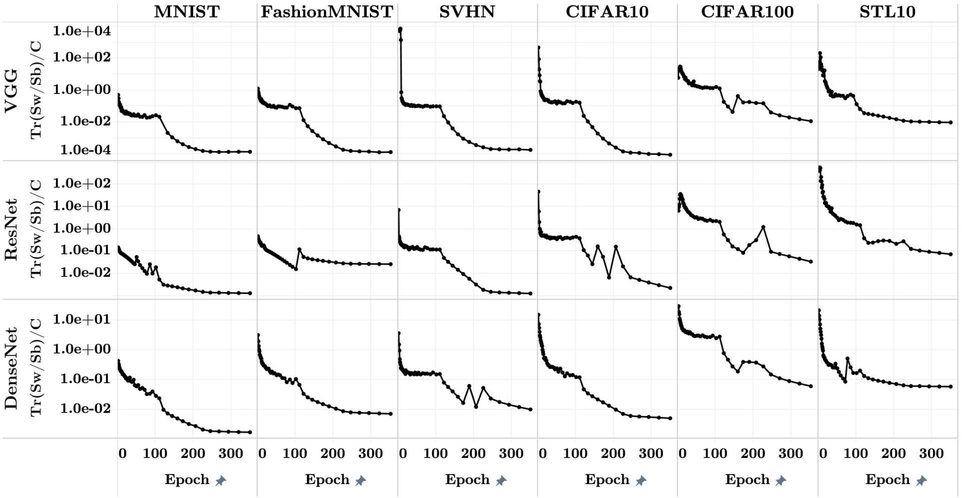
Neural Collapse: An XYZ Story

- Measurement: $\frac{1}{C} Tr\{\Sigma_B^{-1}\Sigma_W\}$
  - $\Sigma_B = \frac{1}{C}\sum_c(\mu_c - \mu_G)^T(\mu_c - \mu_G)$ captures structure of means (bias).
  - $\Sigma_W = \frac{1}{nC}\sum_c\sum_{i=1}^{n}(h_{ic} - \mu_c)^T(h_{ic} - \mu_c)$ captures structure of variance.



Neural Collapse: An XYZ Story

- Measurement: $\frac{1}{C} Tr\{\Sigma_B^{-1}\Sigma_W\}$
- Observation: Shrinking towards 0!
- Implication: Variance is shrinking compared to class means.

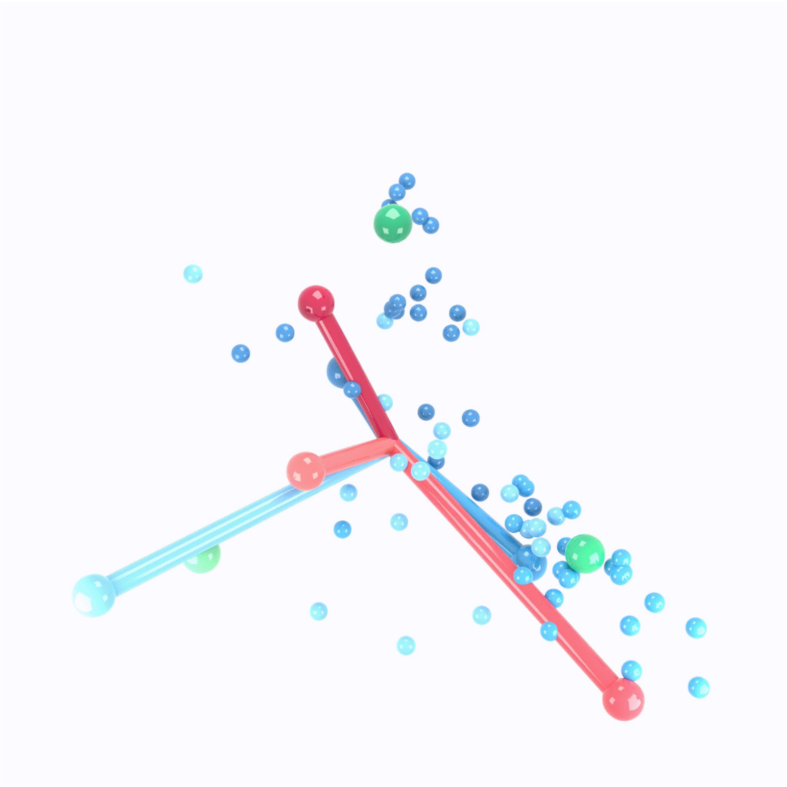


Neural Collapse: An XYZ Story

- Previous works have shown that for fixed last-layer activations, network classifiers converge to maximum-margin classifiers.

- If activations collapse to the same class-means, these classifiers converge to nearest-neighbor.

- The means themselves must be maximally distanced:
  An Equiangular Tight Frame!



Neural Collapse: An XYZ Story

- If ETF hypothesis holds, angles between any two class-means must be the same.
- Check this hypothesis with XYZ: It holds!

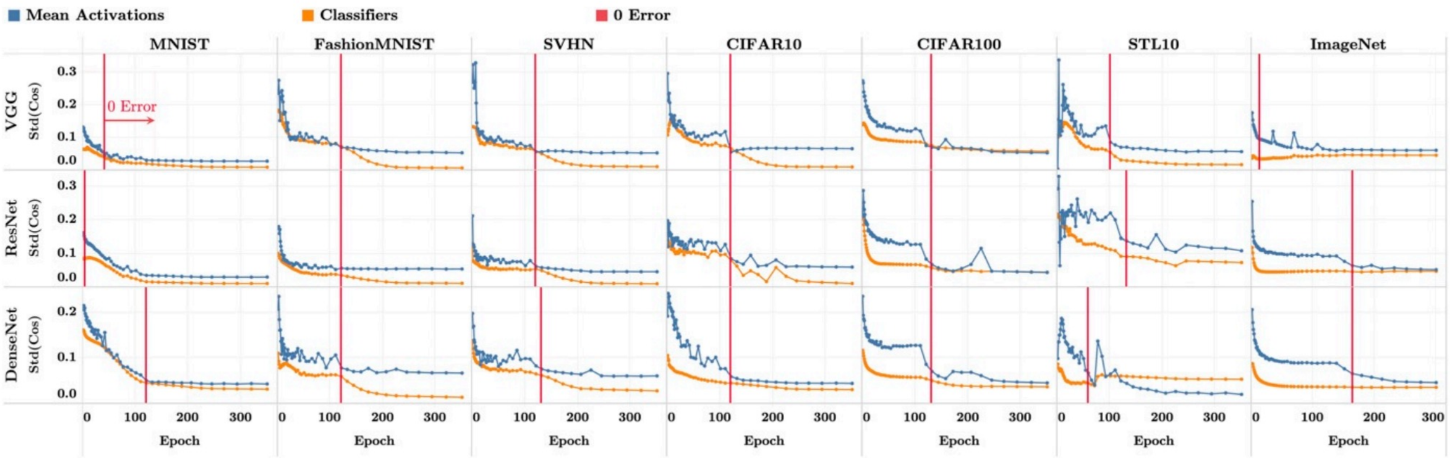


Fig. 3. Classifiers and train class means approach equiangularity. The formatting and technical details are as described in Section 3. In each array cell, the vertical axis shows the SD of the cosines between pairs of centered class means and classifiers across all distinct pairs of classes c and c'. Mathematically, denote $\cos_{\mu}(c, c') = \langle \mu_c - \mu_G, \mu_{c'} - \mu_G \rangle / (\|\mu_c - \mu_G\|_2 \|\mu_{c'} - \mu_G\|_2$ and $\cos_{w}(c, c') = \langle w_c, w_{c'} \rangle / (\|w_c\|_2 \|w_{c'}\|_2)$ where $\{w_c\}_{c=1}^{C}$, $\{\mu_c\}_{c=1}^{C}$, and μ_G are as in Fig. 2. We measure $\mathrm{Std}_{c,c' \neq c}(\cos_{\mu}(c, c'))$ (blue) and $\mathrm{Std}_{c,c' \neq c}(\cos_{w}(c, c'))$ (orange). As training progresses, the SDs of the cosines approach zero, indicating equiangularity.

Neural Collapse: An XYZ Story

- More XYZ experiments:
- Checking equinormness, nearest-neighbor behavior etc.
- Publish and share our findings.

RESEARCH ARTICLE

Prevalence of neural collapse during the terminal phase of deep learning training

Vardan Papyan, X. Y. Han, and David L. Donoho
+ See all authors and affiliations

PNAS October 6, 2020 117 (40) 24652-24663; first published September 21, 2020;
https://doi.org/10.1073/pnas.2015509117

Contributed by David L. Donoho, August 18, 2020 (sent for review July 22, 2020; reviewed by Helmut Boelscskei and Stéphane Mallat)

Article | Figures & SI | Info & Metrics | PDF

Significance

Modern deep neural networks for image classification have achieved superhuman performance. Yet, the complex details of trained networks have forced most practitioners and researchers to regard them as black boxes with little that could be understood. This paper considers in detail a now-standard training methodology: driving the cross-entropy loss to zero, continuing long after the classification error is already zero. Applying this methodology to an authoritative collection of standard deepnets and datasets, we observe the emergence of a simple and highly symmetric geometry of the deepnet features and of the deepnet classifier, and we document important benefits that the geometry conveys—thereby helping us understand an important component of the modern deep learning training paradigm.

Neural Collapse: An XYZ Story

- Multiple follow-up works since September 2020!

2. arXiv:2101.12699 [pdf, other] cs.LG cs.CV math.OC stat.ML
Layer-Peeled Model: Toward Understanding Well-Trained Deep Neural Networks
Authors: Cong Fang, Hangfeng He, Qi Long, Weijie J. Su
Abstract: ...on class-balanced datasets, we prove that any solution to this model forms a simplex equiangular tight frame, which in part explains the recently discovered phenomenon of **neural collapse** in deep learning training [PHD20]. Moreover, when moving to the imbalanced case, our analysis of the Layer-Peeled Model reveals a hit... ▽ More
Submitted 15 February, 2021; **v1** submitted 29 January, 2021; **originally announced** January 2021.

3. arXiv:2101.00072 [pdf, other] cs.LG stat.ML
Explicit regularization and implicit bias in deep network classifiers trained with the square loss
Authors: Tomaso Poggio, Qianli Liao
Abstract: ...-- is also possible in the no-BN and no-WD case. The theory yields several predictions, including the role of BN and weight decay, aspects of Papyan, Han and Donoho's **Neural Collapse** and the constraints induced by BN on the network weights. ▽ More
Submitted 31 December, 2020; **originally announced** January 2021.

4. arXiv:2012.08465 [pdf, other] cs.LG math.CA
Neural Collapse with Cross-Entropy Loss
Authors: Jianfeng Lu, Stefan Steinerberger
Abstract: ..., the global minimum is given by the simplex equiangular tight frame, which justifies the **neural collapse** behavior. We also prove that as $n \to \infty$ with fixed d, the minimizing points will distribute uniformly on the hypersphere and show a connection with the frame potential of Benedetto & Fickus. ▽ More
Submitted 18 January, 2021; **v1** submitted 15 December, 2020; **originally announced** December 2020.

5. arXiv:2011.11619 [pdf, other] cs.LG
Neural collapse with unconstrained features
Authors: Dustin G. Mixon, Hans Parshall, Jianzong Pi
Abstract: **Neural**... ▽ More
Submitted 23 November, 2020; **originally announced** November 2020.