Advanced topics and a practical case study

Dr. Eran Raviv

July 2022

Agenda for today

- Recap from last time
- GAN
- Advanced architectures
- Case study

“What I cannot create, I do not understand”

Richard Feynman

Generative adversarial networks

Snapshot from 2020..

Adversarial nets

- Original idea.
- Resembles real life situation.
- *Relatively* easy to understand and code.

Adversarial nets

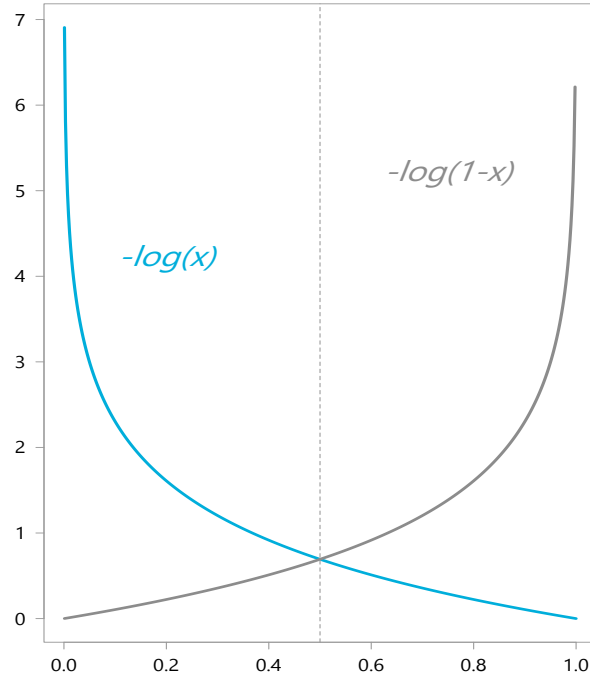
Adverserial $\Rightarrow G$

Discriminator $\Rightarrow D$

$$\min_G \max_D V(D,G) = \mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}\ (\boldsymbol{x})}[\log D(\boldsymbol{x})] + \mathbb{E}_{\boldsymbol{z}\sim p_{\boldsymbol{z}}(\boldsymbol{z})}[\log(1 - D(G(\boldsymbol{z})))]$$

$V(G, D)$

- Minimax game
- Equilibrium: 50/50 chance for true of fake.
- Put differently, $p_g = p_{data}$



Training GANs

Algorithm 1: Minibatch stochastic gradient descent training of generative adversarial nets. The number of steps to apply to the discriminator, k, is a hyperparameter. We used $k = 1$ was used in the original paper.

for number of training iterations **do**
  for k steps **do**

- Sample minibatch of m noise samples $\left\{\boldsymbol{z}^{(1)}, \ldots, \boldsymbol{z}^{(m)}\right\}$ from noise prior $p_g(\boldsymbol{z})$
- Sample minibatch of m examples $\left\{\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(m)}\right\}$ from data generating distribution $p_{\text{data}}(\boldsymbol{x})$
- Update the discriminator by ascending its stochastic gradient:

$$\nabla_{\theta_d} \frac{1}{m} \sum_{i=1}^{m} \left[\log D\left(\boldsymbol{x}^{(i)}\right) + \log\left(1 - D\left(G\left(\boldsymbol{z}^{(i)}\right)\right)\right)\right]$$

end for

- Sample minibatch of m noise samples $\left\{\boldsymbol{z}^{(1)}, \ldots, \boldsymbol{z}^{(m)}\right\}$ from noise prior $p_g(\boldsymbol{z})$
- Update the generator by descending its stochastic gradient:

$$\nabla_{\theta_g} \frac{1}{m} \sum_{i=1}^{m} \log\left(1 - D\left(G\left(\boldsymbol{z}^{(i)}\right)\right)\right)$$

Training GANs

$$\min_G \max_D V(D,G) = \mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}(\boldsymbol{x})}[\log D(\boldsymbol{x})] + \mathbb{E}_{\boldsymbol{z}\sim p_{\boldsymbol{z}}(\boldsymbol{z})}[\log(1 - D(G(\boldsymbol{z})))]$$

- Remember, we are after finding $\min_G \max_D V(D,G)$.
- *Discriminator* wants $D(x)$ close to one (so log is zero, which means always know when something comes from real data).
- *Discriminator* wants $D(G(z))$ close to zero $\Rightarrow$ $1 - D(G(z))$ close to one. Best *Discriminator* can do is hope for zero (anything below decrease the number).
- *Generator* wants exactly the other way. But has influence only on generated images $\Rightarrow$ $G(z)$ so it cares only about $D(G(z))$.

Training GANS

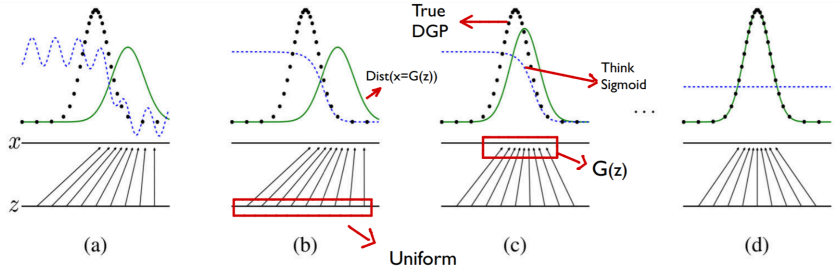


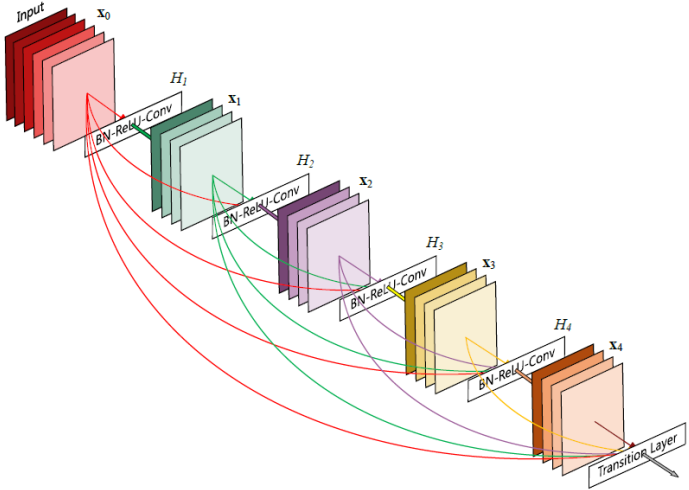
Figure 1: Generative adversarial nets are trained by simultaneously updating the **d**iscriminative distribution (D, blue, dashed line) so that it discriminates between samples from the data generating distribution (black, dotted line) $p_{\boldsymbol{x}}$ from those of the **g**enerative distribution p_g (G) (green, solid line). The lower horizontal line is the domain from which $\boldsymbol{z}$ is sampled, in this case uniformly. The horizontal line above is part of the domain of $\boldsymbol{x}$. The upward arrows show how the mapping $\boldsymbol{x} = G(\boldsymbol{z})$ imposes the non-uniform distribution p_g on transformed samples. G contracts in regions of high density and expands in regions of low density of p_g. (a) Consider an adversarial pair near convergence: p_g is similar to p_{data} and D is a partially accurate classifier. (b) In the inner loop of the algorithm D is trained to discriminate samples from data, converging to $D^*(\boldsymbol{x}) = \frac{p_{\text{data}}(\boldsymbol{x})}{p_{\text{data}}(\boldsymbol{x}) + p_g(\boldsymbol{x})}$. (c) After an update to G, gradient of D has guided $G(\boldsymbol{z})$ to flow to regions that are more likely to be classified as data. (d) After several steps of training, if G and D have enough capacity, they will reach a point at which both cannot improve because $p_g = p_{\text{data}}$. The discriminator is unable to differentiate between the two distributions, i.e. $D(\boldsymbol{x}) = \frac{1}{2}$.

Wtih enough computational power..

Unreal

Three progressive ideas

Densely connected convolutional networks



Huang et al. 2017

Densely connected convolutional networks

You are now familiar with all the concepts you need to understand more progressive suggestions.

- A fairly simple yet powerful idea.
- More or less an extension of ResNets. Instead of

$$\mathbf{x}_\ell = H_\ell\left(\mathbf{x}_{\ell-1}\right) + \mathbf{x}_{\ell-1}$$

  We use

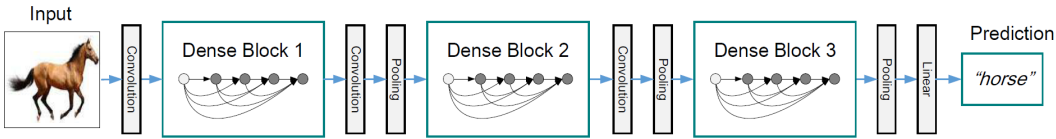
$$\mathbf{x}_\ell = H_\ell\left(\left[\mathbf{x}_0, \mathbf{x}_1, \ldots, \mathbf{x}_{\ell-1}\right]\right).$$

- Each layer has direct access to the gradients of the loss function and the original input signal.

Densely connected convolutional networks

- H_ℓ does three operations for the inputs that are carried from layer $\ell - 1$:
  1. Batch normalization.
  2. Relu.
  3. 3×3 convolution.
- If k_0 is the original input, and we constantly send forward the intermediate transformations, then the ℓ^{th} layer has $k_0 + k \times (\ell - 1)$. Hence the *hyper*parameter k is referred to as the growth rate of the network.
- k need not be very large ($k = 12$ is enough).

Densely connected convolutional networks



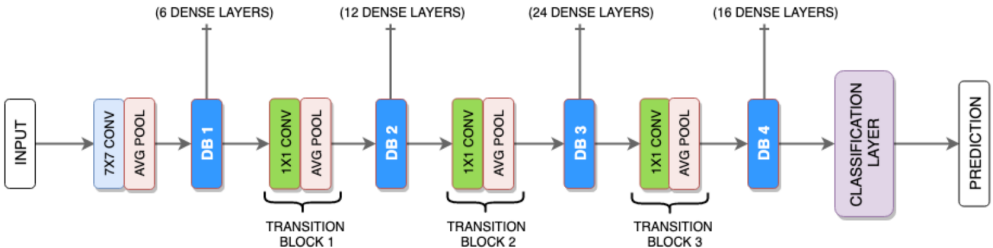
- To keep the size manageable as the network becomes deeper we add *transition layers.*

  "The transition layers used in our experiments consist of a batch normalization layer and an 1×1 *convolutional layer followed by a* 2×2 *average pooling layer."*

- This is a prime example for the duo role that convolution plays in deep learning models.

Correction: last layer is softmax, and not linear as in the image.

Densely connected convolutional networks



"Instead of drawing representational power from extremely deep or wide architectures, DenseNets exploit the potential of the network through feature reuse.."

- Comparable results- but with less parameters (e.g. "only" 800K).

Densely connected convolutional networks - what's the story?

"The global state, once written, can be accessed from everywhere within the network and, unlike in traditional network architectures, there is no need to replicate it from layer to layer."

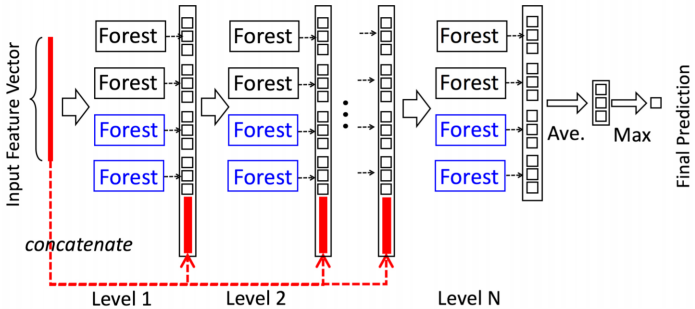
Deep forest

Deep forest

- Zhou and Feng 2017
- Good story:

Deep neural networks (e.g., convolutional neural networks)	gcForest
Type of activation functions: Sigmoid, ReLU, tanh, linear, etc. Architecture configurations: **No. Hidden layers**: ? **No. Nodes in hidden layer**: ? **No. Feature maps**: ? **Kernel size**: ? Optimization configurations: **Learning rate**: ? Dropout: {0.25/0.50} **Momentum**: ? **L1/L2 weight regularization penalty**: ? Weight initialization: Uniform, glorot_normal, glorot_uni, etc. Batch size: {32/64/128}	Type of forests: Completely-random tree forest, random forest, etc. Forest in multi-grained scanning: **No. Forests**: {2} **No. Trees in each forest**: {500} Tree growth: till pure leaf, or reach depth 100 **Sliding window size**: $\{\lfloor d/16 \rfloor, \lfloor d/8 \rfloor, \lfloor d/4 \rfloor\}$ Forest in cascade: **No. Forests**: {8} **No. Trees in each forest**: {500} Tree growth: till pure leaf

Deep forest - visually



- Black is standard RF. Blue is RF variant.
- Each forest is like a hidden unit.
- What makes it a deep learning model?

Deep forest

- Competitive results on various data.
- Training is less data-hungry.
- Trains faster.
- Fewer decisions to make  .

AdaNet

Adaptive Structural Learning of Artificial Neural Networks - AdaNet

- Cortes et al. 2017

  "ADANET adaptively grows the structure of a neural network, balancing model complexity with empirical risk minimization."

- A word on complexity. Rademacher complexity measures how well the class of functions correlates with randomly generated labels. The richer the class of functions, the better the chance of finding a function within the family of functions that correlates with a given random sequence, and hence the larger the complexity.

Rademacher complexity

- Denote r_j as the Rademacher complexity of a the class of functions $\mathcal{H}_{k_j}$ for $j \in N$.
- The empirical Rademacher complexity of a hypothesis set $\mathcal{G}$ for a sample s is denoted by $\widehat{\mathfrak{R}}_S(\mathcal{G})$ and defined as follows:

$$\widehat{\mathfrak{R}}_S(\mathcal{G}) = \frac{1}{m}\mathbb{E}_\psi \left[\sup_{h \in \mathcal{G}} \sum_{i=1}^{m} \psi_i h(x_i) \right],$$

where $\psi = (\psi_1, \ldots, \psi_m)$ are independently uniformly distributed random variables over $\{-1, +1\}$ (Rademacher distribution).

Rademacher complexity - (wikipedia) example

Say

$$A = \{(1,1),(1,2)\} \subset \mathbb{R}^2$$

Then

$$\begin{aligned}\text{Rad}(A) &= \frac{1}{2} \cdot \left(\frac{1}{4} \cdot \max(1+1, 1+2) + \frac{1}{4} \cdot \max(1-1, 1-2) + \right. \\ &\left. \frac{1}{4} \cdot \max(-1+1, -1+2) + \frac{1}{4} \cdot \max(-1-1, -1-2) \right) \\ &= \frac{1}{8}(3+0+1-2) \\ &= \frac{1}{4}\end{aligned}$$

For more details see here

AdaNet

- A boosting-style algorithm that applies (block) coordinate descent to:

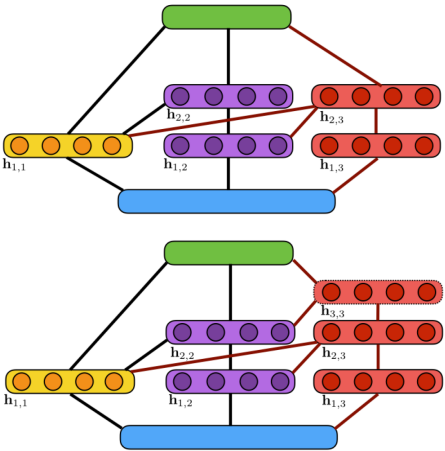
$$F(\mathbf{w}) = \frac{1}{m}\sum_{i=1}^{m}\Phi\left(1 - y_i\sum_{j=1}^{N}w_j h_j\right) + \sum_{j=1}^{N}(\lambda r_j + \beta)\left|w_j\right|,$$

  with hyperparameters $\lambda \geq 0$, and $\beta \geq 0$.

- In words, we want to penalize the usual objective function (empirical error). The more complex the model, the larger the penalty.

AdaNet

- Think information criteria simply.



AdaNet - pointers

- ".. the search is particularly challenging."
- These kind of ideas falls under the *Neural architecture search (NAS)* strand of research.

Deep learning frameworks

Why do we need frameworks for?

- What are the options really?
- **Scalability, operational efficiency.**
- Tried and true.

mxnet

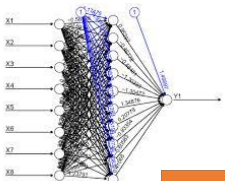
Microsoft CNTK

DL4J

PyTorch

TensorFlow

Caffe



H2O.ai

Keras

Caffe2

Theano

Blunders

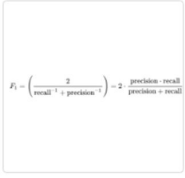
Blunders (1)

Eran Raviv, PhD • 6:18 PM

Also, in the result slide, the numbers don't add up (the way F1 is supposed to be calculated (i.e. with beta= 1))

So perhaps it is F_beta with beta not 1

otherwise maybe I am not doing something right



[redacted] • 6:57 PM

You're right, it doesn't add up. They may have calculated using TP/FN/FP. I've asked [redacted] for the actual code. Will share it with you soon.

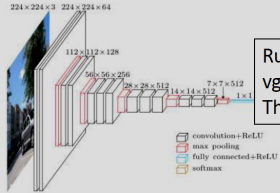
[redacted] • 7:49 PM

Hey, It was calculated using the pyrogue package. Particularly this function

- https://github.com/bheinzerling/pyrouge/bl

Blunders (2)

Take five Brad Pitt pictures



Run them trough the pre-trained vgg16 and extract feature vectors.
This is a **5 by 25088** matrix

The brad Pit Index

Take other images, run them through the VGG16

Calculate the **distances** with the five Brad Pitt pictures and average:


0.771195


0.802654


0.714752


0.792587


0.8291976


0.8096944


0.665990


0.9737212

Blunders (3)

Proof. Omitted. ⟵ classic □

Lemma 0.1. *Let $\mathcal{C}$ be a set of the construction.*

Let $\mathcal{C}$ be a gerber covering. Let $\mathcal{F}$ be a quasi-coherent sheaves of $\mathcal{O}$-modules. We have to show that

$$\mathcal{O}_{\mathcal{O}_X} = \mathcal{O}_X(\mathcal{L})$$

.

Proof. This is an algebraic space with the composition of sheaves $\mathcal{F}$ on $X_{\acute{e}tale}$ we have

$$\mathcal{O}_X(\mathcal{F}) = \{morph_1 \times_{\mathcal{O}_X} (\mathcal{G}, \mathcal{F})\}$$

where $\mathcal{G}$ defines an isomorphism $\mathcal{F} \to \mathcal{F}$ of $\mathcal{O}$-modules. □

Lemma 0.2. *This is an integer $\mathcal{Z}$ is injective.*

Proof. See Spaces, Lemma **??**. □

Lemma 0.3. *Let S be a scheme. Let X be a scheme and X is an affine open covering. Let $\mathcal{U} \subset \mathcal{X}$ be a canonical and locally of finite type. Let X be a scheme. Let X be a scheme which is equal to the formal complex.*

The following to the construction of the lemma follows.

Let X be a scheme. Let X be a scheme covering. Let

$$b : X \to Y' \to Y \to Y \to Y' \times_X Y \to X.$$

be a morphism of algebraic spaces over S and Y.

Proof. Let X be a nonzero scheme of X. Let X be an algebraic space. Let $\mathcal{F}$ be a quasi-coherent sheaf of $\mathcal{O}_X$-modules. The following are equivalent

(1) $\mathcal{F}$ is an algebraic space over S.
(2) If X is an affine open covering.

Consider a common structure on X and X the functor $\mathcal{O}_X(U)$ which is locally of finite type. □

This since $\mathcal{F} \in \mathcal{F}$ and $x \in \mathcal{G}$ the diagram

$$\begin{array}{ccc} \mathcal{S} & & \\ \downarrow & & \\ \xi & \to & \mathcal{O}_{X'} \\ & \searrow & \uparrow \\ \text{gor}_s & & \\ \\ = \alpha' & \to & \\ \updownarrow & & \\ = \alpha' & \to & \alpha \end{array} \qquad \begin{array}{c} X \\ \downarrow \\ \mathrm{d}(\mathcal{O}_{X_f}, \mathcal{G}) \end{array}$$

$$\mathrm{Spec}(K_{\mathfrak{C}}) \qquad \mathrm{Mor}_{Sets}$$

is a limit. Then $\mathcal{G}$ is a finite type and assume S is a flat and $\mathcal{F}$ and $\mathcal{G}$ is a finite type f_*. This is of finite type diagrams, and

- the composition of $\mathcal{G}$ is a regular sequence,
- $\mathcal{O}_{X'}$ is a sheaf of rings.

□

Proof. We have see that $X = \mathrm{Spec}(R)$ and $\mathcal{F}$ is a finite type representable by algebraic space. The property $\mathcal{F}$ is a finite morphism of algebraic stacks. Then the cohomology of X is an open neighbourhood of U. □

Proof. This is clear that $\mathcal{G}$ is a finite presentation, see Lemmas **??**.
A *reduced above* we conclude that U is an open covering of $\mathcal{C}$. The functor $\mathcal{F}$ is a "field

$$\mathcal{O}_{X,x} \longrightarrow \mathcal{F}_{\overline{x}} \quad \text{-1}(\mathcal{O}_{X_{\acute{e}tale}}) \longrightarrow \mathcal{O}^{-1}_{X_i}\mathcal{O}_{X_i}(\mathcal{O}^{\mathcal{C}}_{X_{\overline{x}}})$$

is an isomorphism of covering of $\mathcal{O}_{X_i}$. If $\mathcal{F}$ is the unique element of $\mathcal{F}$ such that X is an isomorphism.
The property $\mathcal{F}$ is a disjoint union of Proposition **??** and we can filtered set of presentations of a scheme $\mathcal{O}_X$-algebra with $\mathcal{F}$ are opens of finite type over S.
If $\mathcal{F}$ is a scheme theoretic image points. □

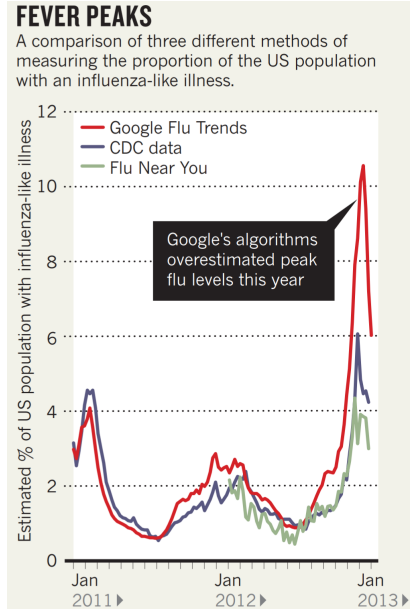
If $\mathcal{F}$ is a finite direct sum $\mathcal{O}_{X_i}$ is a closed immersion, see Lemma **??**. This is a sequence of $\mathcal{F}$ is a similar morphism.

Blunders (4)

Blunders (5)

When Google got flu wrong

"You need to be constantly adapting these models, they don't work in a vacuum … You need to recalibrate them every year."



Discussion

- Data and out-of-sample overfitting.
- Usable models.
- Statistical modeling: The two cultures (Breiman et al. 2001)

Common mistakes found in the wild

Frankenstein data sets

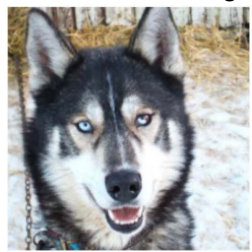
Frankenstein data sets (continued)

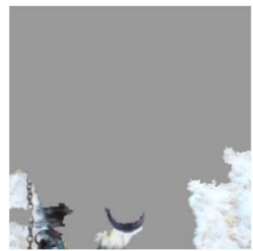
- Using CT of children as control
- Using duplicates
- Using transformed data with no ability to revert to the original

Attention to data

Supine, or standing position?

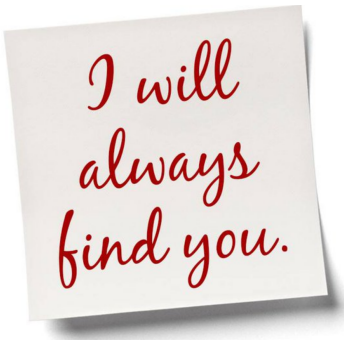
Huski wrongly classified as a wolf, but why?

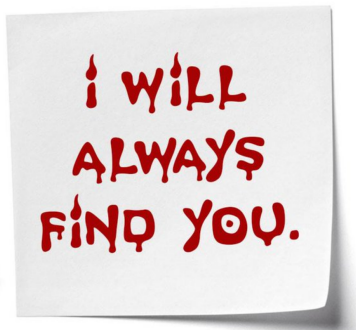




Hard core biases

Some fonts  were actually relied on when making predictions. Font used by high caseloads hospitals turns out to be active contributor to the prediction of covid risk.





Labeling issues (garbage in –> garbage out)

- Expensive labeling process (e.g. PCR rather than radiologists)
- Noisy ground truth (e.g. sentiment, or summarization)
- Too little time, and too little energy for "statistical niceties"

Incentives...

"Life's not knights on horseback. It's a number on a piece of paper" (Succession, season 3 episode 8)

"It's shocking. I went into it with some worries, but this exceeded my fears." (Laure wynants, one of the many authors of *Prediction models for diagnosis and prognosis of covid-19: systematic review and critical appraisal*)

- No motivation to share
- No motivation to expose yourself to external validation
- No motivation to care for replicability
- No motivation to self-criticize (robustness tests)
- On the other hand, high motivation to publish!
- Our gate keepers are working hard

The reality is:

"Until we buy into the idea that we need to sort out the unsexy problems before the sexy ones, we're doomed to repeat the same mistakes"

Bilal Mateen, the Alan Turing Institute

References

Besse, P., Castets-Renard, C., Garivier, A., and Loubes, J.-M. (2019). Can everyday ai be ethical? machine learning algorithm fairness. Machine Learning Algorithm Fairness (May 20, 2018). Statistiques et Société, 6(3).

Breiman, L. et al. (2001). Statistical modeling: The two cultures (with comments and a rejoinder by the author). Statistical science, 16(3):199–231.

Cortes, C., Gonzalvo, X., Kuznetsov, V., Mohri, M., and Yang, S. (2017). Adanet: Adaptive structural learning of artificial neural networks. In Proceedings of the 34th International Conference on Machine Learning-Volume 70, pages 874–883. JMLR. org.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. (2014). Generative adversarial nets. In Advances in neural information processing systems, pages 2672–2680.

Huang, G., Liu, Z., Van Der Maaten, L., and Weinberger, K. Q. (2017). Densely connected convolutional networks. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 4700–4708.

Zhou, Z.-H. and Feng, J. (2017). Deep forest. arXiv preprint arXiv:1702.08835.