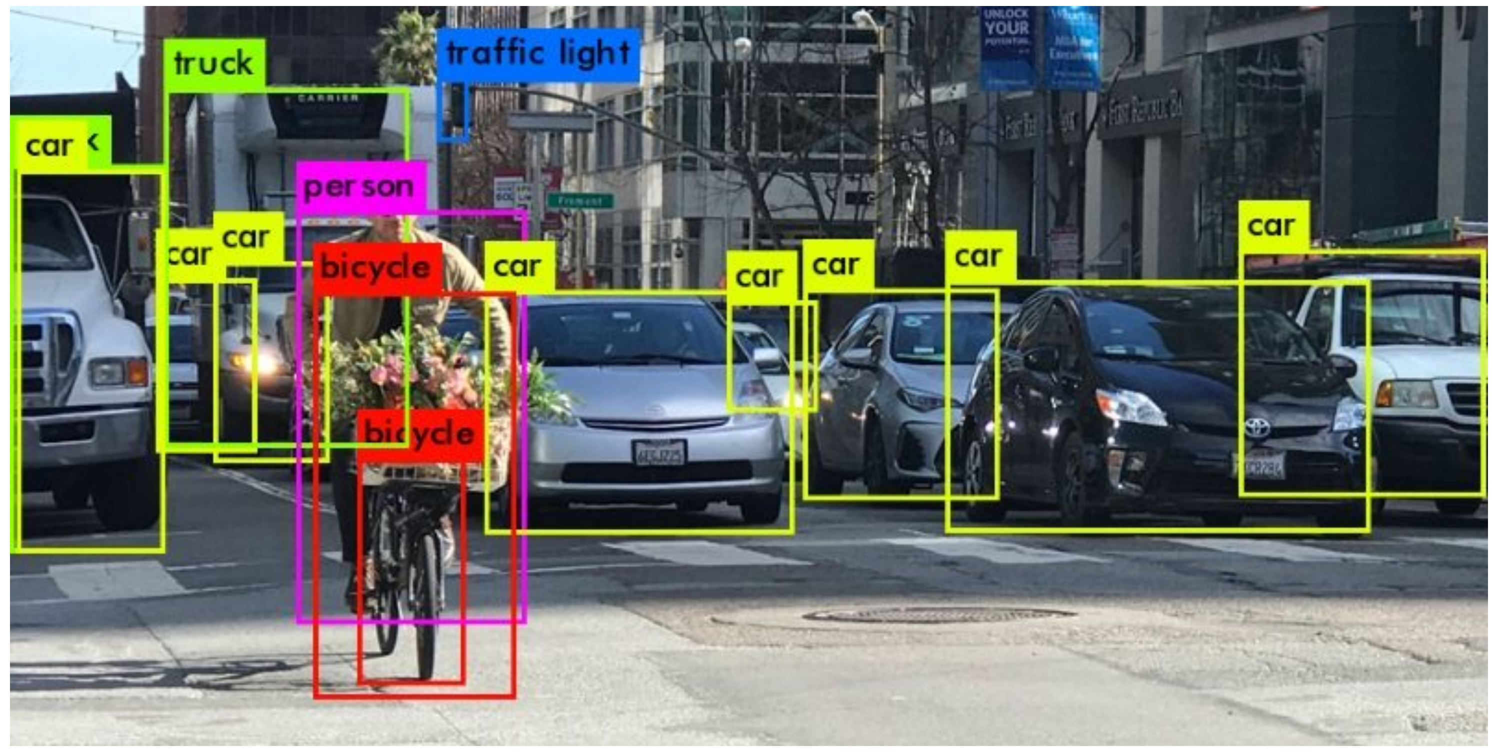
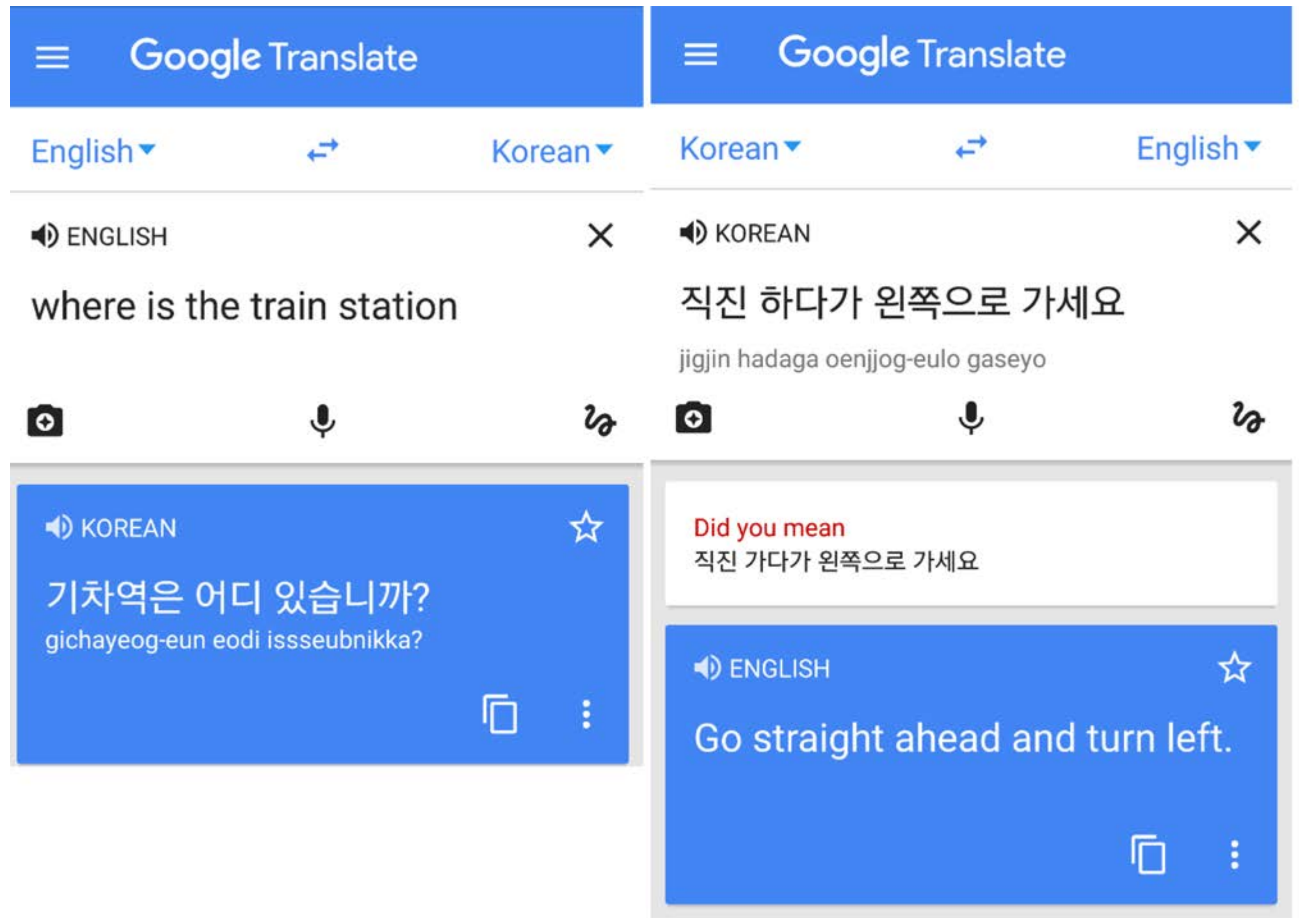
A Blueprint of Standardized and Composable ML

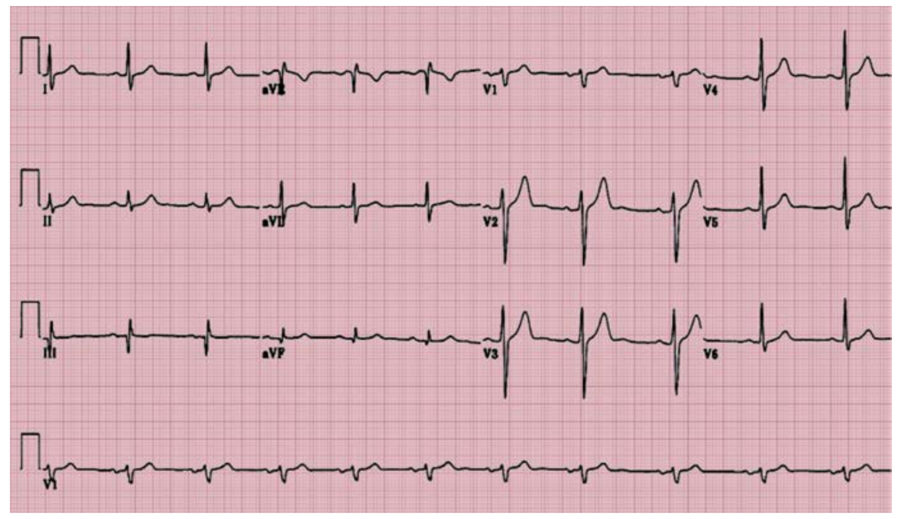
Eric Xing and
Zhiting Hu

Petuum & Carnegie Mellon

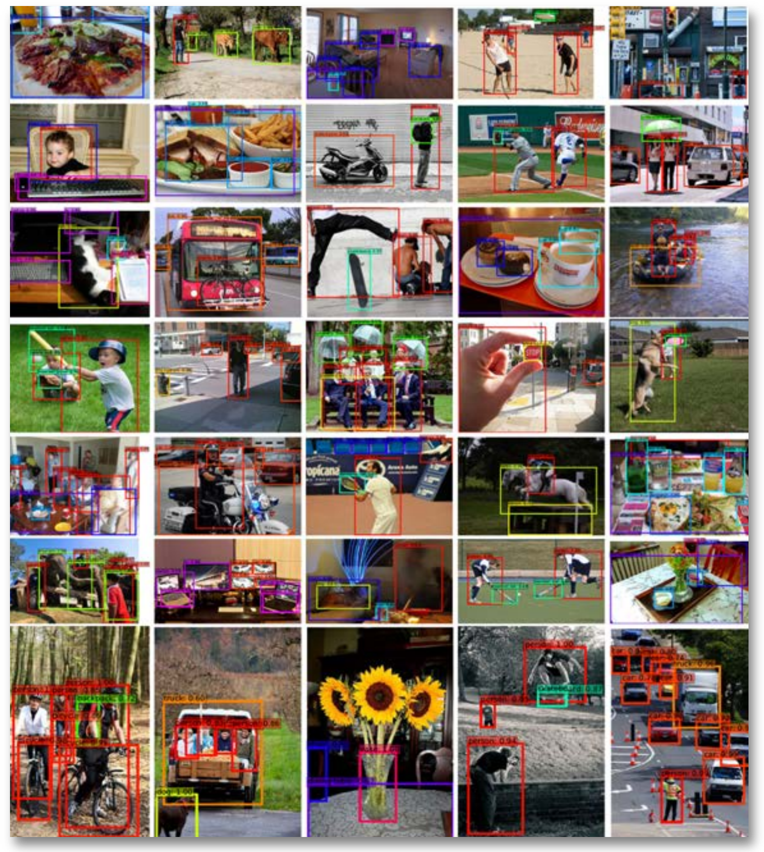
The universe of problems ML/AI is trying to solve







Data and experiences of all kinds



Data examples

Type-2 diabetes is 90% more common than type-1

Constraints

Rewards

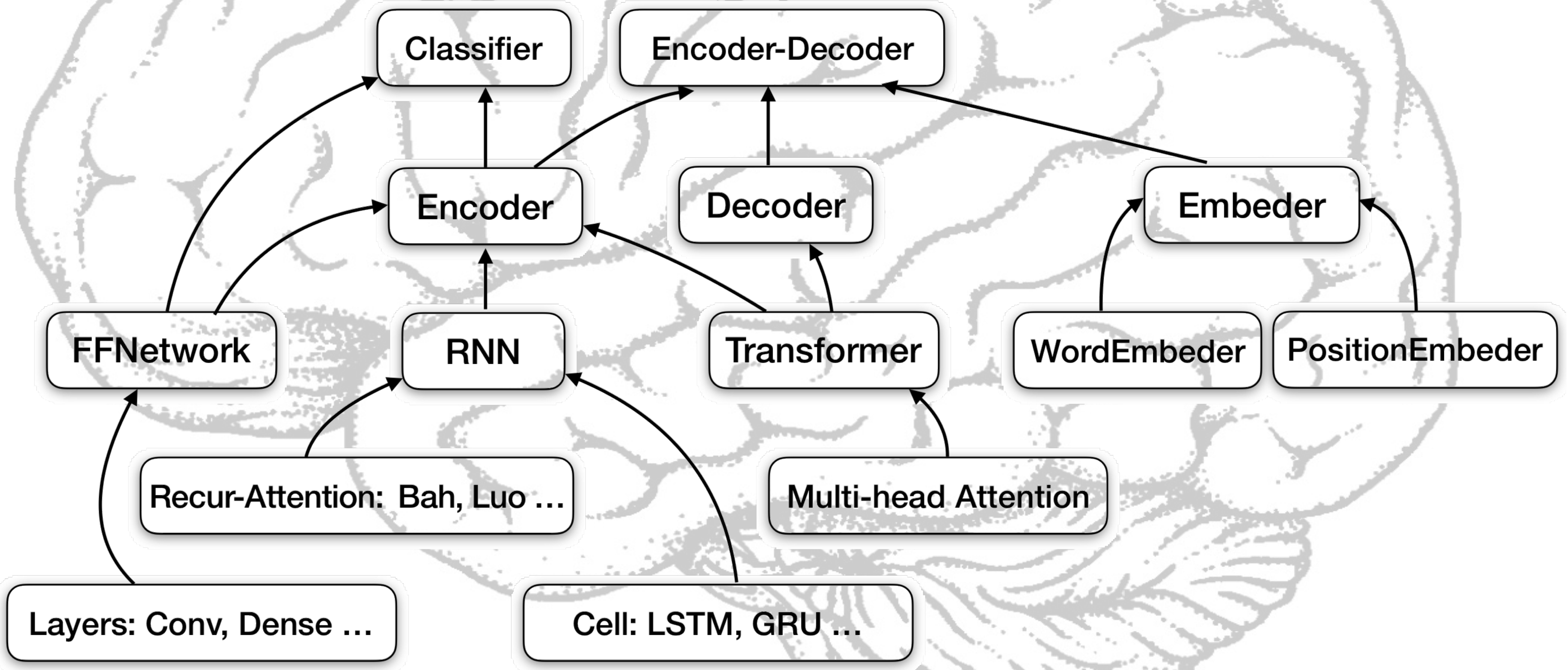
Auxiliary agents

Adversaries

…

And all combinations of of that …

How human beings solve them ALL?



Petuum

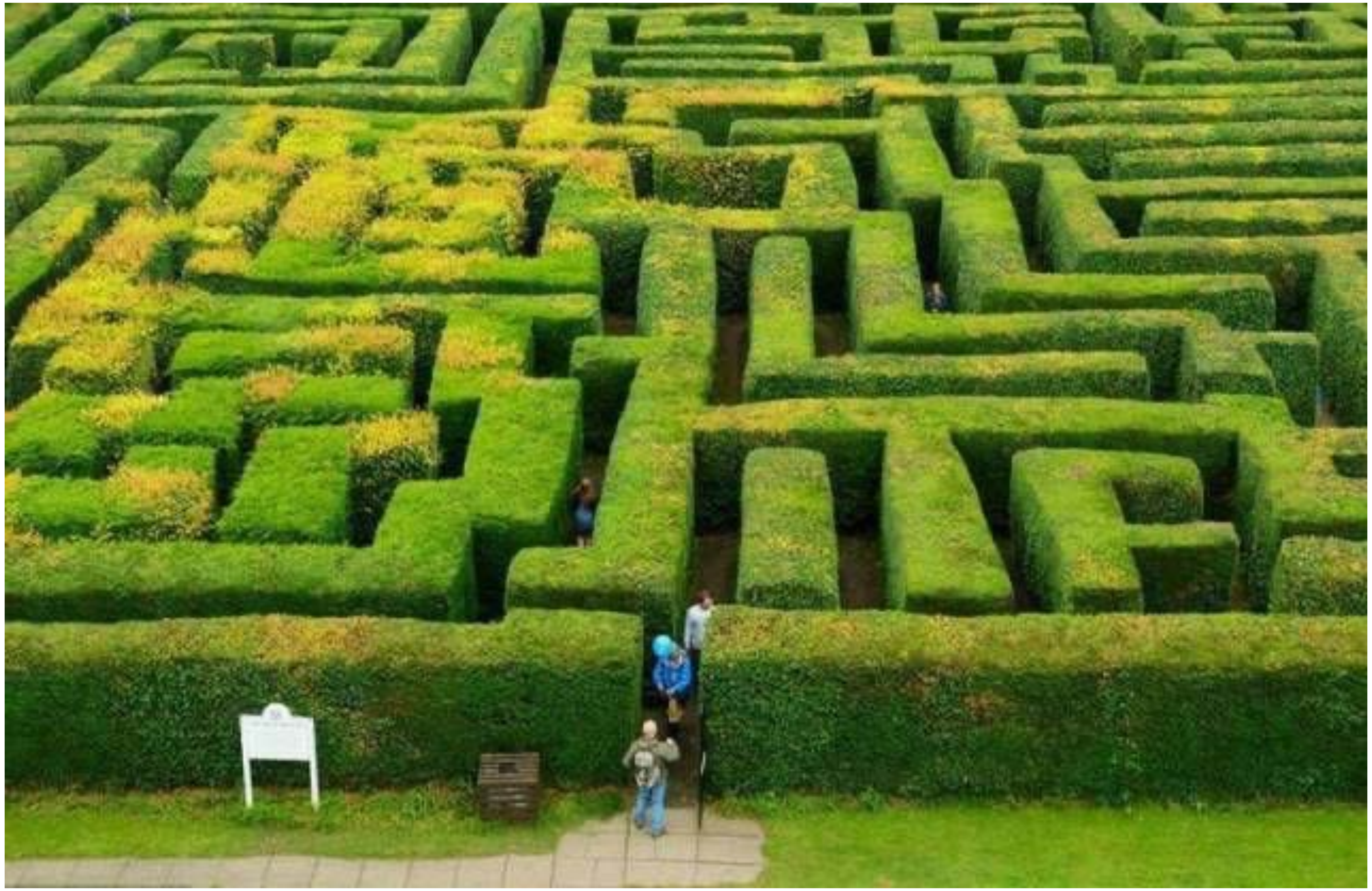
The Zoo of ML/AI Models

- Neural networks
  - Convolutional networks
  - AlexNet, GoogleNet, ResNet
  - Recurrent networks, LSTM
  - Transformers
  - BERT, GPT2
- Graphical models
  - Bayesian networks
  - Markov Random fields
  - Topic models, LDA
  - HMM, CRF
- Kernel machines
  - Radial Basis Function Networks
  - Gaussian processes
  - Deep kernel learning
  - Maximum margin
  - SVMs
- Decision trees
- PCA, Probabilistic PCA, Kernel PCA, ICA
- Boosting

The Zoo of algorithms and heuristics

maximum likelihood estimation
reinforcement learning as inference
data re-weighting
inverse RL
policy optimization
active learning
data augmentation
actor-critic
reward-augmented maximum likelihood
label smoothing
imitation learning
softmax policy gradient
adversarial domain adaptation
posterior regularization
GANs
constraint-driven learning
knowledge distillation
intrinsic reward
generalized expectation
prediction minimization
regularized Bayes
learning from measurements
energy-based GANs
weak/distant supervision

Really hard to navigate, and to realize



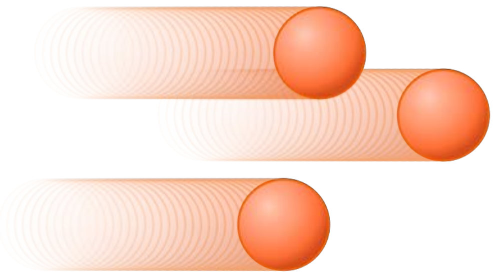
- Depending on individual expertise and creativity
- Bespoke, delicate pieces of art
- Like an airport with different runways for every different types of aircrafts

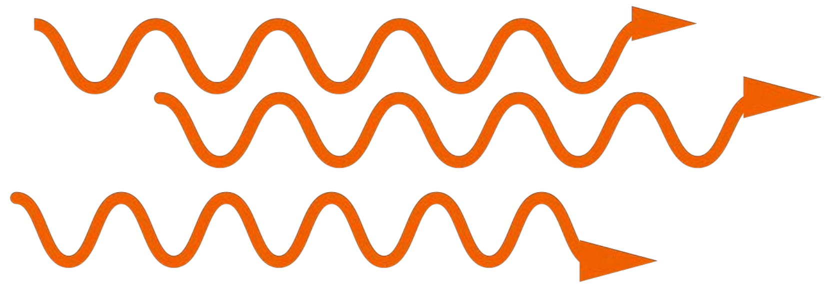
Petuum

Physics in the 1800's

- Electricity & magnetism:
  - Coulomb's law, Ampère, Faraday, ...

- Theory of light beams:
  - Particle theory: Isaac Newton, Laplace, Plank
  - Wave theory: Grimaldi, Chris Huygens, Thomas Young, Maxwell





- Law of gravity
  - Aristotle, Galileo, Newton, …

Petuum

Maxwell's equations

Diverse electro-magnetic theories

Maxwell's Eqns: original form

$e + \frac{df}{dx} + \frac{dg}{dy} + \frac{dh}{dz} = 0$	(1)	Gauss' Law
$\mu\alpha = \frac{dH}{dy} - \frac{dG}{dz}$, $\mu\beta = \frac{dF}{dz} - \frac{dH}{dx}$, $\mu\gamma = \frac{dG}{dx} - \frac{dF}{dy}$	(2)	Equivalent to Gauss' Law for magnetism
$P = \mu\left(\gamma\frac{dy}{dt} - \beta\frac{dz}{dt}\right) - \frac{dF}{dt} - \frac{d\Psi}{dx}$, $Q = \mu\left(\alpha\frac{dz}{dt} - \gamma\frac{dx}{dt}\right) - \frac{dG}{dt} - \frac{d\Psi}{dy}$, $R = \mu\left(\beta\frac{dx}{dt} - \alpha\frac{dy}{dt}\right) - \frac{dH}{dt} - \frac{d\Psi}{dz}$	(3)	Faraday's Law (with the Lorentz Force and Poisson's Law)
$\frac{d\gamma}{dy} - \frac{d\beta}{dz} = 4\pi p'$, $p' = p + \frac{df}{dt}$; $\frac{d\alpha}{dz} - \frac{d\gamma}{dx} = 4\pi q'$, $q' = q + \frac{dg}{dt}$; $\frac{d\beta}{dx} - \frac{d\alpha}{dy} = 4\pi r'$, $r' = r + \frac{dh}{dt}$	(4)	Ampère-Maxwell Law
$P = -\xi p \quad Q = -\xi q \quad R = -\xi r$		Ohm's Law
$P = kf \quad Q = kg \quad R = kh$		The electric elasticity equation (**E** = **D**/ε)
$\frac{de}{dt} + \frac{dp}{dx} + \frac{dq}{dy} + \frac{dr}{dz} = 0$		Continuity of charge

Simplified w/ rotational symmetry

$$\nabla \cdot \mathbf{D} = \rho_V$$

$$\nabla \cdot \mathbf{B} = 0$$

$$\nabla \times \mathbf{E} = -\frac{\partial \mathbf{B}}{\partial t}$$

$$\nabla \times \mathbf{H} = \frac{\partial \mathbf{D}}{\partial t} + \mathbf{J}$$

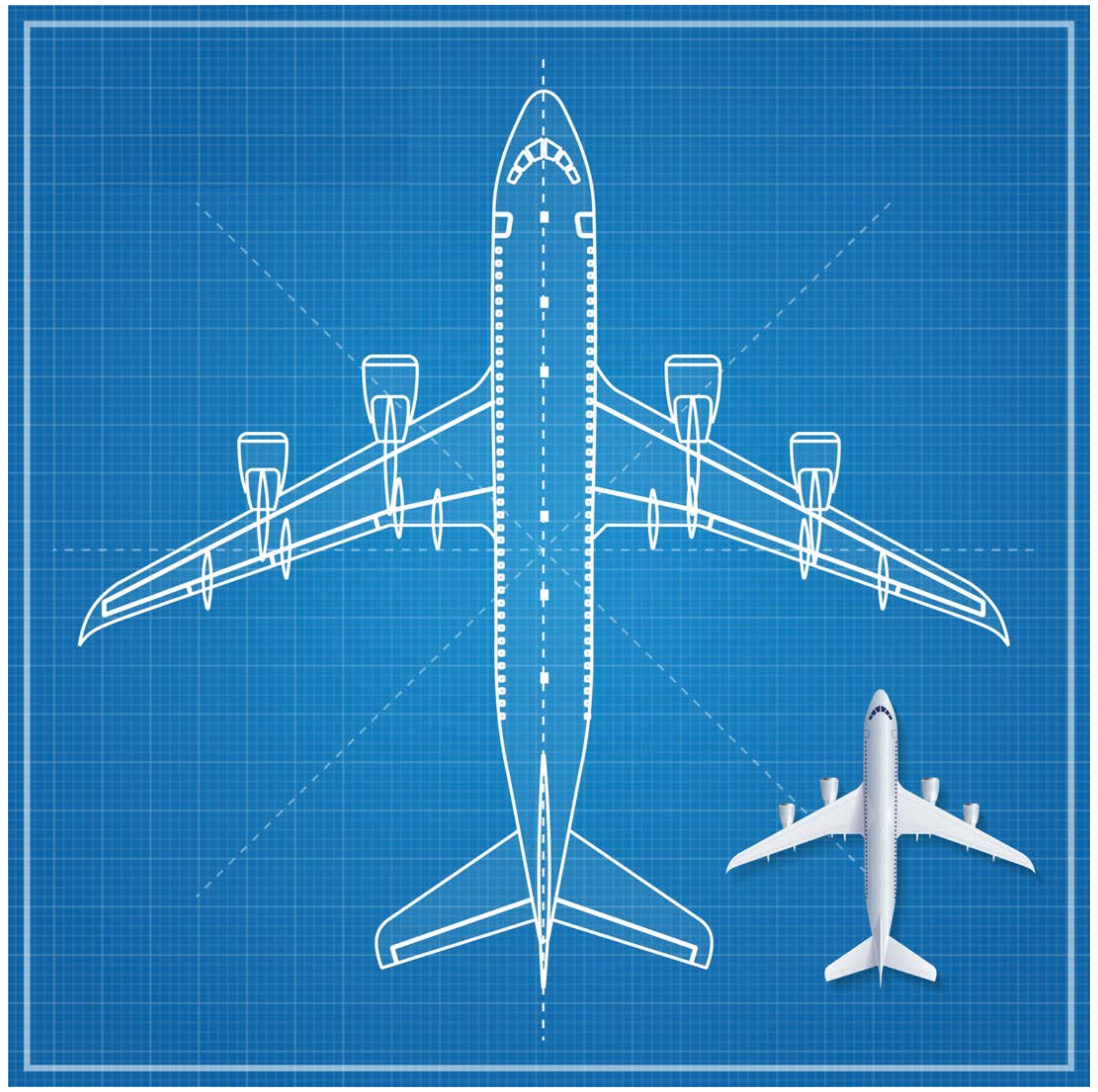
Further simplified w/ symmetry of special relativity

$$\varepsilon^{uvk\lambda}\partial_v F_{k\lambda} = 0$$

$$\partial_v F^{uV} = \frac{4\pi}{c} j^u$$

Petuum

How about a blueprint of ML



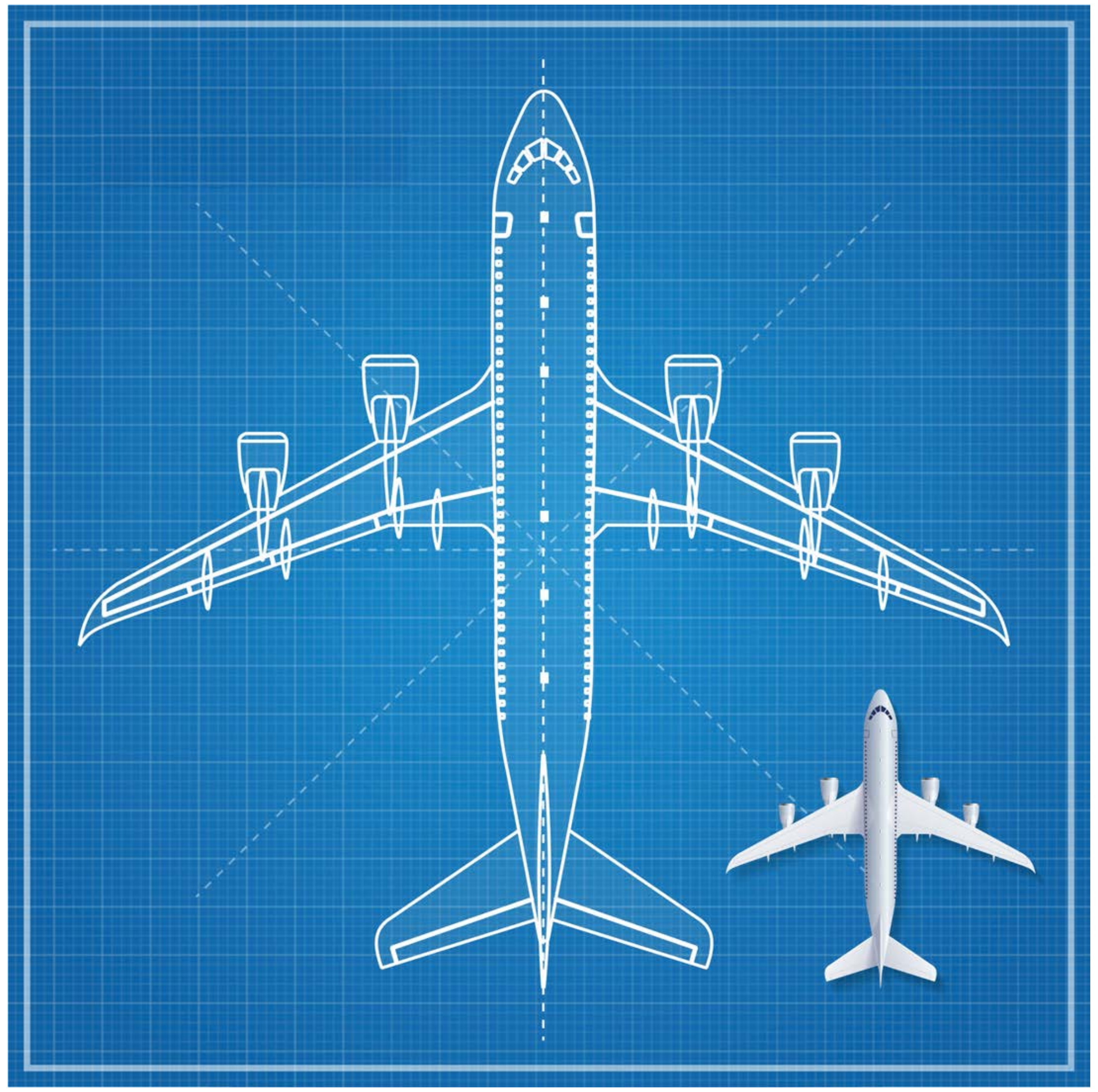
- Loss
- Optimization solver
- Model architecture

$$\min_{\theta} \mathcal{L}(\theta)$$

Optimization solver | Loss | Model architecture

How about a blueprint of ML

- Loss
- Optimization solver
- Model architecture



$$\min_{q,\theta} \; \mathbb{E} - \mathbb{D} - \mathbb{H}$$

Experience Divergence Uncertainty

Petuum

MLE at a close look:

- The most classical learning algorithm

- Supervised:
  - Observe data $\mathcal{D} = \{(\boldsymbol{x}^*, \boldsymbol{y}^*)\}$
  - Solve with SGD

$$\min_{\theta} - \mathbb{E}_{(\boldsymbol{x}^*, \boldsymbol{y}^*) \sim \mathcal{D}} \left[\log p_\theta(\boldsymbol{y}^* | \boldsymbol{x}^*) \right]$$

- Unsupervised:
  - Observe $\mathcal{D} = \{(\boldsymbol{x}^*)\}$, $\boldsymbol{y}$ is latent variable
  - Posterior $p_\theta(\boldsymbol{y}|\boldsymbol{x})$
  - Solve with EM:
    - E-step imputes latent variable $\boldsymbol{y}$ through expectation on complete likelihood
    - M-step: supervised MLE

$$\min_{\theta} - \mathbb{E}_{\boldsymbol{x}^* \sim \mathcal{D}} \left[\log \int_{\boldsymbol{y}} p_\theta(\boldsymbol{x}^*, \boldsymbol{y}) \right]$$

MLE as Entropy Maximization

- Duality between Supervised MLE and maximum entropy, when p is exponential family

Shannon entropy H

$$\min_{p(\boldsymbol{x},\boldsymbol{y})} \; H(p)$$

features $T(\boldsymbol{x},\boldsymbol{y})$

$$s.t.\; \mathbb{E}_p[T(\boldsymbol{x},\boldsymbol{y})] = \mathbb{E}_{(x^*,y^*)\sim\mathcal{D}}[T(\boldsymbol{x},\boldsymbol{y})]$$

data as constraints

Solve w/ Lagrangian method $\Downarrow$

$$p(\boldsymbol{x},\boldsymbol{y}) = \exp\{\boldsymbol{\theta}\cdot T(\boldsymbol{x})\} \; / \; Z(\boldsymbol{\theta})$$

Lagrangian multiplier $\boldsymbol{\theta}$

$$\min_{\theta} -\mathbb{E}_{(\boldsymbol{x}^*,\boldsymbol{y}^*)\sim\mathcal{D}}[\boldsymbol{\theta}\cdot T(\boldsymbol{x},\boldsymbol{y})] + \log Z(\boldsymbol{\theta})$$

Negative log-likelihood

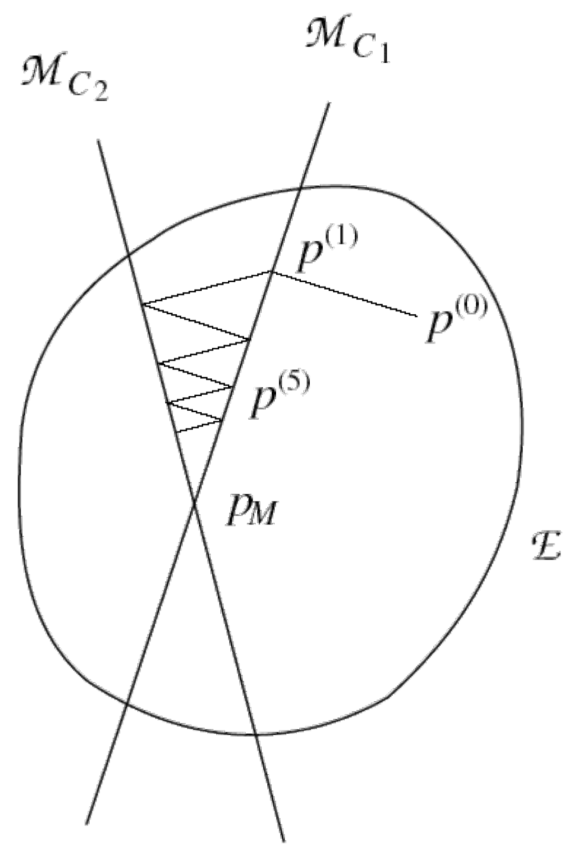
MLE as Entropy Maximization

- Unsupervised MLE can be achieved by maximizing the negative free energy:
  - Introduce auxiliary distribution $q(\boldsymbol{y}|\boldsymbol{x})$ (and then play with its entropy and cross entropy, etc.)

$$\log \int_{\boldsymbol{y}} p_\theta(\boldsymbol{x}^*, \boldsymbol{y}) = \mathbb{E}_{q(\boldsymbol{y}|\boldsymbol{x}^*)}\left[\log \frac{p_\theta(\boldsymbol{x}^*, \boldsymbol{y})}{q(\boldsymbol{y}|\boldsymbol{x}^*)}\right] + \mathrm{KL}\big(q(\boldsymbol{y}|\boldsymbol{x}^*) \,||\, p_\theta(\boldsymbol{y}|\boldsymbol{x}^*)\big)$$

$$\geq H\big(q(\boldsymbol{y}|\boldsymbol{x}^*)\big) + \mathbb{E}_{q(\boldsymbol{y}|\boldsymbol{x}^*)}[\log p_\theta(\boldsymbol{x}^*, \boldsymbol{y})]$$

Algorithms for Unsupervised MLE



$$\min_{\theta} - \mathbb{E}_{\boldsymbol{x}^* \sim \mathcal{D}} \left[\log \int_{\boldsymbol{y}} p_\theta(\boldsymbol{x}^*, \boldsymbol{y}) \right]$$

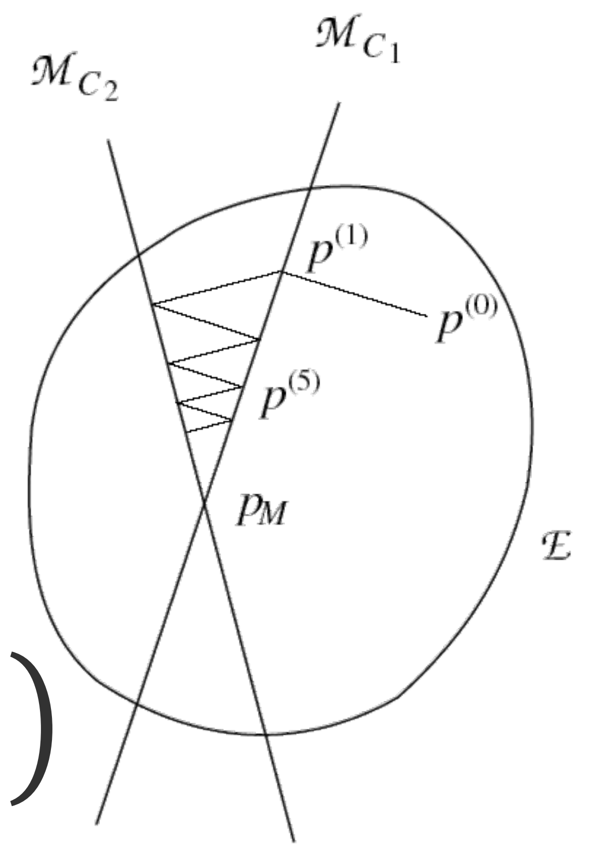
1) Solve with EM

$$\log \int_{\boldsymbol{y}} p_\theta(\boldsymbol{x}^*, \boldsymbol{y}) = \mathbb{E}_{q(\boldsymbol{y}|\boldsymbol{x}^*)} \left[\log \frac{p_\theta(\boldsymbol{x}^*, \boldsymbol{y})}{q(\boldsymbol{y}|\boldsymbol{x}^*)} \right] + \mathrm{KL}\big(q(\boldsymbol{y}|\boldsymbol{x}^*) \,||\, p_\theta(\boldsymbol{y}|\boldsymbol{x}^*)\big)$$
$$\geq H\big(q(\boldsymbol{y}|\boldsymbol{x}^*)\big) + \mathbb{E}_{q(\boldsymbol{y}|\boldsymbol{x}^*)}[\log p_\theta(\boldsymbol{x}^*, \boldsymbol{y})]$$

- E-step: Maximize $\mathcal{L}(q, \boldsymbol{\theta})$ w.r.t q, equivalent to minimizing KL by setting
  $$q(\boldsymbol{y}|\boldsymbol{x}^*) = p_{\theta^{old}}(\boldsymbol{y}|\boldsymbol{x}^*)$$
- M-step: Maximize $\mathcal{L}(q, \boldsymbol{\theta})$ w.r.t $\boldsymbol{\theta}$: $\max_{\theta} \mathbb{E}_{q(\boldsymbol{y}|\boldsymbol{x}^*)}[\log p_\theta(\boldsymbol{x}^*, \boldsymbol{y})]$

Petuum

Algorithms for Unsupervised MLE (cont'd)



$$\log \int_{\boldsymbol{y}} p_\theta(\boldsymbol{x}^*, \boldsymbol{y}) = \mathbb{E}_{q(\boldsymbol{y}|\boldsymbol{x}^*)}\left[\log \frac{p_\theta(\boldsymbol{x}^*, \boldsymbol{y})}{q(\boldsymbol{y}|\boldsymbol{x}^*)}\right] + \mathrm{KL}\big(q(\boldsymbol{y}|\boldsymbol{x}^*) \,||\, p_\theta(\boldsymbol{y}|\boldsymbol{x}^*)\big)$$

$$\geq H\big(q(\boldsymbol{y}|\boldsymbol{x}^*)\big) + \mathbb{E}_{q(\boldsymbol{y}|\boldsymbol{x}^*)}[\log p_\theta(\boldsymbol{x}^*, \boldsymbol{y})]$$

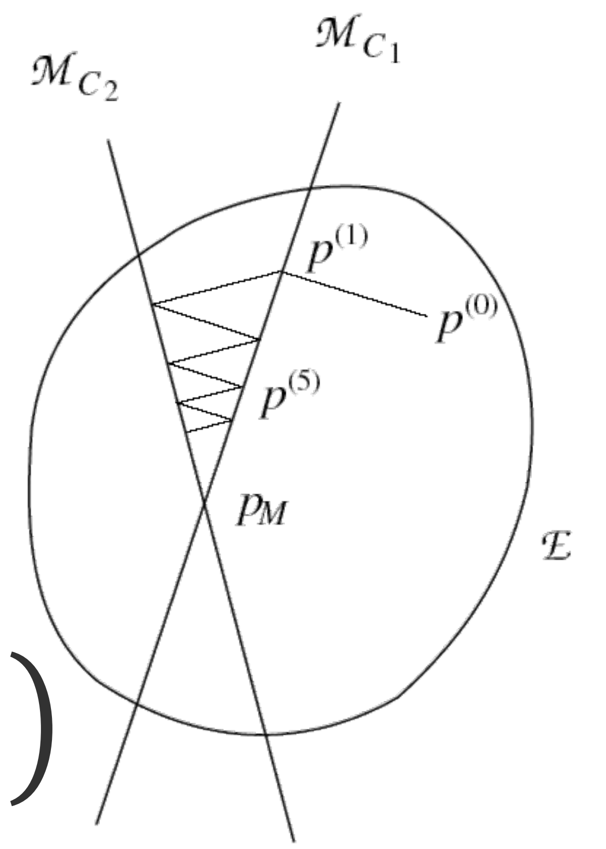
2) When model $p_{\boldsymbol{\theta}}$ is complex, directly working with the true posterior $p_{\boldsymbol{\theta}}(\boldsymbol{y}|\boldsymbol{x}^*)$ is intractable ⇒ Variational EM

- Consider a sufficiently restricted family Q of $q(\boldsymbol{y}|\boldsymbol{x})$ so that minimizing the KL is tractable
  - E.g., parametric distributions, factorized distributions

- E-step: Maximize $\mathcal{L}(q, \boldsymbol{\theta})$ w.r.t $q \in Q$, equivalent to minimizing KL
- M-step: Maximize $\mathcal{L}(q, \boldsymbol{\theta})$ w.r.t $\boldsymbol{\theta}$: $\max_\theta \mathbb{E}_{q(\boldsymbol{y}|\boldsymbol{x}^*)}[\log p_\theta(\boldsymbol{x}^*, \boldsymbol{y})]$

Petuum

Algorithms for Unsupervised MLE (cont'd)



$$\log \int_{\boldsymbol{y}} p_\theta(\boldsymbol{x}^*, \boldsymbol{y}) = \mathbb{E}_{q(\boldsymbol{y}|\boldsymbol{x}^*)}\left[\log \frac{p_\theta(\boldsymbol{x}^*, \boldsymbol{y})}{q(\boldsymbol{y}|\boldsymbol{x}^*)}\right] + \mathrm{KL}\big(q(\boldsymbol{y}|\boldsymbol{x}^*) \,||\, p_\theta(\boldsymbol{y}|\boldsymbol{x}^*)\big)$$

$$\geq H\big(q(\boldsymbol{y}|\boldsymbol{x}^*)\big) + \mathbb{E}_{q(\boldsymbol{y}|\boldsymbol{x}^*)}[\log p_\theta(\boldsymbol{x}^*, \boldsymbol{y})]$$

3) When q is complex, e.g., deep NNs, optimizing q in E-step is difficult (e.g., high variance) ⇒ Wake-Sleep algorithm [Hinton et al., 1995]

- Sleep-phase (E-step): $\min_{\phi} \mathrm{KL}(p_\theta(\boldsymbol{y}|\boldsymbol{x}^*) || q_\phi(\boldsymbol{y}|\boldsymbol{x}^*))$ ----> *Reverse KL*
- Wake-phase (M-step): Maximize $\mathcal{L}(q, \boldsymbol{\theta})$ w.r.t $\boldsymbol{\theta}$: $\max_{\theta} \mathbb{E}_{q(\boldsymbol{y}|\boldsymbol{x}^*)}[\log p_\theta(\boldsymbol{x}^*, \boldsymbol{y})]$

Other tricks: reparameterization in VAE ('2014), control variates in NVIL ('2014)

Quick summary of MLE

- Supervised:
  - Duality with MaxEnt
  - Solve with SGD, IPF …
- Unsupervised:
  - Lower bounded by negative free energy
  - Solve with EM, VEM, Wake-Sleep, …

- Close connections to MaxEnt
- With MaxEnt, algorithms (e.g., EM) arises naturally

Posterior Regularization (PR)

- Make use of constraints in Bayesian learning
  - An auxiliary posterior distribution $q(\theta)$
  - Slack variable ξ, constant weight $\alpha = \beta > 0$

$$\min_{q,\xi} \; -\alpha H(q) - \beta \mathbb{E}_q \left[\log p_\theta(\boldsymbol{x}, \boldsymbol{y}) \right] + \xi$$

$$s.t. \; -\mathbb{E}_q [f_\theta(\boldsymbol{x}, \boldsymbol{y})] \leq \xi$$

[Ganchev et al., 2010]

  - E.g., max-margin constraint for linear regression [Jaakkola et al., 1999] and general models (e.g., LDA, NNs) [Zhu et al., 2014] — *more later*

- Solution for q

$$q(\boldsymbol{\theta}) = \exp\left\{ \frac{\beta \log p_\theta(\boldsymbol{x}, \boldsymbol{y}) + f(\boldsymbol{x}, \boldsymbol{y})}{\alpha} \right\} / Z$$

More general learning leveraging PR

- No need to limit to Bayesian learning
- E.g., Complex rule constraints on general models [Hu et al., 2016], where
  - q can be over arbitrary variables, e.g., $q(\boldsymbol{x}, \boldsymbol{y})$
  - $p_\theta(\boldsymbol{x}, \boldsymbol{y})$ is NNs of arbitrary architectures with parameters $\boldsymbol{\theta}$

$$\min_{q,\theta,\xi} - \alpha H(q) - \beta \mathbb{E}_q \left[\log p_\theta(\boldsymbol{x}, \boldsymbol{y}) \right] + \xi$$

$$s.t.\ \mathbb{E}_{q(\boldsymbol{x},\boldsymbol{y})} \left[1 - r(\boldsymbol{x}, \boldsymbol{y}) \right] \leq \xi$$

E.g., $r(\boldsymbol{x}, \boldsymbol{y})$ is a 1st-order logical rule:

If sentence $\boldsymbol{x}$ contains word ``but''
⇒ its sentiment $\boldsymbol{y}$ is the same as the sentiment after "but"

EM for the general PR

- Rewrite without slack variable:

$$\min_{q,\theta} -\alpha H(q) - \beta \mathbb{E}_q \left[\log p_\theta(\boldsymbol{x}, \boldsymbol{y}) \right] - \mathbb{E}_{q(\boldsymbol{x},\boldsymbol{y})} \left[f(\boldsymbol{x}, \boldsymbol{y}) \right]$$

  - Solve with EM
    - E-step: $q(\boldsymbol{x}, \boldsymbol{y}) = \exp\left\{ \dfrac{\beta \log p_\theta(\boldsymbol{x}, \boldsymbol{y}) + f(\boldsymbol{x}, \boldsymbol{y})}{\alpha} \right\} / Z$
    - M-step: $\min_\theta \mathbb{E}_q \left[\log p_\theta(\boldsymbol{x}, \boldsymbol{y}) \right]$

Reformulating unsupervised MLE with PR

$$\log \int_{\boldsymbol{y}} p_\theta(\boldsymbol{x}^*, \boldsymbol{y}) \geq H\big(q(\boldsymbol{y}|\boldsymbol{x}^*)\big) + \mathbb{E}_{q(\boldsymbol{y}|\boldsymbol{x}^*)}[\log p_\theta(\boldsymbol{x}^*, \boldsymbol{y})]$$

- Introduce arbitrary $q(\boldsymbol{y}|\boldsymbol{x})$

$$\min_{q,\theta,\xi} \; -\alpha H(q) - \beta \mathbb{E}_q\Big[\log p_\theta(\boldsymbol{x}, \boldsymbol{y})\Big] + \xi$$

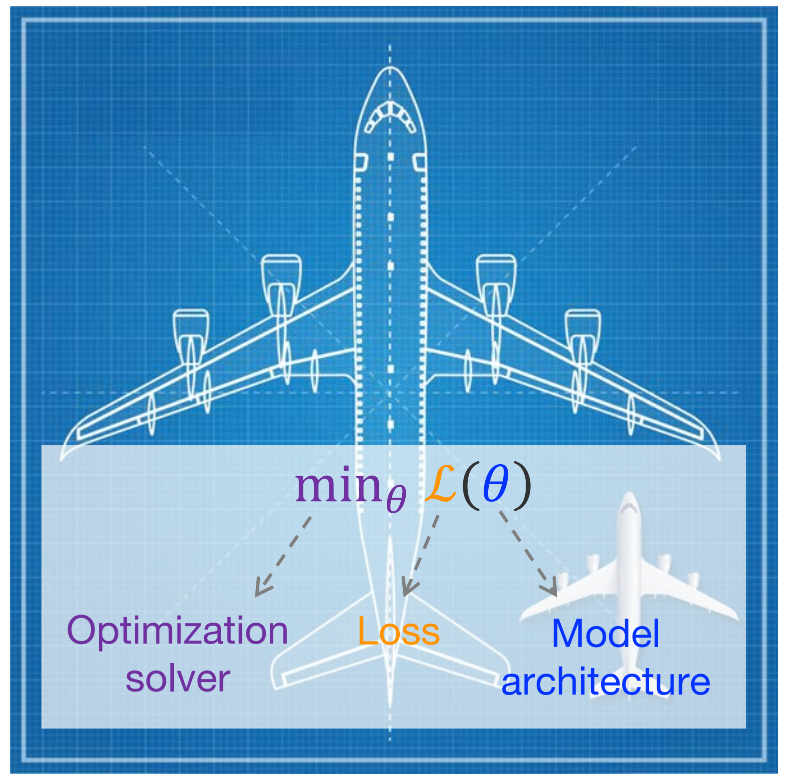
$$s.t. \; -\mathbb{E}_q\Big[f(\boldsymbol{x}\,;\mathcal{D})\Big] < \xi$$

Data as constraint.
Given $\boldsymbol{x} \sim \mathcal{D}$, this constraint doesn't influence the solution of q and $\boldsymbol{\theta}$

- $f(\boldsymbol{x}\,;\mathcal{D}) := \log \mathbb{E}_{x^* \sim \mathcal{D}}[\,\mathbb{1}_{\boldsymbol{x}^*}(\boldsymbol{x})\,]$
  - A constraint saying $\boldsymbol{x}$ must equal to one of the true data points
  - Or alternatively, the (log) expected similarity of $\boldsymbol{x}$ to dataset $\mathcal{D}$, with $\mathbb{1}(\cdot)$ as the similarity measure (we'll come back to this later)
- $\alpha = \beta = 1$

Petuum

The standard equation



$$\min_{q,\theta,\xi\geq 0} \alpha\mathbb{D}\left(q(\boldsymbol{x},\boldsymbol{y}), p_\theta(\boldsymbol{x},\boldsymbol{y})\right) - \beta\mathbb{H}(q) + \xi$$

$$s.t. \; -\mathbb{E}_{q(\boldsymbol{x},\boldsymbol{y})}\left[f(\boldsymbol{x},\boldsymbol{y})\right] < \xi$$

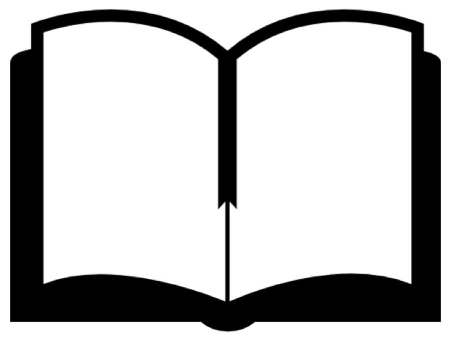
Equivalently:

$$\min_{q,\theta} -\mathbb{E}_{q(\boldsymbol{x},\boldsymbol{y})}\left[f(\boldsymbol{x},\boldsymbol{y})\right] + \alpha\mathbb{D}\left(q(\boldsymbol{x},\boldsymbol{y}), p_\theta(\boldsymbol{x},\boldsymbol{y})\right) - \beta\mathbb{H}(q)$$

3 terms:

Experiences	***Divergence***	***Uncertainty***
(exogenous regularizations)	*(fitness)*	*(self-regularization)*
e.g., data examples, rules	*e.g., Cross Entropy*	*e.g., Shannon entropy*

Textbook
$f(\boldsymbol{x},\boldsymbol{y}| \;.\;)$



Teacher
$q(\boldsymbol{x},\boldsymbol{y})$

Student
$p_\theta(\boldsymbol{x},\boldsymbol{y})$

Uncertainty

Petuum

Re-visit unsupervised MLE under SE

$$\min_{q,\theta} \; -\alpha H(q) - \beta \mathbb{E}_q \left[\log p_\theta(\boldsymbol{x}, \boldsymbol{y}) \right] - \mathbb{E}_q \left[\quad f(\boldsymbol{x}, \boldsymbol{y}) \quad \right]$$

$$f := f(\boldsymbol{x}\,;\,\mathcal{D}) = \log \mathbb{E}_{\boldsymbol{x}^* \sim \mathcal{D}} [\, \mathbb{1}_{\boldsymbol{x}^*}(\boldsymbol{x}) \,] \qquad \alpha = \beta = 1$$

$$q = q(\boldsymbol{y}|\boldsymbol{x})$$

Re-visit supervised MLE under SE

$$\min_{q,\theta} \; -\alpha H(q) - \beta \mathbb{E}_q \left[\log p_\theta(\boldsymbol{x}, \boldsymbol{y}) \right] - \mathbb{E}_{q(\boldsymbol{x},\boldsymbol{y})} \left[\; f(\boldsymbol{x}, \boldsymbol{y}) \quad \right]$$

$$f := f(\boldsymbol{x}, \boldsymbol{y}; \mathcal{D}) = \log \mathbb{E}_{(\boldsymbol{x}^*, \boldsymbol{y}^*) \sim \mathcal{D}} \left[\mathbb{1}_{(\boldsymbol{x}^*, \boldsymbol{y}^*)}(\boldsymbol{x}, \boldsymbol{y}) \right] \qquad \alpha = 1, \beta = \epsilon$$

Petuum

Active learning under SE

$$\min_{q,\theta} -\alpha H(q) - \beta \mathbb{E}_q \left[\log p_\theta(\boldsymbol{x}, \boldsymbol{y}) \right] - \mathbb{E}_{q(\boldsymbol{x},\boldsymbol{y})} \left[f(\boldsymbol{x}, \boldsymbol{y}) \right]$$

$f := f(\boldsymbol{x}, \boldsymbol{y}\,;\, Oracle) + u(\boldsymbol{x})$ $\qquad$ $\alpha = \tau\ (> 0), \beta = \epsilon$

$$f(\boldsymbol{x}, \boldsymbol{y}\,;\, Oracle) = \log \mathbb{E}_{\boldsymbol{x}^* \sim \mathcal{D},\, \boldsymbol{y}^* \sim Oracle(\boldsymbol{x}^*)} \left[\mathbb{1}_{(\boldsymbol{x}^*, \boldsymbol{y}^*)}(\boldsymbol{x}, \boldsymbol{y}) \right]$$

prediction uncertainty on $\boldsymbol{x}$, e.g., Shannon entropy $H(p_\theta(\boldsymbol{y}|\boldsymbol{x}))$

Equivalent to:

- Draw a data point $\boldsymbol{x}^*$ according to $\exp\{u(\boldsymbol{x})/\tau\}$
- Get label $\boldsymbol{y}^*$ for $\boldsymbol{x}^*$ from the oracle
- Maximize log likelihood on $(\boldsymbol{x}^*, \boldsymbol{y}^*)$

Reinforcement learning (RL) under SE -- I

$$\min_{q,\theta} -\alpha H(q) - \beta \mathbb{E}_q \left[\log p_\theta(\boldsymbol{x}, \boldsymbol{y}) \right] - \mathbb{E}_{q(\boldsymbol{x},\boldsymbol{y})} \left[f(\boldsymbol{x}, \boldsymbol{y}) \right]$$

- Map to RL language
  - $\boldsymbol{x}$ – *state* s, $\boldsymbol{y}$ – *action* a
  - $p_d(\boldsymbol{x})$ – state distribution
  - $Q_{\theta^t}(\boldsymbol{x}, \boldsymbol{y})$ – expected future reward of taking action $\boldsymbol{y}$ in state $\boldsymbol{x}$ and continuing the current policy p_{θ^t}

$$Q_{\theta^t}(\boldsymbol{x}, \boldsymbol{y}) = \mathbb{E}_{p_{\theta^t}}\left[\sum_{t=0}^{\infty} \boldsymbol{r_t} \mid \boldsymbol{x}_0 = \boldsymbol{x}, \boldsymbol{y}_0 = \boldsymbol{y}\right]$$

- RL-as-inference [Dayan'97; Levine'18, …]

$$f(\boldsymbol{x}, \boldsymbol{y}) := Q_{\theta^t}(\boldsymbol{x}, \boldsymbol{y}) \quad \alpha = \beta = \tau \ (> 0)$$

Reinforcement learning (RL) under SE -- II

$$\min_{q,\theta} \; -\alpha H(q) - \beta \mathbb{E}_q \left[\log p_\theta(\boldsymbol{x}, \boldsymbol{y}) \right] - \mathbb{E}_{q(\boldsymbol{x},\boldsymbol{y})} \left[\; f(\boldsymbol{x}, \boldsymbol{y}) \; \right]$$

- Map to RL language
  - $\boldsymbol{x}$ $- \; state \; s, \quad \boldsymbol{y} - action \; a$
  - $p_d(\boldsymbol{x})$ – state distribution
  - $Q_{\theta^t}(\boldsymbol{x}, \boldsymbol{y})$ – expected future reward of taking action $\boldsymbol{y}$ in state $\boldsymbol{x}$ and continuing the current policy p_{θ^t} $\quad Q_{\theta^t}(\boldsymbol{x}, \boldsymbol{y}) = \mathbb{E}_{p_{\theta^t}}[\sum_{t=0}^{\infty} \boldsymbol{r_t} \mid \boldsymbol{x}_0 = \boldsymbol{x}, \boldsymbol{y}_0 = \boldsymbol{y}]$

- Policy gradient

$$f(\boldsymbol{x}, \boldsymbol{y}) := \log Q_{\theta^t}(\boldsymbol{x}, \boldsymbol{y}) \qquad \alpha = \beta = 1$$

  - E-step $\quad q(\boldsymbol{x}, \boldsymbol{y}) = p_d(\boldsymbol{x}) p_{\theta^t}(\boldsymbol{y}|\boldsymbol{x}) Q_{\theta^t}(\boldsymbol{x}, \boldsymbol{y}) \; / \; Z$
  - M-step

$$\mathbb{E}_{q(\boldsymbol{x},\boldsymbol{y})}[\; \nabla_\theta \log p_\theta(\boldsymbol{y}|\boldsymbol{x}) \;] = 1/Z \cdot \mathbb{E}_{p_d(\boldsymbol{x}) p_\theta(\boldsymbol{y}|\boldsymbol{x})}[Q_\theta(\boldsymbol{x}, \boldsymbol{y}) \; \nabla_\theta \log p_\theta(\boldsymbol{y}|\boldsymbol{x}) \;]$$ *(Importance sampling est.)*

$$= 1/Z \cdot \nabla_\theta \mathbb{E}_{p_d(\boldsymbol{x}) p_\theta(\boldsymbol{y}|\boldsymbol{x})}[Q_\theta(\boldsymbol{x}, \boldsymbol{y}) \;]$$ *(Log-derivative trick)*

Conventional policy gradient objective

Petuum

Adversarial learning under SE

- For notation simplicity, we use $\boldsymbol{x}$ to replace $(\boldsymbol{x}, \boldsymbol{y})$

$$\min_{q,\theta} \; -\alpha\mathbb{H}(q) + \beta\mathbb{D}\Big(q(\boldsymbol{x}), p_\theta(\boldsymbol{x}) \Big) - \mathbb{E}_{q(\boldsymbol{x})}\Big[f(\boldsymbol{x}) \Big]$$

- Same as supervised MLE: $f := f(\boldsymbol{x}; \mathcal{D}),\ \alpha = 1,\ \beta = \epsilon$
- M-step is to $\min_\theta \mathbb{D}\Big(p_d(\boldsymbol{x}), p_\theta(\boldsymbol{x})\Big)$
- Solve with probability functional descent (PFD) [Chu et al., 2019]
  - $p_\theta(\boldsymbol{x})$ can be optimized by minimizing $\mathbb{E}_{p_\theta}[\Psi(\boldsymbol{x})]$, where $\Psi(\boldsymbol{x})$ is the influence function for $\mathbb{D}$ at p_{θ^t}
  - Ψ is obtained with convex duality

$$\Psi(\boldsymbol{x}) = \text{argmax}_\psi \, \mathbb{E}_{p_\theta}[\psi(\boldsymbol{x})] - \mathbb{D}^*(\psi)$$

Convex conjugate of $\mathbb{D}$

  - So the whole optimization is

$$\min_\theta \max_\psi \mathbb{E}_{p_\theta}[\psi(\boldsymbol{x})] - \mathbb{D}^*(\psi)$$

Petuum

Adversarial learning under SE

- For notation simplicity, we use $\boldsymbol{x}$ to replace $(\boldsymbol{x}, \boldsymbol{y})$

$$\min_{q,\theta} \; -\alpha \mathbb{H}(q) + \beta \mathbb{D}\left(q(\boldsymbol{x}), p_\theta(\boldsymbol{x})\right) - \mathbb{E}_{q(\boldsymbol{x})}\left[f(\boldsymbol{x}) \right]$$

- Same as supervised MLE: $f := f(\boldsymbol{x}; \mathcal{D}),\; \alpha = 1,\; \beta = \epsilon$
- M-step is to $\min_\theta \mathbb{D}\left(p_d(\boldsymbol{x}), p_\theta(\boldsymbol{x})\right)$
- Solve with probability functional descent (PFD) [Chu et al., 2019]
  - $p_\theta(\boldsymbol{x})$ can be optimized by minimizing $\mathbb{E}_{p_\theta}[\Psi(\boldsymbol{x})]$, where $\Psi(\boldsymbol{x})$ is the influence function for $\mathbb{D}$ at p_{θ^t}
  - Ψ is obtained with convex duality

$$\Psi(\boldsymbol{x}) = \mathrm{argmax}_\psi \, \mathbb{E}_{p_\theta}[\psi(\boldsymbol{x})] - \mathbb{D}^*(\psi)$$

  - So the whole optimization is

$$\min_\theta \max_\psi \mathbb{E}_{p_\theta}[\psi(\boldsymbol{x})] - \mathbb{D}^*(\psi)$$

Parameterize ψ with an NN C_ϕ.
E.g., when $\mathbb{D}$ is JSD and
$\psi_\phi(\boldsymbol{x}) := 0.5 \log\left(1 - C_\phi\right) - 0.5 \log 2$

Plugging into the equation recovers vanilla GAN training

Petuum

Adversarial learning under SE – alternative interpretation

$$\min_{q,\theta} \; -\alpha \mathbb{H}(q) + \beta \mathbb{D}\Big(q(\boldsymbol{x}), p_\theta(\boldsymbol{x}) \Big) - \mathbb{E}_{q(\boldsymbol{x})}\Big[\; f(\boldsymbol{x}) \; \Big]$$

- Recall in MLE, f is a fixed function

$$f := f(\boldsymbol{x}\,;\,\mathcal{D}) = \log \mathbb{E}_{\boldsymbol{x}^* \sim \mathcal{D}} \left[\mathbb{1}_{\boldsymbol{x}^*}(\boldsymbol{x}) \right]$$

- Intuitively, see f as a similarity metric that measures similarity of sample $\boldsymbol{x}$ against real data $\mathcal{D}$
- Instead of the above manually fixed metric, can we learn a metric f_ϕ?

Adversarial learning under SE – alternative interpretation

- Augment the standard objective to account for ϕ:

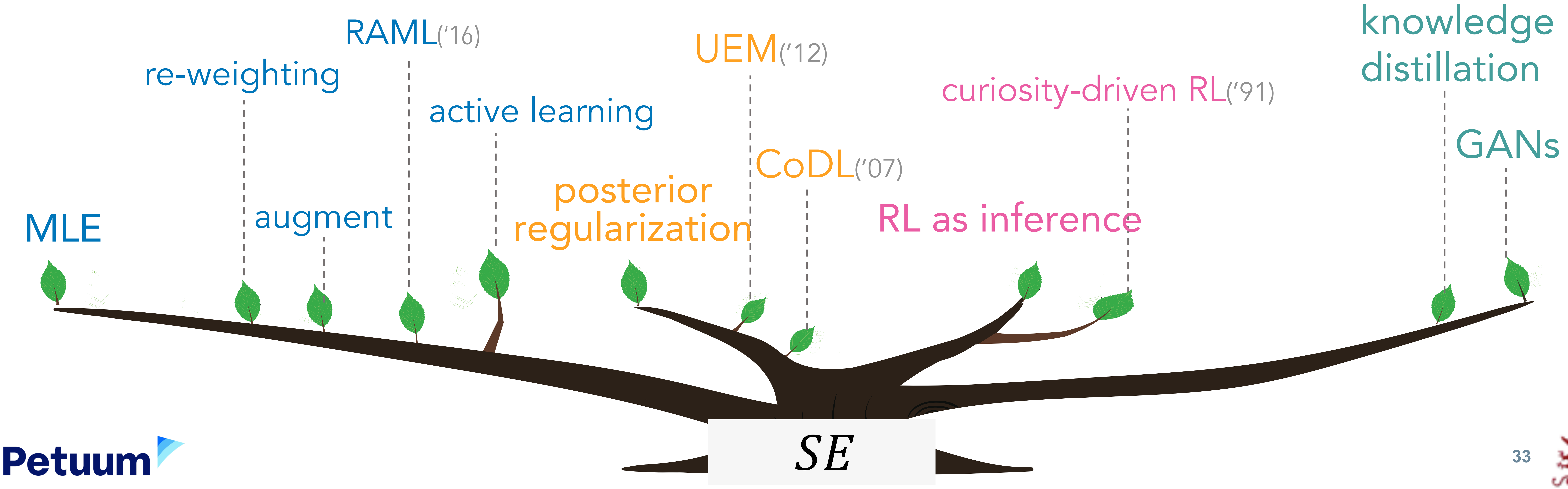
$$\min_{\theta} \max_{\phi} \min_{q} -\alpha \mathbb{H}(q) + \beta \mathbb{D}\Big(q(\boldsymbol{x}), p_{\theta}(\boldsymbol{x}) \Big) - \mathbb{E}_{q(\boldsymbol{x})}\Big[f_{\phi}(\boldsymbol{x}) \Big] + \mathbb{E}_{p_d(\boldsymbol{x})}\Big[f_{\phi}(\boldsymbol{x}) \Big]$$

- Set $\alpha = 0, \beta = 1$. Under mild conditions, the objective recovers:
  - Vanilla GAN [Goodfellow et al., 2014], when $\mathbb{D}$ is JS-divergence and f_{ϕ} is a binary classifier
  - f-GAN [Nowozin et al., 2016], when $\mathbb{D}$ is f-divergence
  - W-GAN [Arjovsky et al., 2017], when $\mathbb{D}$ is Wasserstein distance and f_{ϕ} is a 1-Lipschitz function

* *Proofs adapted from Farnia & Tse 2018: "A Convex Duality Framework for GANs"*

More algorithms recovered by SE

- Data augmentation / re-weighting / RAML
- Unified EM (UEM) / Constraint-driven learning (CoDL)
- Curiosity-driven RL
- Knowledge distillation



A table of ALL models/paradigms

Algorithm	f	α	β	$\mathbb{D}$
Unsupervised MLE	$f(\boldsymbol{x}; \mathcal{D})$	1	1	CE
Supervised MLE	$f(\boldsymbol{x}, \boldsymbol{y}; \mathcal{D})$	1	ϵ	CE
Active Learn.	$f(\boldsymbol{x}, \boldsymbol{y}; \mathcal{D}) + u(\boldsymbol{x})$	temp., > 0	ϵ	CE
Reward-augment MLE	$f_{\text{metric}}(\boldsymbol{x}, \boldsymbol{y}; \mathcal{D}, r)$	1	ϵ	CE
PG for Seq. Gen.	$f_{\text{metric}}(\boldsymbol{x}, \boldsymbol{y}; \mathcal{D}, r)$	1	1	CE
Posterior Reg.	$f_{rule}(\boldsymbol{x}, \boldsymbol{y})$	weight, > 0	α	CE
Unified EM	$f_{rule}(\boldsymbol{x}, \boldsymbol{y})$	weight, $\in \mathbb{R}$	1	CE
Policy Gradient (PG)	$\log Q^{ex}(\boldsymbol{x}, \boldsymbol{y})$	1	1	CE
+ Intrinsic Reward	$\log Q^{ex}(\boldsymbol{x}, \boldsymbol{y}) + Q^{in}(\boldsymbol{x}, \boldsymbol{y})$	1	1	CE
RL as inference	$Q^{ex}(\boldsymbol{x}, \boldsymbol{y})$	temp., > 0	α	CE
Vanilla GAN	binary classifier	0	1	JSD
f-GAN	discriminator	0	1	f-divg.
WGAN	1-Lipschitz discriminator	0	1	W dist.

Paradigms not (yet) covered by SE:

- Meta learning
- Lifelong learning
- …

Interesting future work to study the connections

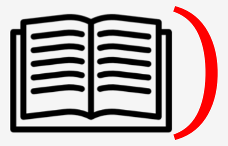
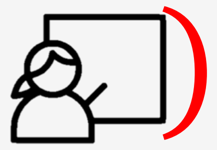
Learning with ALL experiences

- Distinct experiences are used in learning in the **same** way

↓

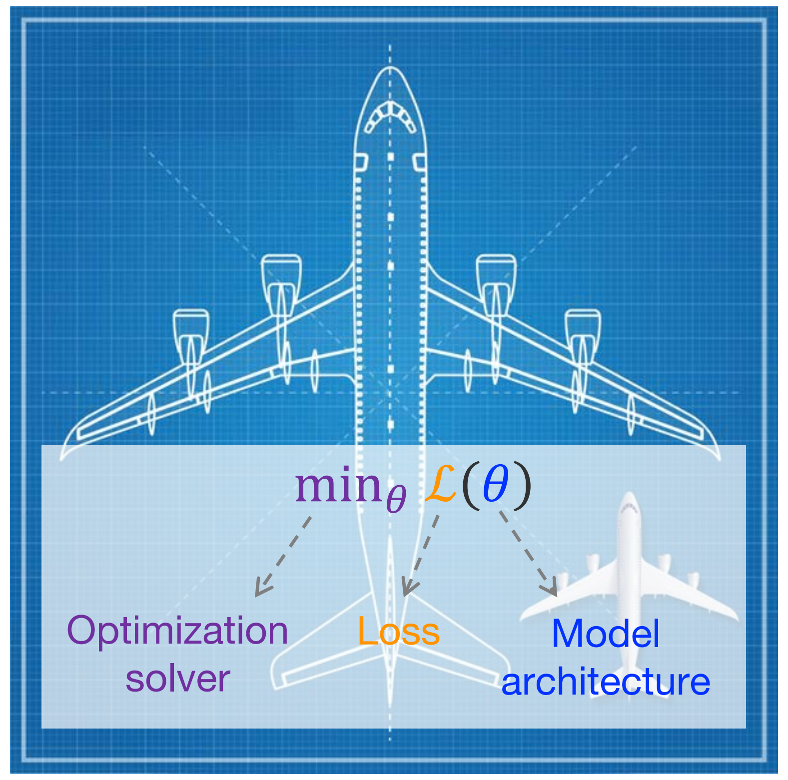
- Plug arbitrary available experiences into the learning procedure!

$$\mathcal{P}(f, \alpha, \beta)$$

$f = w_1 \cdot f(\boldsymbol{x} \mid$  $) + w_2 \cdot f(\boldsymbol{x} \mid$  $) + w_3 \cdot f(\boldsymbol{x} \mid$  $) + w_4 \cdot f(\boldsymbol{x} \mid$  $) + \cdots$

Focus on **what** to use, instead of worrying about **how** to use

The zoo of optimization solvers



$$\min_{q,\theta} -\alpha\mathbb{H}(q) + \beta\mathbb{D}\left(q(\boldsymbol{x}), p_\theta(\boldsymbol{x}) \right) - \mathbb{E}_{q(\boldsymbol{x})}\left[f(\boldsymbol{x}) \right]$$

Optimization of the loss, subject to $q \in \mathcal{P}_{\text{prob}}$.
Convex to q when $\alpha, \beta > 0$ and $\mathbb{D}$ is convex

- Like the Standard Equation as a *master loss* for many paradigms, is there a *master solver* for optimization of loss?
- No (yet) such a general algorithm
- Alternating GD:
  - Most widely used
  - EM, Variational EM (Variational inference), Wake-Sleep, …

Petuum

The extended EM as a primal solver

$$\min_{q,\theta} -\alpha\mathbb{H}(q) + \beta\mathbb{D}\Big(q(\boldsymbol{x}), p_\theta(\boldsymbol{x}) \Big) - \mathbb{E}_{q(\boldsymbol{x})}\Big[f(\boldsymbol{x};.) \Big]$$

when $\alpha, \beta > 0$ and $\mathbb{D} = \mathrm{CE}$

(1) reference in closed form:

$$q(\boldsymbol{x}) = \exp\left\{ \frac{\beta \log p_\theta(\boldsymbol{x}) + f(\boldsymbol{x};\ .)}{\alpha} \right\} / Z$$

(2) matching the model to the reference:

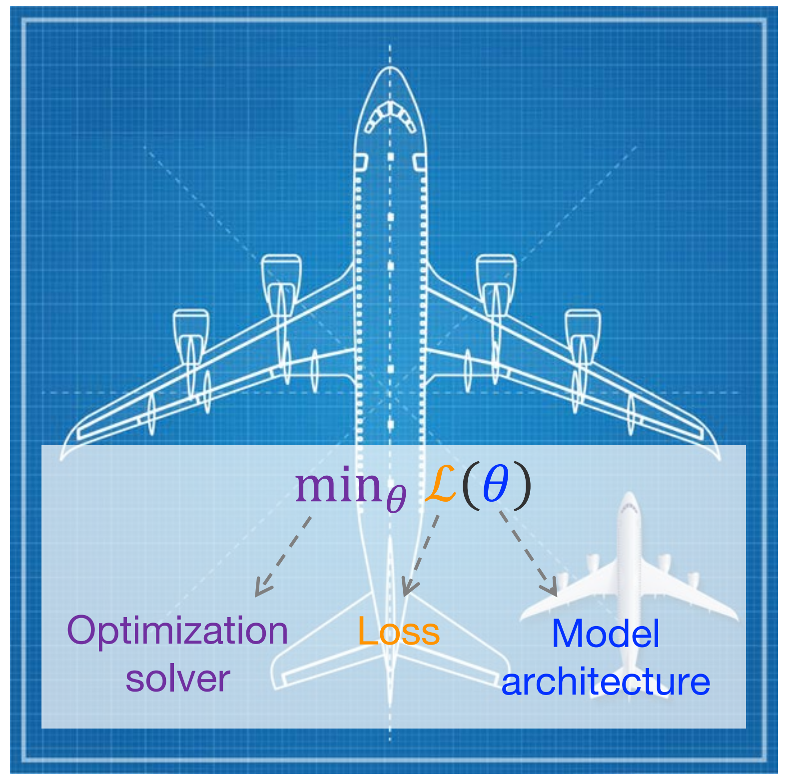
$$\min_\theta \mathbb{E}_{q(\boldsymbol{x})}\Big[\log p_\theta(\boldsymbol{x}) \Big]$$

Generalization of the classic Variational EM

- Generalized *E-step*
  Support all types of experiences (Teacher)
- *M-step*
  (Student)

- Limitations: e.g., not applicable when $\mathbb{D}$ is other divergence measures
- The EM as a template has been further enhanced/adapted in different ways in various paradigms
  - in RL: TRPO, PPO, MaxEnt inverse RL, …
  - in GANs: many extensions to stabilize training

Some "advanced" (specialized) techniques



$$\min_{q,\theta} -\alpha\mathbb{H}(q) + \beta\mathbb{D}\Big(q(\boldsymbol{x}), p_\theta(\boldsymbol{x}) \Big) - \mathbb{E}_{q(\boldsymbol{x})}\Big[f(\boldsymbol{x}) \Big]$$

Optimization of the loss, subject to $q \in \mathcal{P}_{\text{prob}}$.
Convex to q when $\alpha, \beta > 0$ and $\mathbb{D}$ is convex

- Alternating GD:
  - EM, Variational EM (Variational inference), Wake-Sleep, …
  - SGD, Back-propagation (BP)
- Convex duality, Lagrangian -- Kernel Tricks
- Integer linear programming (ILP)
- Probability functional descent (PFD) [Chu et al., 2019] -- Influence function, gives a neat formulation of GAN-like optimization and a few others

I: Duality

- Structured MaxEnt Discrimination (SMED) [Zhu and Xing, 2013]:

$$\min_{q,\,\xi\geq 0} \; -\alpha H(q) - \beta \mathbb{E}_q\left[\; \log p(\boldsymbol{\theta}) \;\right] + U(\xi)$$

$$s.t. \; -\mathbb{E}_q[\,\Delta F_i(\boldsymbol{y};\boldsymbol{\theta}) - \Delta \ell_i(\boldsymbol{y})\,] \leq \xi_i \qquad \forall i$$

  - Solve the (primal) Lagrangian:

$$q(\boldsymbol{\theta}) = \exp\left\{ \frac{\beta \log p(\boldsymbol{\theta}) + \sum_{i,\boldsymbol{y}\neq \boldsymbol{y}_i^*} \lambda_i(\boldsymbol{y})(\Delta F_i(\boldsymbol{y};\boldsymbol{\theta}) - \Delta \ell_i(\boldsymbol{y}))}{\alpha} \right\} / Z(\boldsymbol{\lambda})$$

  - Solve Lagrangian multipliers $\boldsymbol{\lambda}$ from the dual problem (when $p(\boldsymbol{\theta}) = \mathcal{N}(\boldsymbol{\theta}|0,I)$; $U(\xi) = \sum \xi_i$,)

$$\max_{\boldsymbol{\lambda}\geq 0,\, \sum \lambda_i = 1} \sum_{i,\boldsymbol{y}\neq \boldsymbol{y}_i^*} \lambda_i(\boldsymbol{y}) \Delta \ell_i(\boldsymbol{y}) - \frac{1}{2} \left| \sum_{i,\boldsymbol{y}\neq \boldsymbol{y}_i^*} \lambda_i(\boldsymbol{y}) \Delta T_i(\boldsymbol{y}) \right|^2$$

Allows kernel trick for nonlinear interactions b/w experiences

II: Influence Function and Probability Functional Descent

- Gradient descent in the space of probability measures $\mathcal{P}(X)$

$$\min_{p \in \mathcal{P}(X)} \mathcal{J}(p) \qquad \mathcal{J}: \mathcal{P}(X) \to \mathbb{R} \text{ : a probability functional}$$

- Influence function $\Psi_p(x)$:

Gateaux differential of $\mathcal{J}$ at p in the direction $\chi = q - p$

$$d\mathcal{J}_p(\chi) = \int_X \Psi_p(x)\chi(dx)$$
$$= \mathbb{E}_q[\Psi_p(x)] - \mathbb{E}_p[\Psi_p(x)]$$

- With a linear approximation $\tilde{\mathcal{J}}(p)$ to $\mathcal{J}(p)$ around p_0:

$$\tilde{\mathcal{J}}(p) = \mathcal{J}(p_0) + d\mathcal{J}_{p_t}(p - p_0) = \mathbb{E}_{x\sim p}\left[\, \Psi_{p_0}(x)\, \right] + const.$$

- Thus, once we obtain the influence function, we can optimize p by decreasing $\mathbb{E}_{x\sim p}\left[\, \Psi_{p_0}(x)\, \right]$

[Chu et al., 2019]

Adversarial learning using PFD

$$\mathcal{J}(p_\theta) = \mathbb{D}\left(\; p_d(\boldsymbol{x}), p_\theta(\boldsymbol{x}) \; \right)$$

- Often no closed-form influence function, e.g., when $\mathbb{D}$ is JSD or W-distance
- Approximate with convex duality:
  - Convex conjugate $\mathcal{J}^*(\psi) = \sup_u \int_x \psi(\boldsymbol{x}) u(dx) - \mathcal{J}(u)$
  - Influence function is obtained via $\Psi_{p_\theta}(x) = \mathrm{argmax}_\psi \, \mathbb{E}_{\boldsymbol{x} \sim p_\theta}[\psi(\boldsymbol{x})] - \mathcal{J}^*(\psi)$
  - Parameterize ψ as below to recover optimization of generator and discriminator

$$\psi_{\boldsymbol{\phi}}(\boldsymbol{x}) := 0.5 \log\left(1 - C_\phi\right) - 0.5 \log 2$$

$$\Psi_{JS} = \mathrm{argmax}_\phi \, \mathbb{E}_{p_{data}}\left[\log C_\phi\right] - \mathbb{E}_{p_\theta}\left[\log\left(1 - C_\phi\right)\right]$$

- The whole optimization of $\mathcal{J}(p)$ is thus

$$\min_\theta \max_\psi \mathbb{E}_{p_{data}}\left[\log C_\phi\right] - \mathbb{E}_{p_\theta}\left[\log\left(1 - C_\phi\right)\right]$$

[Chu et al., 2019]

RL using PFD

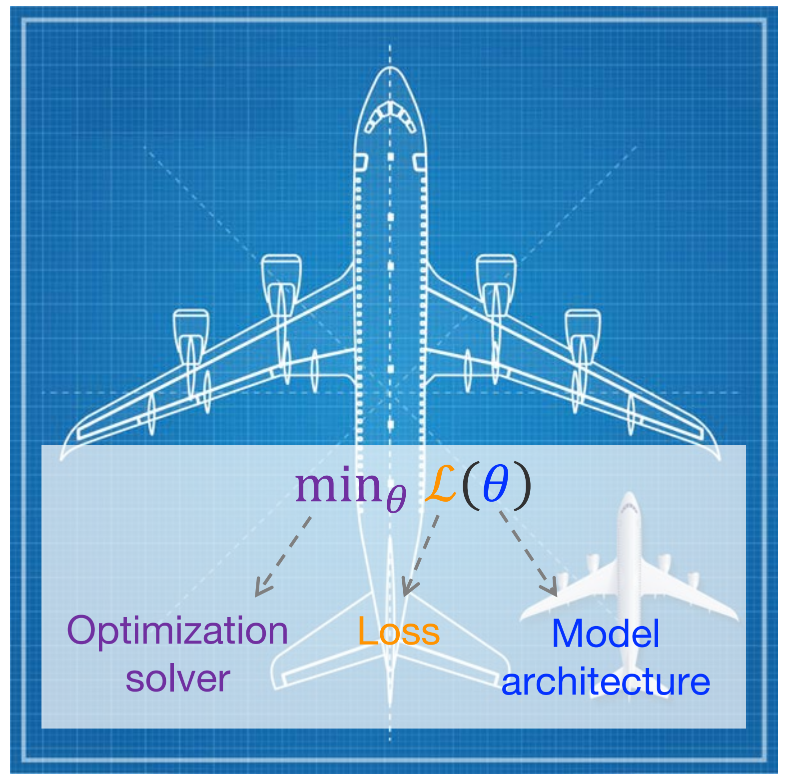
- E.g., Policy iteration in RL
  - (Conventional) loss: $\mathcal{J}(p_\theta) = -\mathbb{E}_{p_d(\boldsymbol{x})}\mathbb{E}_{p_\theta(\boldsymbol{y}|\boldsymbol{x})}[\,Q(\boldsymbol{x},\boldsymbol{y})\,]$

    $p_d(\boldsymbol{x})$ – state distribution; $p_\theta(\boldsymbol{y}|\boldsymbol{x})$ – policy

  - Influence function $\Psi_{p_\theta^t}(\boldsymbol{y}) = -\mathbb{E}_{p_d(\boldsymbol{x})}[\,Q(\boldsymbol{x},\boldsymbol{y})\,]$

  - Thus, optimize p_θ by minimizing

$$\mathbb{E}_{p_\theta}\left[\Psi_{p_\theta^t}(\boldsymbol{y})\right] = -\mathbb{E}_{p_d(\boldsymbol{x})}\mathbb{E}_{p_\theta(\boldsymbol{y}|\boldsymbol{x})}[\,Q(\boldsymbol{x},\boldsymbol{y})\,]$$



Model architecture

- Relatively well explored:
  - Neural network design
  - Graphical model design
  - Compositional architectures

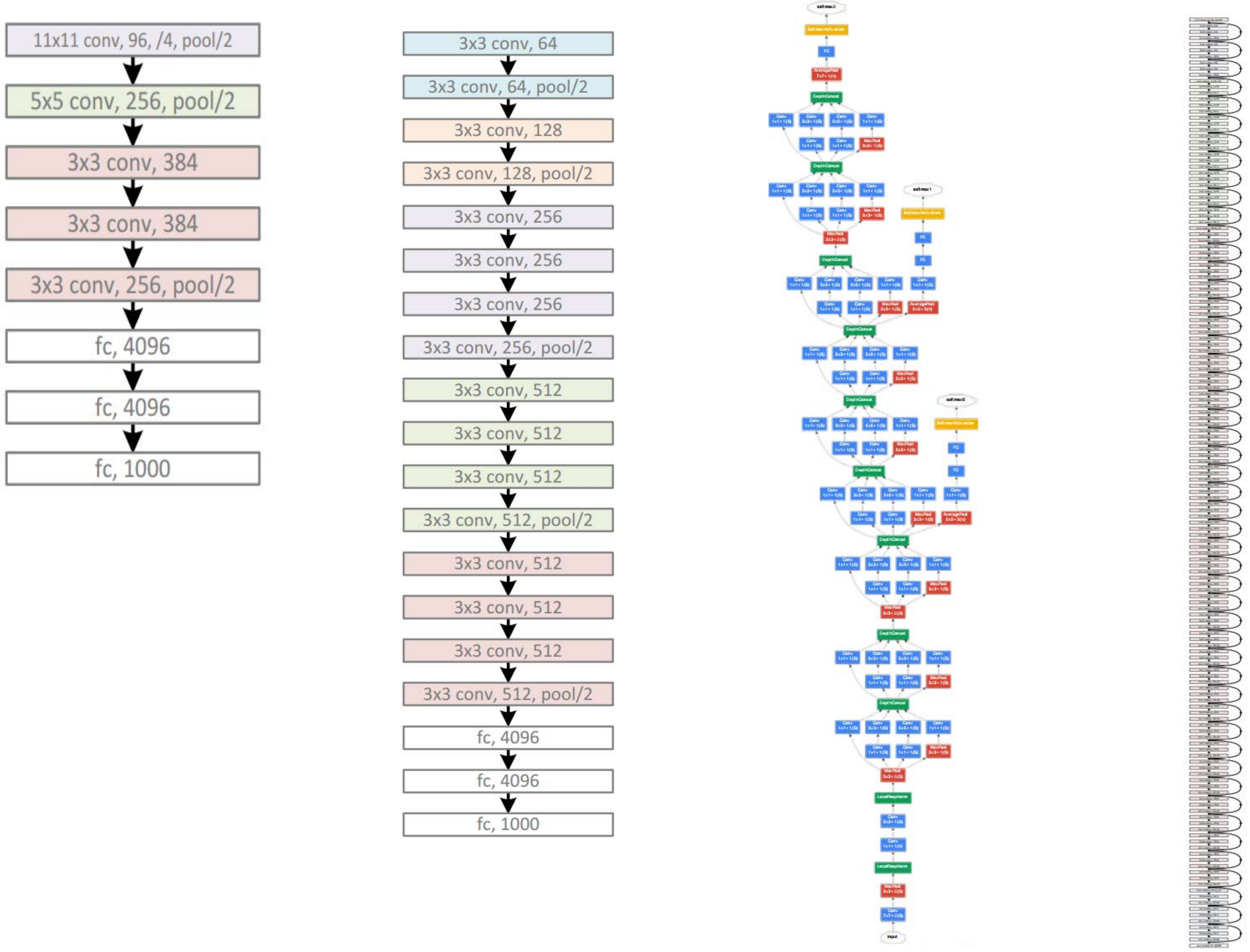
$$\min_{q,\theta} -\alpha\mathbb{H}(q) + \beta\mathbb{D}\left(q(\boldsymbol{x}), p_{\theta}(\boldsymbol{x}) \right) - \mathbb{E}_{q(\boldsymbol{x})}\left[f(\boldsymbol{x}) \right]$$

Model architecture

- Relatively well explored:
  - Neural network design
  - Graphical model design
  - Compositional architectures

- Activation functions
  - Linear and ReLU
  - Sigmoid and tanh
  - Etc.
- Layers
  - Fully connected
  - Convolutional & pooling
  - Recurrent
  - ResNets
  - Etc.

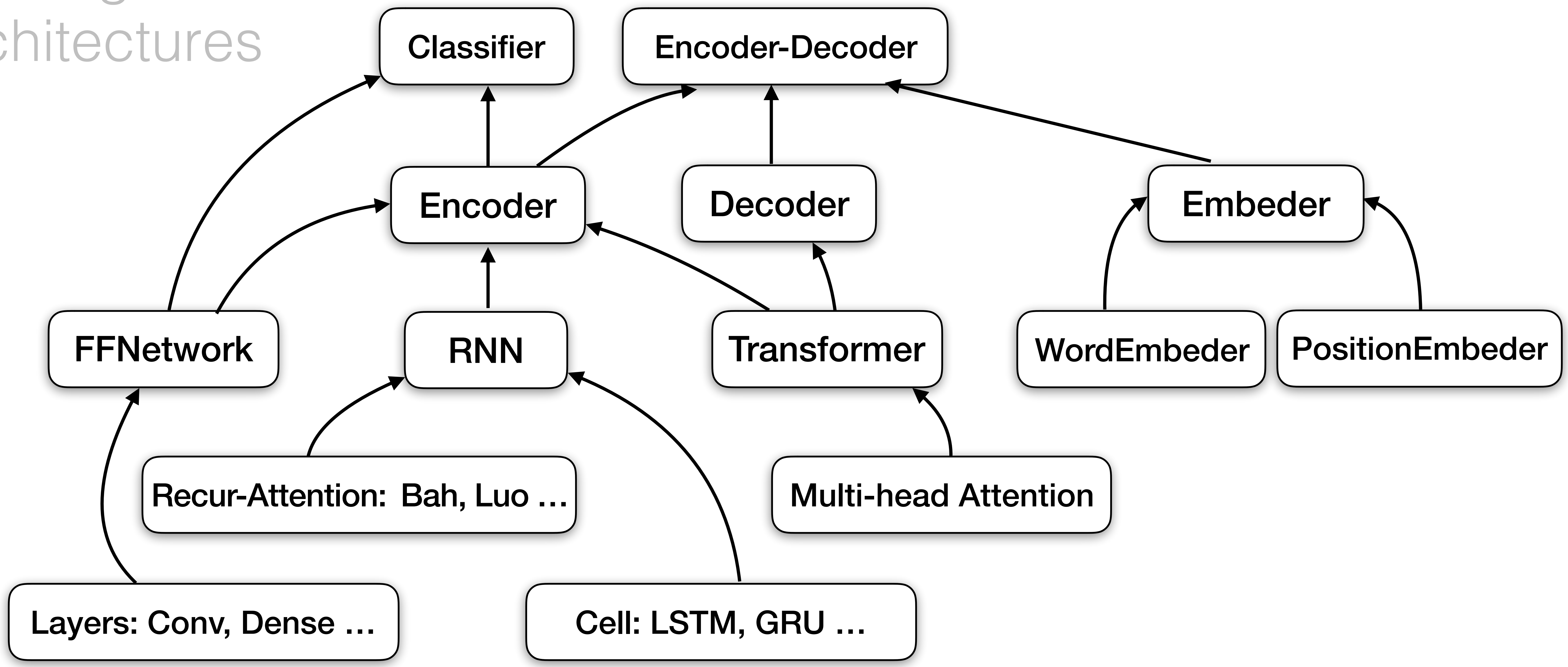
AlexNet 8 layers | VGG 19 layers | GoogleNet 22 layers | ResNet 152 layers



Model architecture

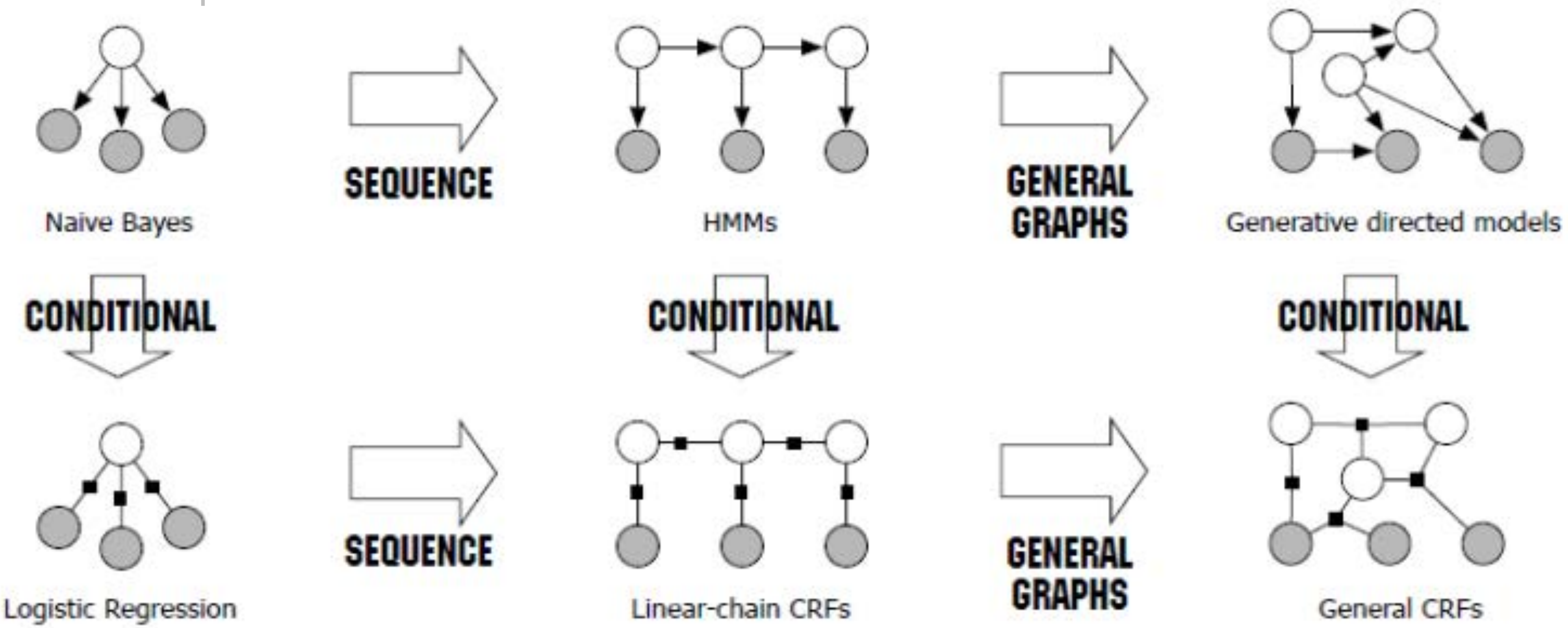
- Relatively well explored:
  - Neural network design
  - Graphical model design
  - Compositional architectures

Neural network components



Model architecture

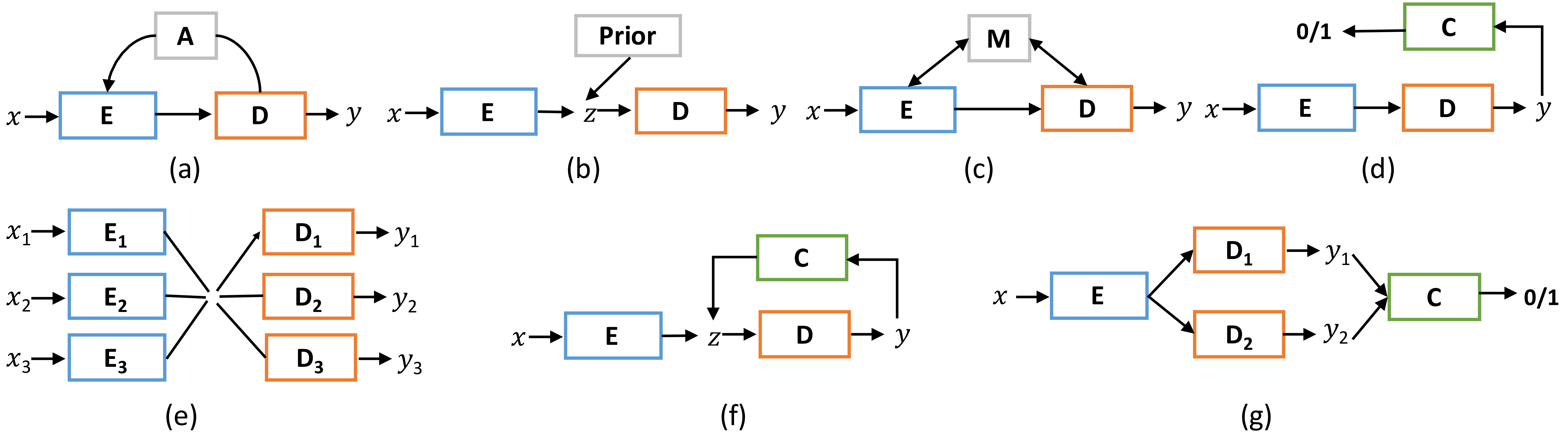
- Relatively well explored:
  - Neural network design
  - Graphical model design
  - Compositional architectures



[Courtesy: Sutton & McCallum, 2010]

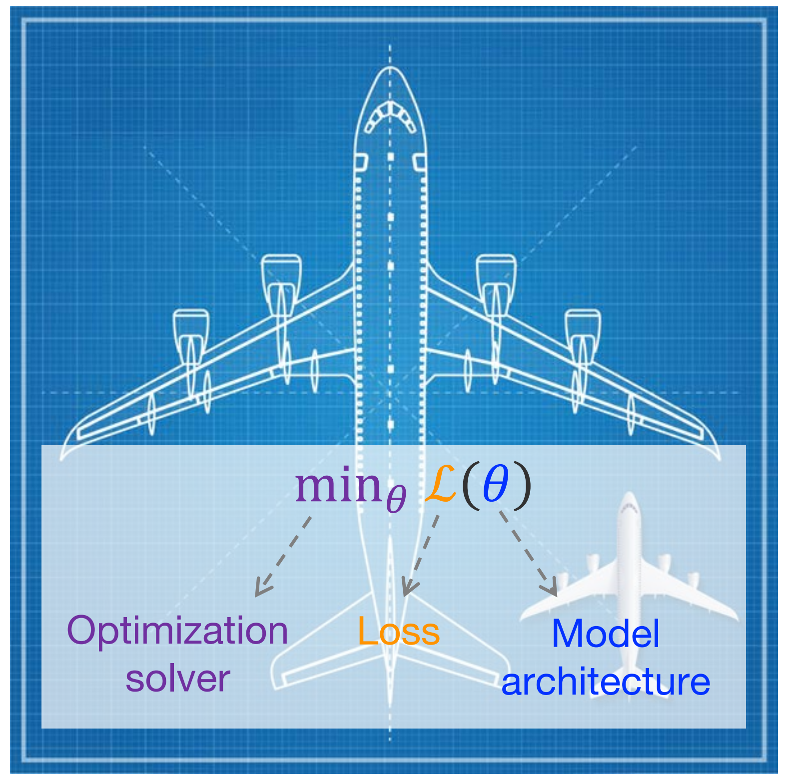
Model architecture

- Relatively well explored:
  - Neural network design
  - Graphical model design
  - Compositional architectures



E refers to encoder, D to decoder, C to Classifier, A to attention, Prior to prior distribution, and M to memory

Summary: a blueprint of ML



- Loss
  - Standard equation

$$\min_{q,\theta} -\mathbb{E}_{q(\boldsymbol{x},\boldsymbol{y})}\left[f(\boldsymbol{x},\boldsymbol{y})\right] + \alpha\mathbb{D}\left(q(\boldsymbol{x},\boldsymbol{y}), p_\theta(\boldsymbol{x},\boldsymbol{y})\right) - \beta\mathbb{H}(q)$$

- Algorithm
  - The extended EM algorithm gives a general primal solution in many cases
  - PFD gives a neat formulation for some cases (e.g., GANs)
- Model architecture: vast library of building blocks → compositionality

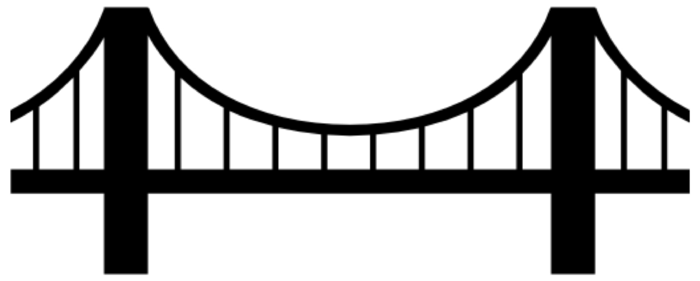
Next: practical implications of the ML blueprint

Why this is useful?

- Learning with ALL experiences
- Complex interaction between experiences
- Multi-agent game theoretic learning using all experiences

Learning with ALL experiences: *Empowering algorithms*

- Unifying perspective of diverse paradigms (each tailored for a specific type of experience) under SE



- Combining or integrating different experiences
- Re-use or repurpose originally specialized algorithms
  - Systematic idea transfer and solution exchange
  - Solving challenges in one paradigm by applying well-known solutions from another
  - Accelerate innovations across research areas

Petuum

Learning with ALL experiences:
Empowering algorithms – Ex.1

- Rules in PR ⇔ Reward in RL
- Empower **reward learning** algo. to **learning rules** [Hu et al., 2018]

Algorithm	f	α	β	$\mathbb{D}$
Unsupervised MLE	$f(\boldsymbol{x}; \mathcal{D})$	1	1	CE
Supervised MLE	$f(\boldsymbol{x}, \boldsymbol{y}; \mathcal{D})$	1	ϵ	CE
Active Learn.	$f(\boldsymbol{x}, \boldsymbol{y}; \mathcal{D}) + u(\boldsymbol{x})$	temp., > 0	ϵ	CE
Reward-augment MLE	$f_{\text{metric}}(\boldsymbol{x}, \boldsymbol{y}; \mathcal{D}, r)$	1	ϵ	CE
PG for Seq. Gen.	$f_{\text{metric}}(\boldsymbol{x}, \boldsymbol{y}; \mathcal{D}, r)$	1	1	CE
Posterior Reg.	$f_{rule}(\boldsymbol{x}, \boldsymbol{y})$	weight, > 0	α	CE
Unified EM	$f_{rule}(\boldsymbol{x}, \boldsymbol{y})$	weight, $\in \mathbb{R}$	1	CE
Policy Gradient (PG)	$\log Q^{ex}(\boldsymbol{x}, \boldsymbol{y})$	1	1	CE
+ Intrinsic Reward	$\log Q^{ex}(\boldsymbol{x}, \boldsymbol{y}) + Q^{in}(\boldsymbol{x}, \boldsymbol{y})$	1	1	CE
RL as inference	$Q^{ex}(\boldsymbol{x}, \boldsymbol{y})$	temp., > 0	α	CE
Vanilla GAN	binary classifier	0	1	JSD
f-GAN	discriminator	0	1	f-divg.
WGAN	1-Lipschitz discriminator	0	1	W dist.

Petuum

Learning with ALL experiences: *Empowering algorithms – Ex.2*

- Data in supervised MLE ⇔ Reward in RL
- Empower **reward learning** algo. to **learning data augmentation** [Hu et al., 2019]

Algorithm	f	α	β	$\mathbb{D}$
Unsupervised MLE	$f(\boldsymbol{x}; \mathcal{D})$	1	1	CE
Supervised MLE	$f(\boldsymbol{x}, \boldsymbol{y}; \mathcal{D})$	1	ϵ	CE
Active Learn.	$f(\boldsymbol{x}, \boldsymbol{y}; \mathcal{D}) + u(\boldsymbol{x})$	temp., > 0	ϵ	CE
Reward-augment MLE	$f_{\text{metric}}(\boldsymbol{x}, \boldsymbol{y}; \mathcal{D}, r)$	1	ϵ	CE
PG for Seq. Gen.	$f_{\text{metric}}(\boldsymbol{x}, \boldsymbol{y}; \mathcal{D}, r)$	1	1	CE
Posterior Reg.	$f_{rule}(\boldsymbol{x}, \boldsymbol{y})$	weight, > 0	α	CE
Unified EM	$f_{rule}(\boldsymbol{x}, \boldsymbol{y})$	weight, $\in \mathbb{R}$	1	CE
Policy Gradient (PG)	$\log Q^{ex}(\boldsymbol{x}, \boldsymbol{y})$	1	1	CE
+ Intrinsic Reward	$\log Q^{ex}(\boldsymbol{x}, \boldsymbol{y}) + Q^{in}(\boldsymbol{x}, \boldsymbol{y})$	1	1	CE
RL as inference	$Q^{ex}(\boldsymbol{x}, \boldsymbol{y})$	temp., > 0	α	CE
Vanilla GAN	binary classifier	0	1	JSD
f-GAN	discriminator	0	1	f-divg.
WGAN	1-Lipschitz discriminator	0	1	W dist.

Petuum

Learning with ALL experiences:
Empowering algorithms – Ex.3

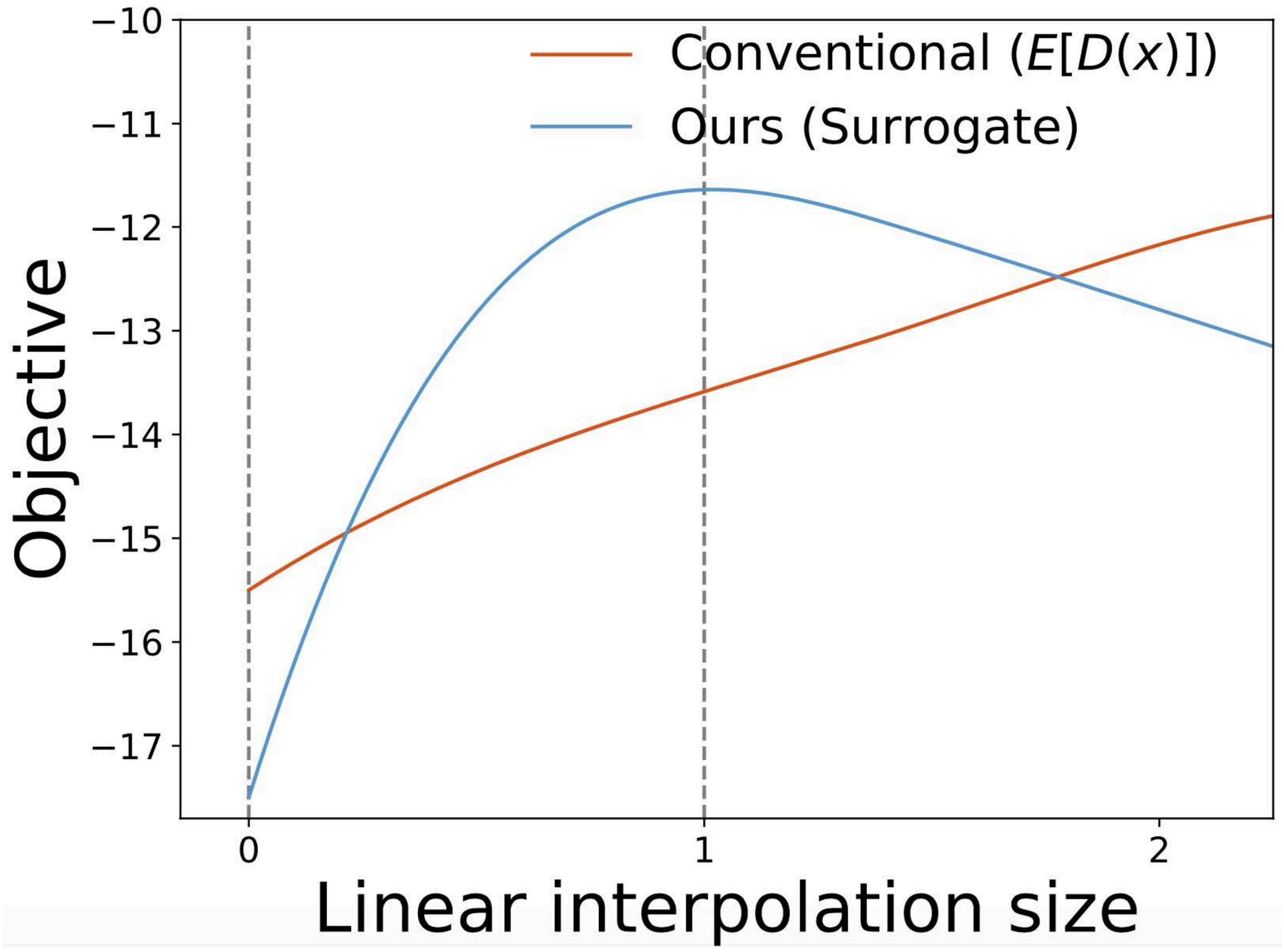
- GANs ⇔ RL ⇔ VI
- Empower **RL/VI** algo. (e.g., PPO) to **stabilize GAN training** [Wu et al., 2020]

Algorithm	f	α	β	$\mathbb{D}$
Unsupervised MLE	$f(\boldsymbol{x}; \mathcal{D})$	1	1	CE
Supervised MLE	$f(\boldsymbol{x}, \boldsymbol{y}; \mathcal{D})$	1	ϵ	CE
Active Learn.	$f(\boldsymbol{x}, \boldsymbol{y}; \mathcal{D}) + u(\boldsymbol{x})$	temp., > 0	ϵ	CE
Reward-augment MLE	$f_{\text{metric}}(\boldsymbol{x}, \boldsymbol{y}; \mathcal{D}, r)$	1	ϵ	CE
PG for Seq. Gen.	$f_{\text{metric}}(\boldsymbol{x}, \boldsymbol{y}; \mathcal{D}, r)$	1	1	CE
Posterior Reg.	$f_{rule}(\boldsymbol{x}, \boldsymbol{y})$	weight, > 0	α	CE
Unified EM	$f_{rule}(\boldsymbol{x}, \boldsymbol{y})$	weight, $\in \mathbb{R}$	1	CE
Policy Gradient (PG)	$\log Q^{ex}(\boldsymbol{x}, \boldsymbol{y})$	1	1	CE
+ Intrinsic Reward	$\log Q^{ex}(\boldsymbol{x}, \boldsymbol{y}) + Q^{in}(\boldsymbol{x}, \boldsymbol{y})$	1	1	CE
RL as inference	$Q^{ex}(\boldsymbol{x}, \boldsymbol{y})$	temp., > 0	α	CE
Vanilla GAN	binary classifier	0	1	JSD
f-GAN	discriminator	0	1	f-divg.
WGAN	1-Lipschitz discriminator	0	1	W dist.

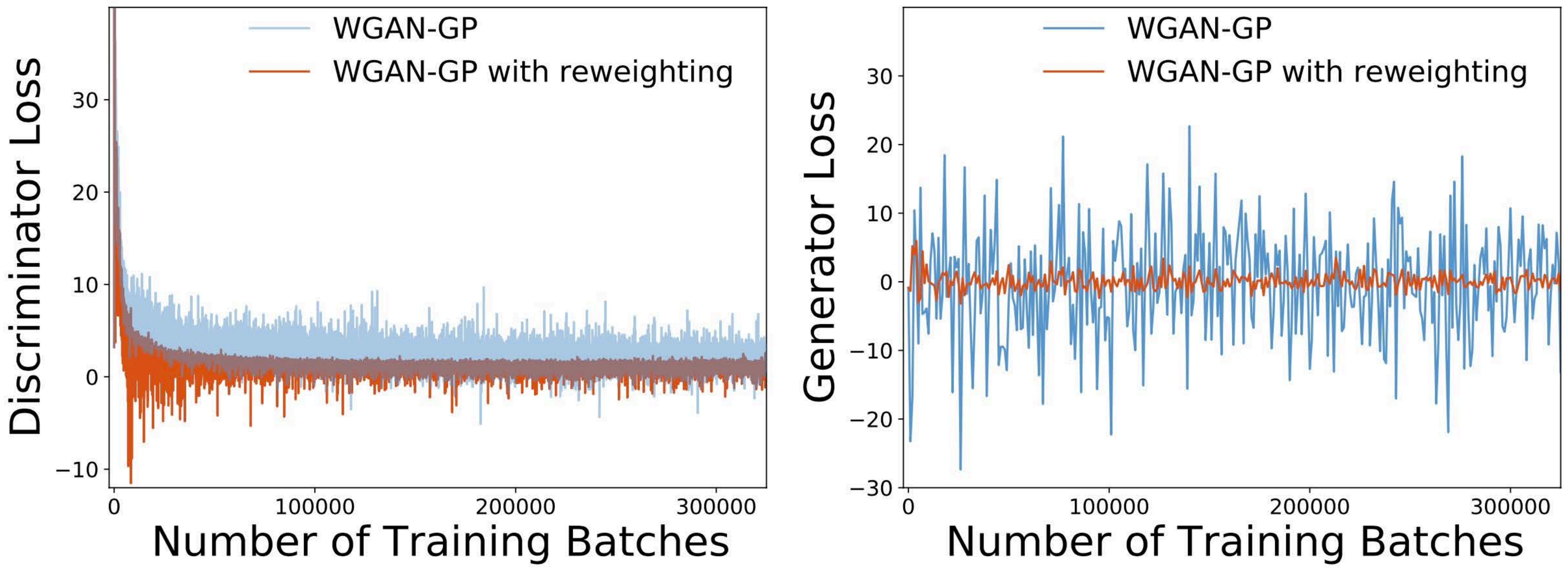
Petuum

Learning with ALL experiences:
Empowering algorithms – Ex.3

- GANs ⇔ RL ⇔ VI
- Empower **RL/VI** algo. (e.g., PPO) to **stabilize GAN training** [Wu et al., 2020]



(a) Re-use PPO objective for GAN training: discourage excessively large updates by "trapping" the update size around 1



(b) Re-use importance weighting in a VI perspective: greatly reduced variance in both generator and discriminator losses

Improved performance on a range of problems, including image generation, text generation, and text style transfer

Petuum

Learning with ALL experiences:
Experience compositionality – Ex.1

- Distinct experiences are all modeled with $f(\boldsymbol{x}, \boldsymbol{y})$
- Combine and plug different f functions into SE to drive learning

$$\min_{q,\theta} - \mathbb{E}_{q(\boldsymbol{x},\boldsymbol{y})}\left[f(\boldsymbol{x},\boldsymbol{y})\right] + \alpha \mathbb{D}\left(q(\boldsymbol{x},\boldsymbol{y}), p_\theta(\boldsymbol{x},\boldsymbol{y})\right) - \beta \mathbb{H}(q)$$

$$f(\boldsymbol{x},\boldsymbol{y}) = w_1 \cdot f_{data} + w_2 \cdot f_{rules} + w_3 \cdot f_{reward} + \cdots$$

- Enable applications for controllable content generation

Controllable text generation

f = sentiment classifier
+ linguistic rules
+ language model

Controlling sentiment

Pos	*The film is full of imagination!*
Neg	*The film is strictly routine!*

Petuum

[Hu et al., 2017; Yang et al., 2018]

Learning with ALL experiences:
Experience compositionality – Ex.2

- Distinct experiences are all modeled with $f(\boldsymbol{x}, \boldsymbol{y})$
- Combine and plug different f functions into SE to drive learning

$$\min_{q,\theta} -\mathbb{E}_{q(\boldsymbol{x},\boldsymbol{y})}\left[f(\boldsymbol{x},\boldsymbol{y}) \right] + \alpha \mathbb{D}\left(q(\boldsymbol{x},\boldsymbol{y}),\, p_\theta(\boldsymbol{x},\boldsymbol{y}) \right) - \beta \mathbb{H}(q)$$

$$f(\boldsymbol{x},\boldsymbol{y}) = w_1 \cdot f_{data} + w_2 \cdot f_{rules} + w_3 \cdot f_{reward} + \cdots$$

- Enable applications for controllable content generation

Fashion image generation

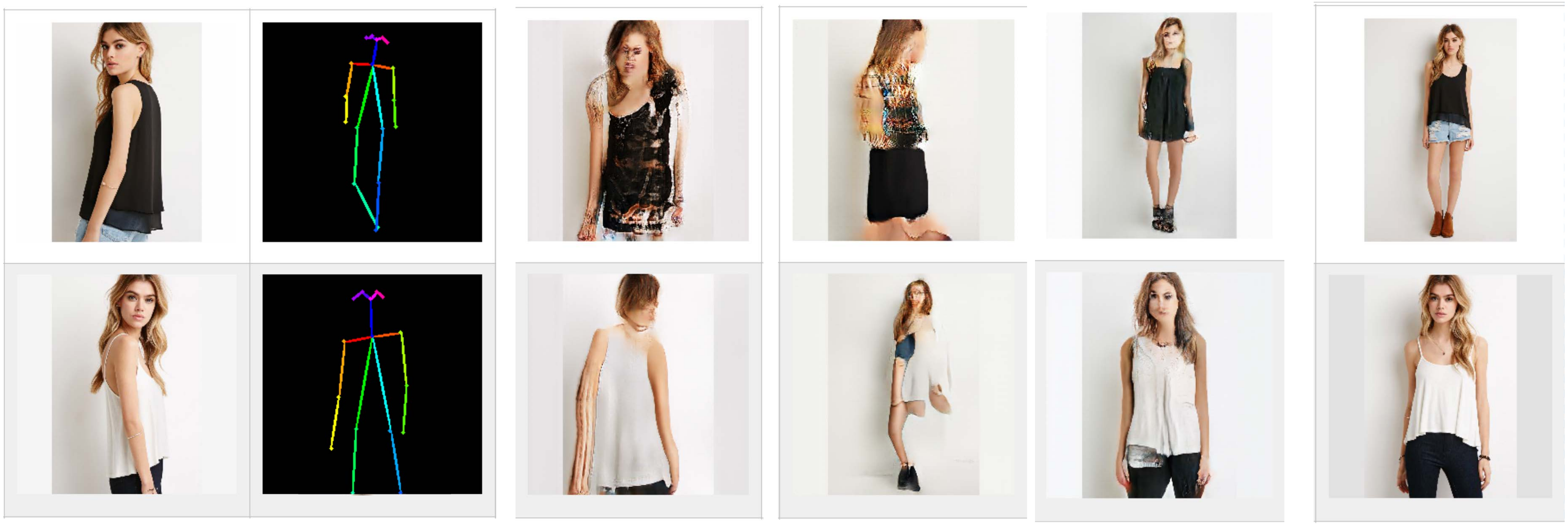
f = (small) data
+ human gesture constraints

Source

Generated images under different poses

Learning with ALL experiences:
Experience compositionality – Ex.2

source | *target pose* | Base model | + Fixed knowledge | **+ Learned knowledge (Ours)** | *true target*



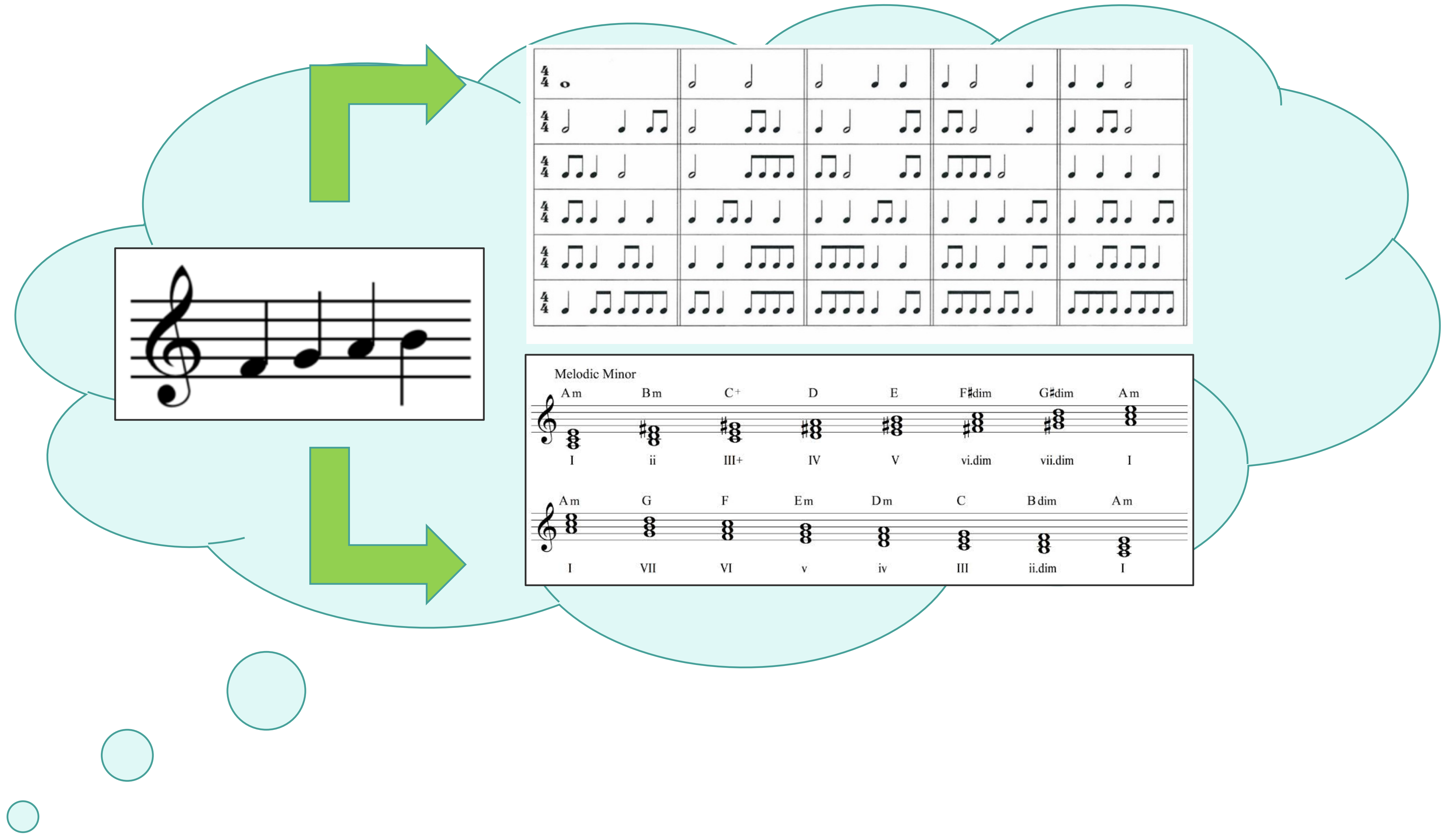
Petuum

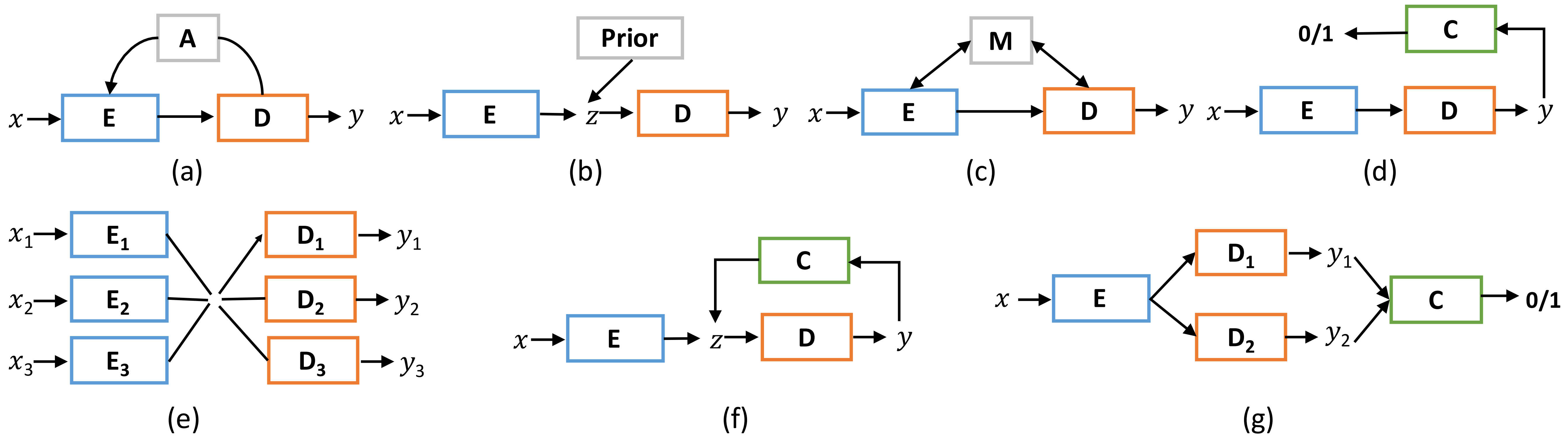
Operational compositionality

- Build ML applications like composing music

$$\min_{\theta} \mathcal{L}(\theta)$$

optimization Loss Model architecture

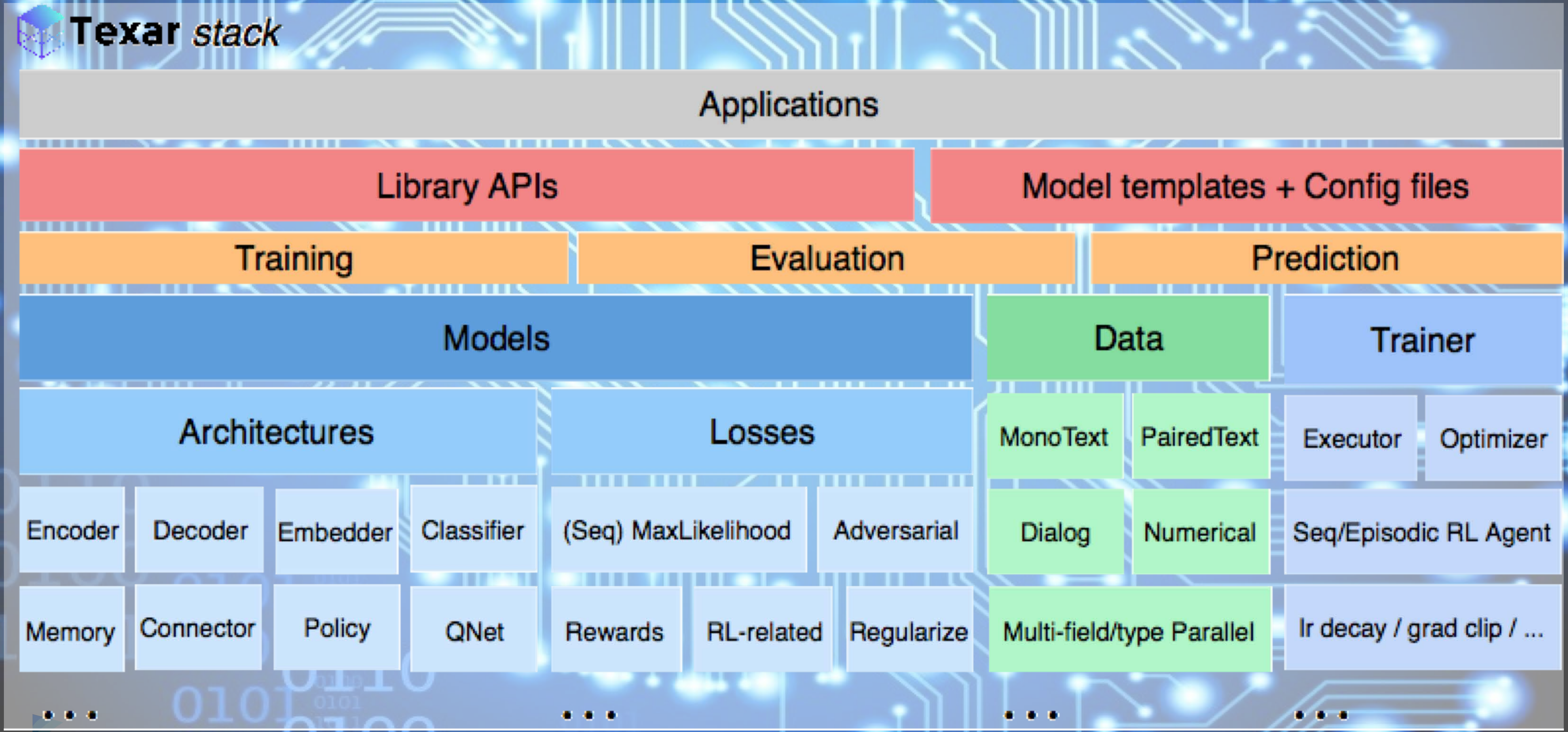




Texar

Open-source toolkit for composable ML

Texar Stack – Operationalized "View" of Composable ML

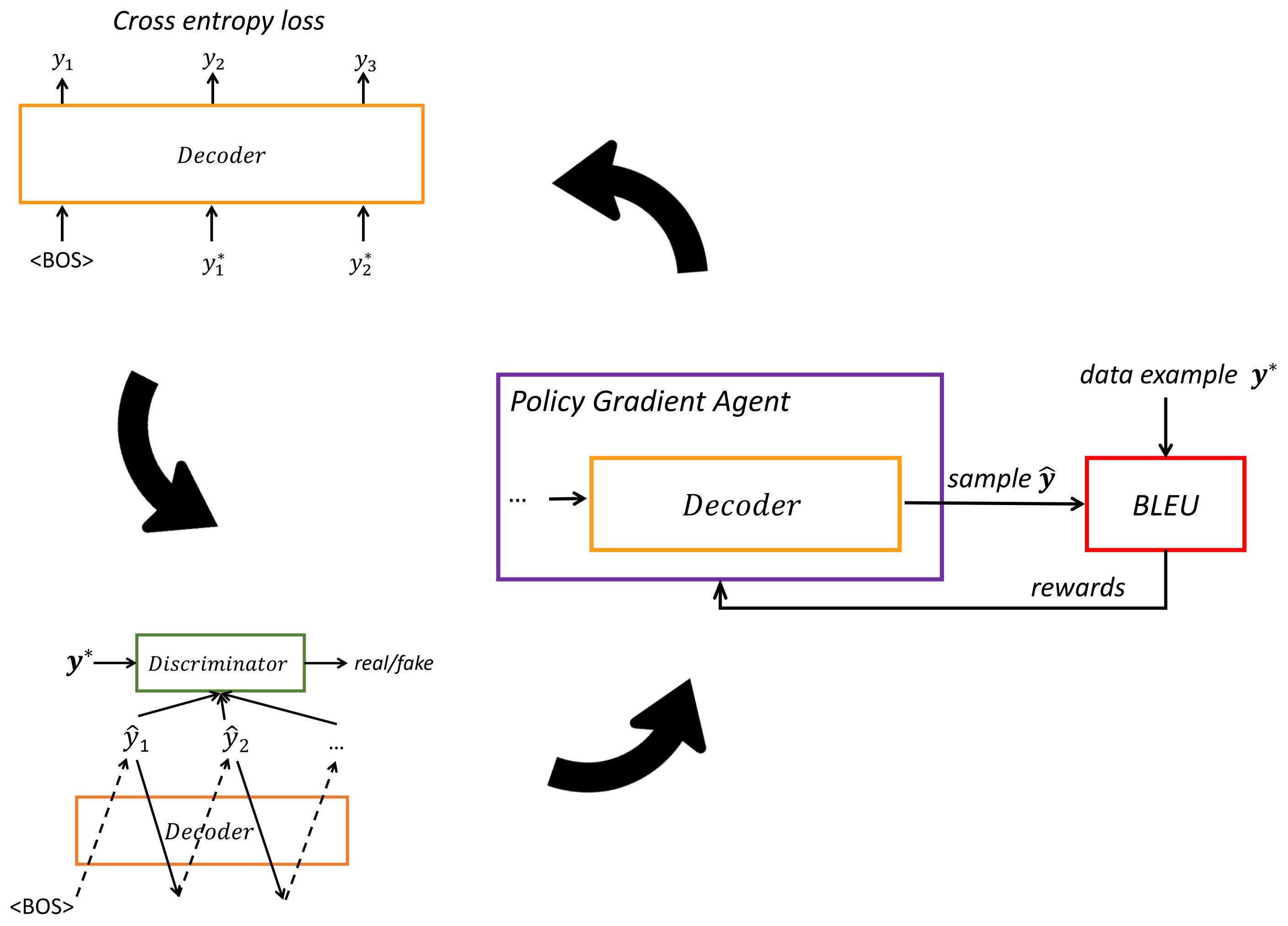


Composable ML with Texar

- Highly modularized programming
  - Data, structure, loss, learning, …
  - Intuitive conceptual-level APIs

```
# Read data
dataset = PairedTextData(data_hparams)
batch = DataIterator(dataset).get_next()
# Encode
embedder = WordEmbedder(dataset.vocab.size, hparams=embedder_hparams)
encoder = TransformerEncoder(hparams=encoder_hparams)
enc_outputs = encoder(embedder(batch['source_text_ids']),
                      batch['source_length'])
# Build decoder
decoder = AttentionRNNDecoder(memory=enc_outputs,
                              hparams=decoder_hparams)
# Maximum Likelihood Estimation
## Teacher-forcing decoding
outputs, length, _ = decoder(decoding_strategy='teacher-forcing',
                             inputs=embedder(batch['target_text_ids']),
                             seq_length=batch['target_length']-1)
## Cross-entropy loss
loss = sequence_sparse_softmax_cross_entropy(
    labels=batch['target_text_ids'][:,1:], logits=outputs.logits, seq_length=length)
```

- Easy switch between learning algorithms
  - Plug in & out modules
  - No changes to irrelevant parts



Petuum

Food for thoughts: How far would this take us?

- Physics

It is only slightly overstating the case to say that ***physics is the study of symmetry****.*

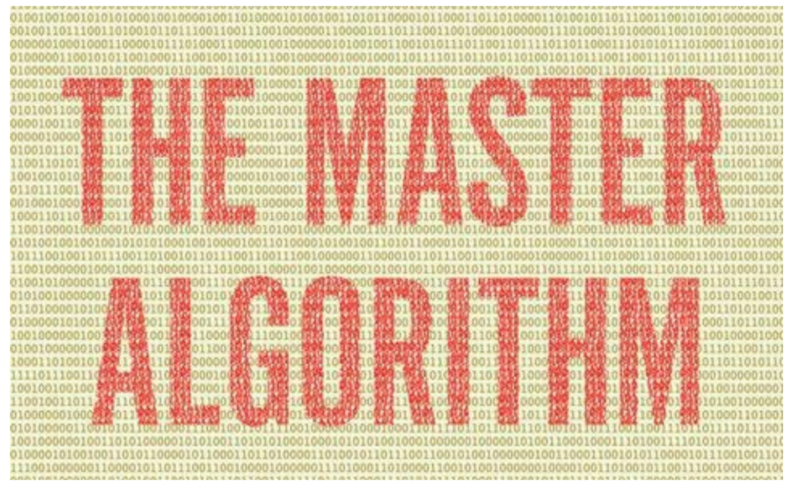
-- Phil Anderson (1923-2020), Physicist, Nobel laureate

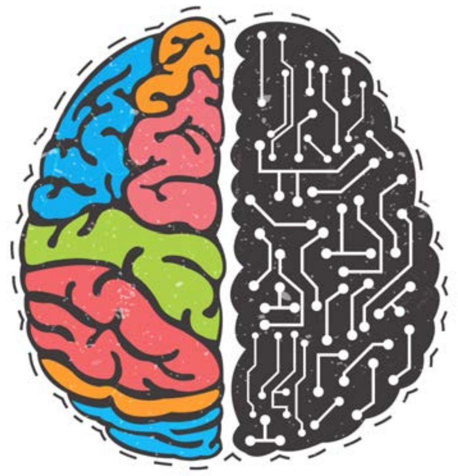
Food for thoughts: How far would this take us?

- Physics

Maxwell's equations	General relativity	Standard model	? Theory of everything
1861	1910s	1970s	

- Machine Learning





Unified way of thinking
- ✦ Systematic understanding
- ✦ Automated solution creation
- ✦ Improved ML accessibility

Toward unified theoretical analysis

- How do we characterize learning with different experiences?
  - E.g., data examples, rules, reward, auxiliary models (discriminators), …
  - Combinations of above experiences
- What's the appropriate statistical tool to characterize learning with logical rules? Can we guarantee performance improvement when using more experiences? What if experiences are noisy?
- A possible direction:
  - Existing theoretical analyses deal with learning with data examples, online learning, reinforcement learning, .. in silos
  - With the standard equation, can we re-purpose the analyses to other paradigms, e.g., learning with logical rules?

References

[1] Jun Zhu, Ning Chen, and Eric P Xing. 2014. Bayesian inference with posterior regularization and applications to infinite latent SVMs. JMLR(2014).

[2] Jun Zhu and Eric P Xing. 2009. Maximum Entropy Discrimination Markov Networks. JMLR(2009).

[3] Zhiting Hu, Xuezhe Ma, Zhengzhong Liu, Eduard Hovy, and Eric Xing. 2016.Harnessing deep neural networks with logic rules. In ACL.

[4] Zhiting Hu, Haoran Shi, Bowen Tan, Wentao Wang, Zichao Yang, Tiancheng Zhao, Junxian He, Lianhui Qin, Di Wang, Xuezhe Ma, et al. 2019. Texar: A modularized, versatile, and extensible toolkit for text generation. ACL(2019).

[5] Zhiting Hu, Bowen Tan, Russ R Salakhutdinov, Tom M Mitchell, and Eric P Xing. 2019. Learning data manipulation for augmentation and weighting. In NeurIPS.

[6] Zhiting Hu, Zichao Yang, Xiaodan Liang, Ruslan Salakhutdinov, and Eric P Xing. 2017. Toward controlled generation of text. In ICML.

[7] Zhiting Hu, Zichao Yang, Ruslan Salakhutdinov, and Eric P Xing. 2018. On Unifying Deep Generative Models. In ICLR.

[8] Zhiting Hu, Zichao Yang, Russ R Salakhutdinov, Lianhui Qin, Xiaodan Liang, Haoye Dong, and Eric P Xing. 2018. Deep generative models with learnable knowledge constraints. In NeurIPS.

64

Thanks!