Notes from ICML 2018

My first complete conference attendance! I tried to get a wide cross-section of important developments in machine learning. Each 1-2 hour time block consisted of a number of sessions grouped by topic, of which I would necessarily choose one. Under these session headings, each paragraph typically corresponds to one accepted paper. Please don't be bothered if you find that I totally missed the point of your work, or neglected to mention it under its session heading. My background knowledge isn't equal across all areas, and it's very hard to focus for so many hours with jetlag!

Some talks/works I recommend are Toward a Theoretical Understanding of Deep Learning, Intelligence by the Kilowatt-Hour, and the Google Magenta talk. For our applications at Mythic, start with Noise2Noise, and the talks on neural architecture search.

7/10 Tutorial: learning with temporal point processes

Used to model discrete events in continuous time: disease dynamics, financial trading, social media activity, etc.

Process characterized by intensity function:
lambda*(t) = 1/dt Pr(event happening between t and t+dt)

"Infection" events (e.g., getting a flu strain, having seen a movie) modeled as pair (u_i, t_i) of user and time.

Examples

Admittedly, most of the problems presented seemed like toy models, much simpler than reality.

lambda*_u(t)= (1 - N_u(t)) sum_v b_vu sum kappa (t - t_i) where
N_u(t) = 1 if user is already infected, 0 otherwise
b_vu is influence of user v on user u
t_i is when user v got infected??

lambda*_u(t)= mu_u sum_v sum_e b_vu sum kappa (t - t_i)
mu_u are events of user u's own initiatives
e are events where user v sends a message to user u at time t_i

7/10 Tutorial: toward theoretical understanding of DL

Goal of theory is to develop theorems that sort through or support concepts and intuitions that lead to new ideas.

Trying to understand:

- When can we find decent solutions? Highly nonconvex, worst-case NP-hard…
- Why do nets generalize well?
- Role of depth
- Unsupervised learning and GANs
- Simpler methods to replace deep learning?

Optimization

In high dimensions there are a lot of different directions (exp(d) many whose pairwise angle is > 60 degrees), but not *too* many (at most exp(d/epsilon) covering all angles to within epsilon)

No clean characterization of loss (we can't explain mathematically what makes something a cat), so treat it as a blackbox.

If second derivative is bounded, can take small enough gradient steps to guarantee reaching a stationary point. But what about saddle points? If noise is added to the gradient descent (as in SGD), they're easily avoided!

In poly(d/epsilon) points, can reach an approximate local min, in the sense that |gradient| < epsilon and hessian > -epsilon I

Approximate 2nd order methods can be faster in theory, but so far don't seem to produce better nets.

Overparametrization helps: say you generate data by feeding random vectors into a 2-layer net. It's hard to train a new set of the same size on the same data; it helps to make the hidden layer wider than in the original net!

Generalization

It was once believed that deep nets are regularized enough to remove the "excess" capacity. However, recent experiments show the excess capacity is still there. So why do deep nets generalize well?

Effective capacity is roughly log (# distinct a priori models).

Analogy: If a system has 2^k states, we say it has k bits.
Precise notions include VC dimension and Rademacher complexity.

Theorem: Test loss - training loss <= sqrt(effective capacity / # training samples) = sqrt(N / m)

It was once believed that SGD prefers “flat” minima over “sharp” ones. Makes intuitive sense, but hard to make precise, and recently shown false for certain notions of “sharp”.

VGG higher layers seem to reject/attenuate noise injected at lower levels. (try to draw similar plots for Mythic noise as a function of layer number?)
What does it mean mathematically? If layer transforms x + eta to M(x + eta), and the noise portion M*eta is supposed to be small, it suggests that the singular values of x must be very concentrated. The relevant degrees are freedom in x will correspond to the largest singular values. Random noise will mostly fall into the less relevant dimensions.

Future work: need arguments that involve more properties of the training algorithm or data distribution.

Role of Depth

Can increase representation. Even in cases where the representation power is obviously equivalent (like composing multiple linear operations), it may accelerate the optimization landscape. Provide a sort of “momentum”.

Unsupervised Learning

Manifold hypothesis: dataset is supported on a low-dimensional set, so may be useful to train a net (e.g. autoencoder) to do dimensionality reduction. Using the compact representation, downstream tasks might need fewer training data.

Some theory was presented to argue that equilibria of GAN objective may suffer from mode-collapse. But in practice? Use birthday paradox test to estimate support size of different GANs. Bigger discriminators seem to help, as the theory predicts.

Parting Words

Best theory will emerge from engaging with real data and real deep net training. .Nonconvexity and attendant complexity seems to make armchair theory less fruitful.

7/10 Tutorial: Defining & Designing Fair Algorithms

Note to self to see the recording. There’s a lot of buzz around fairness criteria this year.

7/10 Tutorial: Variational Bayes and Beyond

If you're interested in the math behind variational inference (e.g. VAEs) and whether they're the right thing to do, check out the recording of this tutorial!

7/11 Invited Talk: AI and Security

As AI controls more systems, attackers will have higher incentives to target them. Very high-level overview of the field: adversarial examples, software vulnerabilities, verification techniques, etc. Developers ought to differentiate between regression testing (on noisy training data) vs security testing (on intentionally poisoned training data).

7/11 Best Paper: obfuscated gradients give a false sense of security: circumventing defences to adversarial examples

Adversarial examples are interesting not only for security reasons, but also because they present cases that humans find easier than machines.

How to construct adversarial loss? Use gradient descent to find smallest change in the input with the largest change in the network's response. Obfuscated gradient defences attempt to mess up the gradient landscape by inserting less smooth functions, but can be defeated by smooth approximation of the difficult functions.

13 total defence papers at ICLR'18, most are broken. As a community, we need to change how we evaluate defences: instead of testing it against existing attacks, we need to act like an attacker (who has read our paper!), trying hard to defeat the new defence we want to propose. Rather than adding complexity and hoping this trips up attackers (a fragile approach), we should strive for simplicity and understandability. Make sure you're able to break the state-of-the-art defences before working on new defences. As a community, we should do strong re-evaluations of existing defence methods.

Threat model: assumptions on what the adversary can do.

7/11 Session 1: Representation Learning

The session started off with two very cool papers that use hyperbolic geometry to learn hierarchical latent structure. The space can be modeled as a unit ball, but with a metric which

makes distances and volumes grow to infinity as one approaches the boundary. This provides a convenient continuous space in which to embed tree structures, with the root(s) at the center, and sufficient space near the boundaries for an exponentially large number of leaf nodes. They develop a Riemannian gradient optimization technique. Blog: hyperbolicdeeplearning.com

Instead of one-hot vectors, propose the more compact “K-D” code: D integers, each ranging from 1 to K. If there are N classes, we can have a lot of redundancy (K^D >> N) while still preserving O(logN) size for the code. To train, use the discrete code during forward pass, and use tempering softmax with an annealed temperature during backward pass.

7/11 Session 2A: Other Applications

2nd talk of the session: medical treatment can be viewed as a discrete RL policy. Online policy can be costly, dangerous or unethical. We can’t observe the counterfactual outcome, e.g., what would happen had we gone with a different treatment than what was actually used.
We can do off-policy learning by using important sampling on an experienced physician’s past history, to evaluate the distribution given by the new policy. However, empirical risk minimization may not generalize well enough. This leads to some discussion of bounds and generalization methods. Future work (before this can be applied to real patients) is to generalize the method to supervised settings and to datasets without propensity work.

2D dataset: a bunch of students get scores on a bunch of questions. Want to identify the latent factors, such as difficulty, that account for these scores, and fill in missing data. The Rasch model is fairly powerful, and currently used in computerized adaptive testing. This work proposes a new error estimator for the Rasch model parameters, and performs a rigorous analysis of it. There was a bit of discussion from the audience on whether these models could be misused in high-stakes applications like admitting students or appointing judges.

Want to use RL to control a camera to track some object. They built an end-to-end prediction network, from raw visual frame to camera movement action, trained with the A3C algorithm.

7/11 Session 2B: Neural Network Architectures

The first talk in this session combines features of traditional adaptive control theory (like linear proportional feedback) and biological neurons (whose activations spike at sparse moments in time, and have recurrence connections with delay). I missed some important details, but this project looks very cool.

PredRNN++, a recurrent video prediction architecture that includes dual temporal and spatial memories, and highway connections. Tested on the Moving MNIST and KTH datasets, and worked with Chinese government to better weather forecasting than their old system!

Image restoration techniques have typically focused on either new architectures or new loss functions (e.g., adversarial, perceptual). However, standard convolutional architectures can outperform newer methods if you optimize its hyperparameters. To do an evolutionary optimization, represent CAE architectures by directed acyclic graphs with integer parameters for things like layer size.
Observations about winning models: skip connections at the first layer, more layers with large (5x5) filters.

Dual learning is a framework that leverages the symmetric (primal-dual) structure of AI tasks to obtain effective feedback or regularization signals to enhance training/learning effectiveness. For example, learning the mappings X -> Y and Y -> X can be composed as autoencoders X -> Y -> X or Y -> X -> Y. In this example, the same two networks can be combined in autoencoding in X as well as in Y.

7/11 Session 3A: Generative Models

Theoretical analysis of when GANs converge. Practical recommendations:

- Use alternating instead of simultaneous gradient descent
- Don't use momentum
- Use regularization
- Simple zero-centered gradient penalties for the discriminator
- Progressively growing architectures might not be important with good regularization

GAN variational objectives include information-theoretic divergences (like KL), integral probability metrics (like Wasserstein), and kernel discrepancy metrics. Chi^2-GAN has a bunch of equivalent formulations.

The last talk in this session gives a framework to measure diversity in GANs. They found that training on generated data yields poor results, which suggests that GANs may *not* be suitable for data augmentation.

7/11 Session 3B: Society Impacts of Machine Learning

The best paper talk examines different fairness criteria, all of which aim to protect "protected groups" from discrimination or harmful outcomes. One may maximize utility subject to the constraint of this criterion. The "demographic parity" criterion may be too strict; for example, giving loans equally to all groups can cause the disadvantaged group to receive loans they won't be able to pay back, harming their credit score. "Equal opportunity" is a milder criterion, only requiring an equal accept rate among members who are true positives (e.g., would repay their loans).

Many machine learning systems function poorly on minority groups, causing less retention, which in term biases the poison data, completing a vicious cycle. Rawlsian criterion is to minimize the expected loss to the least fortunate group. Since the loss is averaged over *groups*, under-represented groups are weighted properly and don't lead to a vicious cycle. Future work: tradeoff between outlier resistance and fairness.

Data points (s_i, u_i, y_i) consist of sensitive ("protected") attributes s_i, non-sensitive attributes u, and target value y_i. Can consider fairness as a hard or soft criterion of non-dependence on s_i. In a linear model y = sTa + uTb, the best estimator that depends only on u is y^ = uTb.

How to sample diverse images? Determinantal Point Process (DPP): choose vectors which make a large area (i.e., matrix determinant) together. To pick k_i elements from group G_i, repeatedly pick some i for which there are still less than k_i elements, and orthogonally project out the chosen fector from all remaining vectors.

7/12 Invited Talk: Intelligence per kilowatt-hour

Free Energy (aka "useful" energy that can be converted into work) = Energy - Entropy

Black holes are thermodynamic objects, with information content proportional to surface area. The holographic principle says this is the densing possible information packing.

According to E.T. Jaynes, free energy is a subjective quantity, because entropy is a degree of ignorance we have about the microscopic degrees of freedom of a system.

Regularization/generation by minimizing description length of a dataset
= L(Data | Model) + L(Model)
The first term is O(#data), the second is O(1). Hence, it's justified to use a more complex model only when the dataset is large.

Hinton's variational bayes can be interpreted as a version of Free Energy = Energy - Entropy.

Contrast between variational inference and MCMC sampling, where the goal is to estimate a quantity like E_{p(theta|X)} f(theta):

- In variational inference, you find the closest distribution q(theta) within some "nice" set in which the computation is easy. This is deterministic (low variance) but high bias.
- In MCMC, you sample according to p(theta|X); we only need the ability to evaluate a potentially unnormalized p at arbitrary points. This is unbiased in the limit, but high variance and it's hard to mix modes.

Langevin Dynamics: by adding a noise term to SGD, it's possible to, instead of approaching maximum likelihood, to asymptotically sample from the correct posterior distribution. Smaller epsilon decreases variance and increases bias (because it takes longer to converge??)

The thermal ceiling: as our models grow, it becomes challenging for the thermal cost to stay under the economic benefit. Edge devices in particular have power constraints. AI algorithms will be measured by the amount of intelligence per kilowatt-hour.

Can we use Bayesian statistics to reduce energy in models? Noise introduced to weights propagates to activations. Somehow (I didn't catch the details), this can inform how we quantize weights and prune activation nodes.

Many referenced works may be worth looking at: Probabilistic Binary Networks, DIfferentiable Quantization, Spiking Neural Networks. During surveillance, if we fire only 0.1% of the time, we might save a lot of energy.

7/12 Session 1: Structured Prediction

In structured prediction, rather than computing a simple classification, we output something more complex such as a parse tree, or a labelling of all the image regions corresponding to objects, or a set of vertices in R^D defining a polytope. While in classification we may output logits corresponding to the discrete classes, in polytope prediction we want to output a more general *potential* over the space R^D.
Dynamic Programming Bellman Equation: v_i(theta) = max_j theta_ij + v_j(theta)
Replace the max and argmax with smooth versions (twice-differentiable with non-zero Hessian) using a function Omega(x). Specific choices of Omega lead to the softmax and sparsemax operators as special cases. Will have to read paper to understand this better. The authors provide a PyTorch package called didyprog.

The next (3rd) talk takes reviews with known sentence-level graph structure. It combines sentence level sentiment prediction to predict the score in the review.

7/12 Session 2A: Supervised Learning

Want a classifier that's robust to small changes in the data distribution. Distributionally Robust Supervised Learning (DRSL) optimizes the loss according to the worst-case distribution shift within an f-divergence ball. However, it turns out DRSL produces the same kind of normal classifiers that are not robust, just with increased sensitivity to outliers. This is because the f-divergence ball is too flexible. To get robustness, we have to impose structural assumptions on the distribution shift. This can be applied to fairness in ML (see ML talks), in which case the structural assumptions would be related to the protected demographics.

Say we want predictions to be monotonic wrt k'th feature: e.g., house price predictions wrt square footage. Order the test points and project them onto the monotonicity constraint (need to read details). They were able to enforce shape constraints on the prediction function without hurting predictive accuracy.

Noise2Noise. Normally we train image denoisers on noisy -> clean image pairs. However, inspired by the fact that clean images are often averages of many noisy images (i.e., long exposure times), maybe we don't need a clean target! Instead, use a noisy target, sampled independently from the noise in the source image. On average, the gradients to the various noisy images will point toward the clean image! We don't need an explicit model of the noise for this to work: the only assumption is that the mean of the noisy images would be the clean one (if using L2 loss), or the median (if using L1 loss). An example where the mean is correct is underexposed or low-light images. An example where the median is correct is watermarks or randomly imposed text. For a fixed exposure budget, noisy data (which we can obtain more of) may yield superior results to clean data.

7/12 Session 2B: Neural Network Architectures

Music data can be converted into a sequence and fed to an LSTM to predict the next note identify repeated motifs. However, LSTM suffers from short memory. Maybe we should look for earlier matches of high length and/or low edit distance to the present suffix. MotifNet presents a smooth/learnable relaxation of dynamic programming to perform this matching. It's a lot more complicated than an LSTM but appears to perform robustly. It's not yet scaled up enough to handle real music.

Another music talk followed, this time by Google's Magenta. Some very cool interpolations of drum beats and animal sketches. Lots more at: g.co/magenta/musicvae

Autoregressive generative models: RNN outputs p(x_t | x_1, …, x_{t-1}). Unlike with GANs, it's possible to evaluate the probability density function.
PixelRNN has to summarize entire history in hidden state, which leads to problems with long-range dependencies.
PixelCNN/PixelCNN++ are better because they have immediate access to a limited receptive field, but still can't easily access outside this receptive field.
SNAIL architecture interleaves 1D causal convolution with self-attention. PixelSNAIL combines the residual blocks from PixelCNN with 2D causal convolutions and self-attention. It's found that the effective receptive field for the gradients effectively covers all previous points in the sequential order.

CNNs appear to beat RNNs in many sequence problems. However, to get long-scale dependence, many layers or large kernels are needed, resulting in lots of parameters and

compute. Self-attention is useful, but has quadratic complexity in the number of pixels. The Image Transformer scales self-attention to larger images by generating them in blocks.

7/12 Session 3: Optimization (Non-Convex)

Regularization improves generalization and sometimes storage cost.
CNNs have fewer weights (pruning) used across an image (repeating). Sometimes the weights are low-precision (quantization), or decayed to prevent large values.
Each of these techniques can be thought of as a constraint on the weights. This work quantifies the benefit of regularization via the covering dimension of the constraint set.

Many works referenced optimization on Riemannian manifolds, so I searched for background knowledge: https://arxiv.org/abs/1407.5965

7/13 Test of Time Award

One of the earliest works to use deep learning for language processing! They found rare words were not being trained well. To tackle this, they pulled similar words' representations closer together (e.g., feline is similar to cat). As neural networks are able to learn deep features, we're moving from feature engineering to architecture engineering.

7/13 Session 1: Computer Vision

Inexact graph matching problem: want to find a graph that roughly matches a second graph or a subgraph of it. They formulate higher-order matching between two sets of points, given partial or incomplete information, as a matching between two random clique complexes. Consider the graph whose vertex set are the k-cliques of G, with an edge between them if they share a face in G... this work involves sophisticated graph theory and spectral theory that can't be gleaned from a short talk. Experimentally, they are able to match visual scenes taken from different angles, with occlusion.

Context-free grammars (CFGs) are natural choices to model sequences of human actions, which can be non-Markovian (may depend on history). Note that in the Chomsky hierarchy, CFGs are one step closer to fully general Turing machines, than Markov models.
The Earley parser processes strings of a given CFG by iterating 3 operations: predict, scan, complete. When given noisy sequence data (e.g., audio or video), past methods require it to be segmented and classified into symbols to be parsed. However, this is non-optimal, because grammar rules are not considered in the classification process. The present work integrates the Earley parser with a classifier to parse sequence data which is neither segmented nor labeled. It achieves state-of-the-art for human activity prediction.

A program is an interpretable and executable way to describe behaviors. For example, when watching a cooking recipe, or a maze-solving demonstration, or a video game, a human can recognize where the demonstration diverges because, e.g., we're lacking an ingredient. In program synthesis, we try to synthesize the underlying program from a set of demonstration sequences (e.g., videos). Each demo is encoded. The encodings are fed to a reviewer LSTM that identifies what makes each particular demo unique, and converts them to a feature representation for each demo.. A relation module tries to identify the branching conditions (e.g., if-else) of the program by comparing pairs of demo features. The relation module thus converts the demo features into a program vector, which a decoder converts into a program.

Appearance condition and motion condition are used to generate realistic videos. There's an appearance discriminator and a motion discriminator. Video classification and keypoint regression are done to form better representations of the video. Loss is adversarial + perceptual ranking + the auxiliary tasks. Experimentally, they're able to generate realistic transitions between emotional expressions, and aircraft handling signals (arm motions). GitHub source coming soon.

7/13 Session 2: Neural Network Architectures

There are two dimensions to optimize: FLOPs and memory bandwidth. WaveNet requires 100 million FLOPs to generate one second of audio, and its sequential nature gives it a high memory bandwidth as well. Sparse WaveRNN makes several optimizations to become efficient enough for real-time generation on a mobile CPU! They increase the state size of the RNN but make the connections between them sparse. They also reduce memory bandwidth by generating 16 audio samples at a time.

How to make architecture search more efficient? Two existing methods are ENAS (covered in a Wednesday session that I missed: https://arxiv.org/abs/1802.03268) and SMASH.
A linear-sized "one-shot model" can encapsulate an exponential search space. Zeroing out subsets of the operations yields a candidate architecture. But how is it that a single shared set of weights can be used to evaluate a diverse set of architectures? Let's reframe: start with a large neural network and prune away the least useful operations. We try zeroing out some operations, and check both how much the output changes, and how much accuracy changes. One-shot training set behavior appears to be a good predictor of one-shot validation set behavior, which appears to be a good predictor of final model behavior.

Many neural network architectures take an existing human-designed model as a starting point, and make improvements. However, they perform layer-wise modifications that can't modify network topology. This work uses a tree-structured RL meta-controller. It makes decisions such as whether to transform a node to have multiple child nodes.

Long-term dependency propagation through RNNs is improved by adding an unsupervised auxiliary loss to the objective. The auxiliary loss makes the RNN to either reconstruct previous events or predict next events in the sequence.

7/13 Invited Talk: Language to Action: towards interactive task learning with physical agents

Grounded hierarchical task structure with semantic labels allow reasoning and communication about tasks and planning between robots and humans. Humans understand each other using the common ground of mutual knowledge, shared beliefs, perceptual co-presence (access to the same scene when we're located together), and shared past experience.

Robots, even if co-present, don't have shared perceptual basis because they don't understand the environment the same way. Robots also don't have our wealth of commonsense knowledge. We have a lot of implied knowledge: "straw the strawberries into pieces" implies that we'll likely need to place the strawberries on a cutting board and use a knife. We need to ground language to perception, action, and action-effect prediction.

Can we explicitly model physical causality? Crowdsourcing was used to identify relations between verbs, effects, and nouns. Robot has an interaction policy (trained by RL) for asking questions that allow it to learn from a human.

Action-effect prediction: we hope an agent can map a verb-noun description of an action, to the pictures that correspond to its effect, and vice-versa. It should learn from a small number of annotated examples to make it possible to learn through communication. Action-effect embedding is trained interactively as the human provides more examples. Finished with an amusing video of a human "teaching" a robot how to make a smoothie.

There was an audience question about the role of logical representations vs neural network representations. Underlying high-level logical concepts are lower-level representations. Perhaps as we obtain more data, neural networks can be used to build up the logical representations. Hence, the two approaches are not opposed to one another.

7/13 Invited Talk: Building Machines That Learn and Think Like People

The fields of AI and cognitive psychology might finally be mature enough to talk to one another. Let's reverse-engineer how intelligence works in the human mind and brain. Could we build a machine that grows into intelligence like a baby into an adult? Gotta reverse-engineer:

- "Core cognition": intuitive physics, social psychology

- “Child as scientist”: learning by experimentation

Probabilistic programs integrate our best ideas on intelligence:
- Symbolic languages for knowledge representation, composition and abstraction
- Probabilistic inference for causal reasoning under uncertainty and flexible inductive bias
- Neural networks for pattern recognition

We can infer people’s goals and constraints by observing behavior which optimizes the unknown goals. This is the idea behind “inverse optimization”.

Vision as inverse graphics: “Pearl meets Hinton meets Marr”. Using conditional independence to model inverse causality. Generative model provides training data for recognition task.

To go from a single 2D image to a 3D reconstruction, we might go through an intermediate 2.5D representation, such as a map of depth and surface normals.

Another example of symbolic and neural representations working together: many works on learning physics engines take for granted that there are discrete objects, with pairwise relations between them, and use neural networks to learn the nature of their interactions.

The goal of learning: making your code more awesome, algorithms that write algorithms. Pretty much every part of the programming process has some analog in human learning, which we must understand algorithmically. DREAMCODER has a wake phase in which it writes programs and a sleep phase in which it improves its representations.

7/13 Session 3: Causal Inference

The causal bandit problem can be thought of as a whitebox generalization of the multi-armed bandit problem. Given a causal graph and set of allowable interventions, we want to choose an intervention to optimize some outcome within T experiments. Unlike in multi-armed bandits, we can observe the values of intermediate variables after every experiment. By using the experiments to estimate model parameters, this work provides the first theoretically guaranteed algorithm for general causal bandits.

Much past work focused on learning causal models from observations. However, it’s also useful to learn models from interventions: for example, we may alter some gene and observe the result in a biological process. Variables are nodes in a DAG, modeled in a structural equation as a function of their parents and some independent noise. This work proposes a consistent, non-parametric algorithm for learning the structure, and characterizes equivalence classes which cannot be distinguished if we only have observations (limit of identifiability). Two graphs are in the same I-markov equivalence class (i.e., cannot be distinguished by observations and I-interventions) iff, after adding subgraphs representing the interventions in I, they have the

same skeleton (i.e., if we ignored directions of edges) and v-structures. Furthermore, provided that the empty set (i.e., null intervention) is in I, general interventions provide exactly the same equivalence classes as hard interventions (i.e., interventions that set each variable to a constant, rather than depending on its parents).

Given a budget of k interventions, each being a modification of one variable in I, want to maximize the number of edges of the causal graph whose directions we can resolve. Since this objective is submodular, a greedy algorithm achieves a 1 - 1/e approximation of the optimal value. Furthermore, the experiments can all be determined in advance, allowing them to be performed in parallel.

HAFVF, a new change detection algorithm inspired by animal behavior, with applications to RL in bandit tasks, denoising, auto-regressive models, and SGD. It tries to detect when the environment changes, and therefore the learner should forget what it knows. Current volatility estimate is conditioned on the past.

Motivating application: pricing a product in the digital economy. You might plot demand vs price and fit a regression, but in reality there are lots of confounders such as seasonality (maybe winter sales correspond to points with high price and high demand). Inspired by this problem, this work defines the partial linear regression (PLR) problem, develops a bunch of theory, and an efficient algorithm.

Want to infer causal structure. Point estimates result in too many high-scoring DAGs, and exact Bayesian posterior inference is intractable, so we're forced to consider approximate methods. Naive MCMC, modifying an edge at a time, is too slow; sampling on topological orderings speeds up mixing.

7/14 Game-Theoretic Mechanisms For Data and Information

https://gradanovic.github.io/gtmdi18ws/papers/GaMeDATA18_paper_6.pdf
DataBright. Amazon Turk data quality sucks because there's very little incentive to do well. Instead of paying data contributors once per contribution, after which they lose ownership of the data, how about we pay them per usage of the data, continuously? But there's a trust issue: if contributors are not paid immediately, data can be stolen and no payment received.
A distributed ledger can track ownership of the data (who should get paid) and transactions (who should pay). Since owners have a vested interest in keeping the dataset clean, they can also do quality control. Thus, the current "shareholders" vote to accept or reject new additions to the dataset. A user can then choose a model and the data to train on, and pays both the dataset owners and the training compute workers in cryptocurrency. Splitting the data from the labels (supposedly) prevents the workers from stealing any useful data.

7/14 Adaptive and Learning Agents

http://ala2018.it.nuigalway.ie/papers/ALA_2018_paper_29.pdf
IBM developed an inverse reinforcement learning method that only needs to be trained on an expert's sequence of states (not even actions!). An RNN tries to predict the next state, and then an RL agent attempts to achieve the predicted state.

7/14 AutoML (afternoon portion)

Best Paper Award on Active Learning by RL

Unlabeled data is cheap, annotation is expensive, and not all annotations are particularly useful. This motivates active learning, where you iteratively select queries on which to spend your annotation budget. Hand-engineered criteria produce inconsistent results; instead, let's try to learn active learning! RL phrasing: policy actions correspond to the query (choice of data point to annotate), and the reward is the increment of classifier accuracy. But how to generalize across different heterogeneous datasets?

Inspired by DietNet, run on fixed-dimension embedding of data points. RL objective is augmented by autoencoder and entropy terms for regularization. Need to train on multiple datasets for meta-network to learn to domain-adapt. Note that if the horizon is long (i.e., we want the ability to annotate very large datasets), it takes much longer to train the meta-learner. For this reason, unsupervised learning, which typically benefits from a lot of data, would be hard to meta-learn.

Invited Talk (Chelsea Finn): Meta-Learning

Goals of meta-learning: effectively reuse data on new tasks, replace manual engineering of architectures and hyperparameters, adapt to unexpected scenarios, learn to use weak supervision. Can be viewed as a few-shot supervised regression on datasets.
f: x_train, y_train, x_test -> y_test.

Four desirable properties for meta-learning algorithms.
Expressive power: the ability for f to represent a range of learning procedures (universal learning function approximator)
Consistency: learned learning procedure will solve any task, given enough data (even if out-of-distribution)
First-order (i.e., no need to differentiate through learning): yields better scalability to long learning processes
Uncertainty awareness: to deal with ambiguity

Blackbox approaches that have expressive power but not consistency:
y_test = f(x_train, y_train, x_test; theta) or y_test = f(x_test; g(x_train, y_train; theta))
They're also hard to optimize because they essentially have to learn how to learn *from scratch".

More structured approaches can be based on nearest neighbors in a learned metric space, or gradient descent from learned initialization.
To model ambiguity, can we sample classifiers? Want to learn a prior where a random kick can put us in different modes.
Many meta-learning works were referenced, meeting different subsets of the four properties.

Evolutionary Algorithms and RL for Architecture Search

RL approach: an LSTM chooses a sequence of "actions" which piece components together until a cell is formed.

Evolutionary approach: maintain a population of candidate networks, kill the oldest one, choose two parents and combine+mutate to form a new candidate. This seems to produce better architectures (along the model size vs accuracy tradeoff) than the RL approach.

Panel and Conclusions

A controversial suggestion brought up at the panel: should conferences automatically reject papers who don't open-source their models and code? If ICML and NIPS adopted such a policy, DeepMind would have to release their work in order to retain their research scientists!

The next AutoML challenge will take place at NIPS 2018.

7/15 CausalML

We have agents like self-driving cars that make interventions (not just observations) on the world, and we need the means to reason about their effects. Another example is functional genomics, where we want a causal model of disease so that we can intervene with a drug.

Donald Rubin offered an analogy between causal reasoning and quantum mechanics: you may be able to define two quantities, but you can only choose one to observe (e.g., the outcome if a particular patient takes a drug vs doesn't take the drug). Causality is motivated by physics (John Wheeler 1961), psychology & consciousness (Julian Jaynes 1964), and experimental design (William Cochran 1968). Don't believe something just because you read it in a respected book! It's more important to understand the big ideas than to understand the details!

The randomized experiment was a genius idea: eliminates confounding effects on treatment.
Invariance: P(recovery | treatment, status) remains invariant after the intervention on treatment.
In ML, there's a tradeoff between invariance and prediction. Usually, the knob is turned toward the latter, maximizing predictive accuracy. However, there are benefits to searching for invariant relationships, as these are likely to be causal and therefore remain stable (i.e., generalize better) under distribution changes. Anchor regression adds a penalty term to residual least squares, yielding models that are invariant to shift interventions in the chosen anchor variables.

Predictive models may learn policy-dependent relationships, which are non-causal and do not generalize well. For example, data from one hospital, where the doctors are aggressive in finding the right specialist to provide treatment, will show different mortality rates than another hospital. We say a model is *stable* if its parameters aren't sensitive to the dataset distribution, and *robust* if its performance is not sensitive. Supervised learning algorithms are highly sensitive to the policy used to choose actions in the training data.
"Counterfactual Normalization" (CN): want to learn predictive models that are stable to changes in policy even when not all confounders are observed. Instead of estimating P(Y | X), let's estimate P(Y(a) | X) for each treatment action a, where X are features and Y is the outcome. Some assumptions are made about the data-generating process, e.g., that all factors affecting a are observed (in X). Time of measurement can be a confounder, so model explicitly as a marked point process (MPP), where each event comes with a timestamp.
With baseline GP, the risk of new patient depends on *why* they were given the treatment. The counterfactual Gaussian process (CGP) makes stable predictions when the policy changes.
Example: smoking -> meningitis -> blood pressure, where "smoking" is unobserved.
In some hospitals, smokers are way less likely to be prescribed beta blockers, so we also have a confounding path: smoking -> beta blocker -> blood pressure. CN is stable to changes in the policy P(beta blocker | smoking)!
We start by identifying "vulnerable" variables, which have an unstable path to the condition (meningitis in this example) where one or more distributions along the path may be perturbed across datasets. To get a stable model, only learn influence along stable paths.

Note from a poster: policy improvement methods in RL typically assumes unconfoundedness in the observational data. In reality, the choice of actions may be correlated with later state transitions via unmodeled common causes.

The final talk opened a causal discussion of fairness criteria. Motivations include criminal justice.
Some criteria take the forms Yhat indep A, Yhat indep A | Y, or Y indep A | Yhat.
Equity: similar people should be treated similarly.
Equal protection: people shouldn't be discriminated on the basis of protected attributes.
Luck egalitarianism: people shouldn't be disadvantaged by circumstances beyond their control.
Observational notions are inadequate. Ignoring protected attributes is inadequate, as their influence can leak through proxy variables, and we may want to cancel out their effects explicitly. Unfortunately, the politicized public policy discussion favors solutions that simply ignore the protected attributes.

Most criteria are group notions: it's ok for some individuals to be harmed if an equal number are helped. A more individual notion (albeit harder to measure and satisfy) is counterfactual fairness: P(Yhat(a,U)=y | X=x,A=a) = P(Yhat(a',U)=y | X=x,A=a) for all y and a'.

On the panel, Rubin stressed that if you're doing to do an observational study, you should strip out the outcome data. Design the study in detail, with agreement from experts on both sides, before looking at the data! He also talked about the flaws of randomized trials, where compliance is imperfect, and long-term effects are not studied. He recommends always following up, never putting too much trust in a single study.