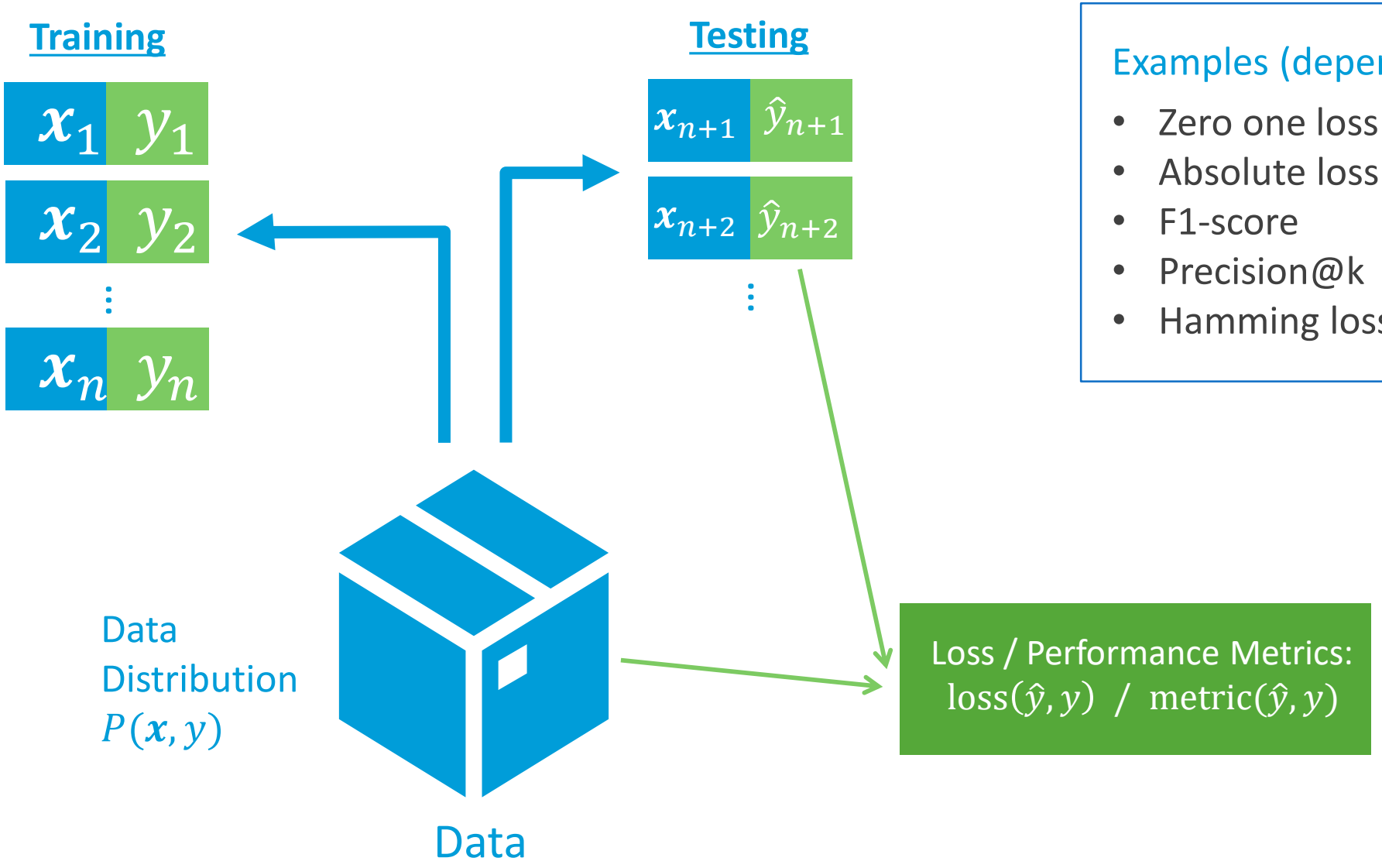
Performance-Aligned Learning Algorithms using Distributionally Robustness Principle

Rizal Fathony
Post-Doctoral Fellow @ Carnegie Melon University

Joint work with: Anqi Liu, Kaiser Asif, Mohammad Bashiri, Wei Xing, Sima Behpour, Xinhua Zhang, Brian Ziebart, Zico Kolter.

Supervised Learning | Classification

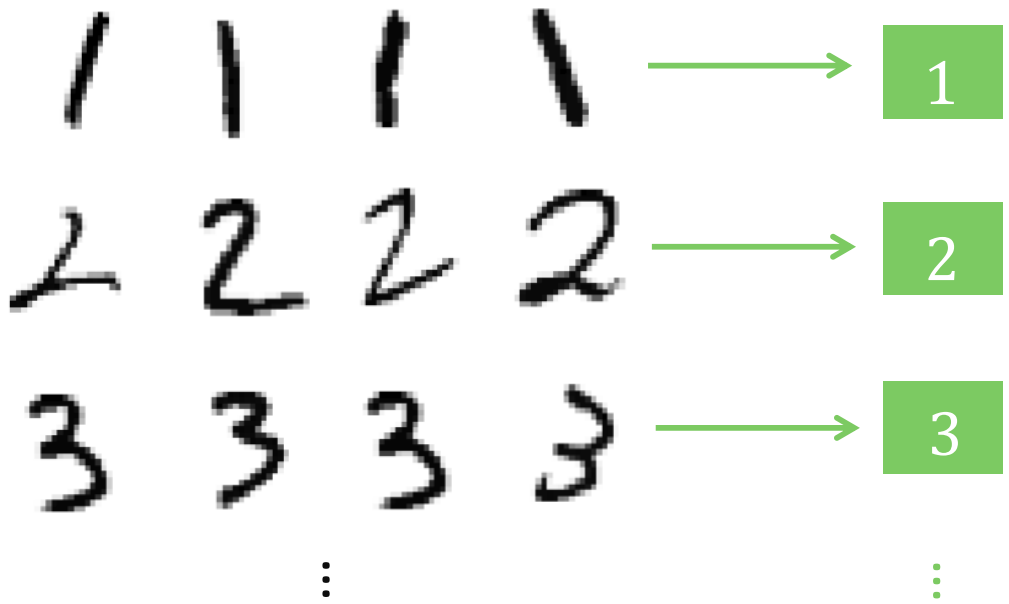


Examples (depend on the task)

- Zero one loss / accuracy metric
- Absolute loss (for ordinal regression)
- F1-score
- Precision@k
- Hamming loss (sum of 0-1 loss)

Binary/Multiclass Classification

Example: Digit Recognition



Performance Metric: Accuracy

$$\text{accuracy}(\hat{\boldsymbol{y}}, \boldsymbol{y}) = \frac{1}{n}\sum_i I(\hat{y}_i = y_i)$$

Loss Metric: Zero-One Loss

$$\text{loss}(\hat{\boldsymbol{y}}, \boldsymbol{y}) = \frac{1}{n}\sum_i I(\hat{y}_i \neq y_i)$$

Ordinal Regression/Classification

Example: Movie Rating Prediction

→ 1 ★☆☆☆☆

→ 2 ★★☆☆☆

⋮ ⋮

→ 5 ★★★★★

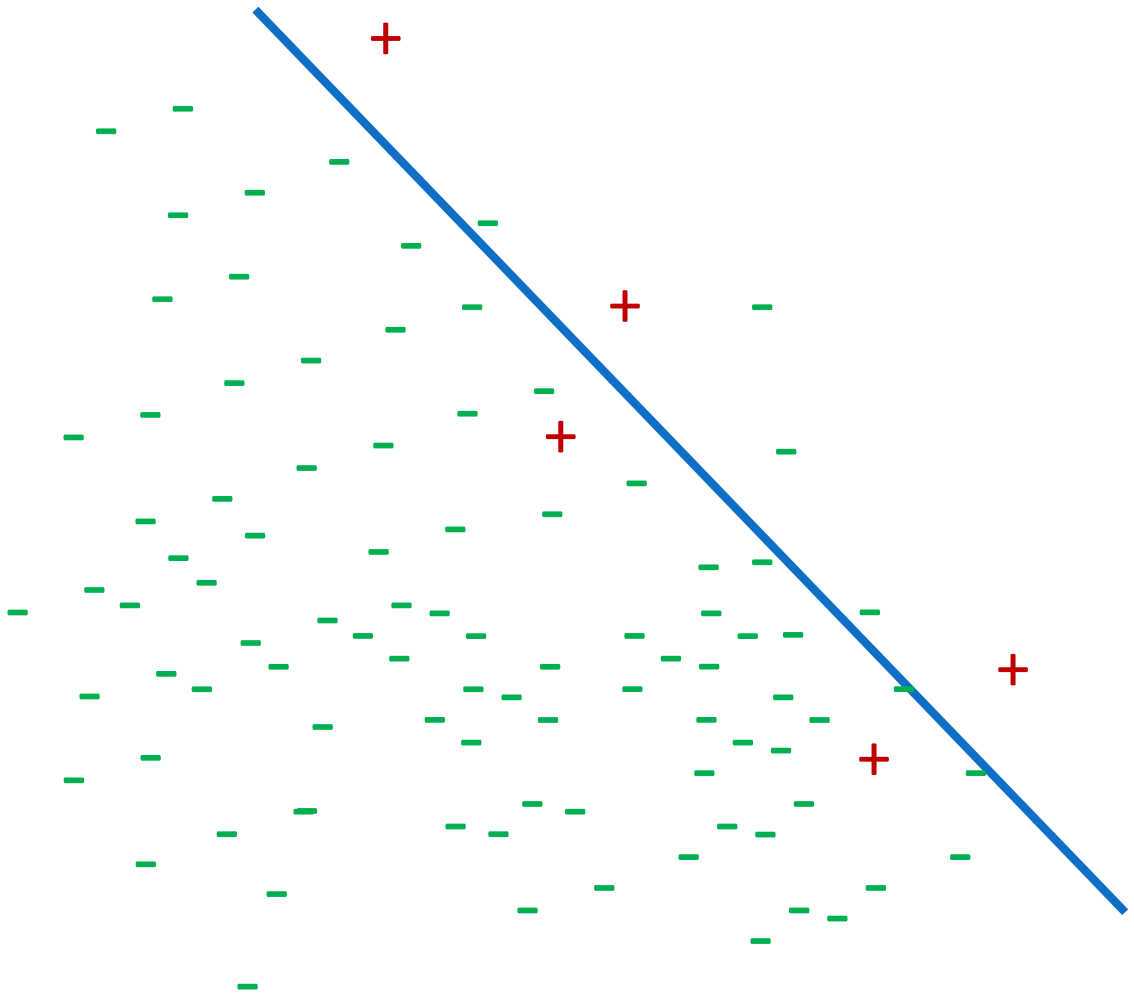
Loss Metric: Absolute Loss

$$\text{absloss}(\hat{\boldsymbol{y}}, \boldsymbol{y}) = \frac{1}{n}\sum_i |\hat{y}_i - y_i|$$

Predicted vs Actual Label:

Distance ↑ → Loss ↑

Classification with Imbalance Datasets



Confusion Matrix

		Actual		
		Positive	Negative	
Pred.	Positive	True Pos. (TP)	False Pos. (FP)	Predicted Pos. (PP)
	Negative	False Neg. (FN)	True Neg. (TN)	Predicted Neg. (PN)
		Actual Pos. (AP)	Actual Neg. (AN)	All Data (ALL)

Performance Metric: F1 Score

$$\mathrm{F1}score(\hat{\boldsymbol{y}}, \boldsymbol{y}) = \frac{2\ \mathrm{TP}}{\mathrm{AP} + \mathrm{PP}}$$

Learning Tasks & Loss/Performance Metric

Machine Learning Tasks	Popular Loss/Performance Metrics
Imbalance Datasets	- F1-Score - Area under ROC Curve (AUC) - Precision vs Recall
Medical classification tasks	- Specificity - Sensitivity - Bookmaker Informedness
Information retrieval tasks	- Precision@k - Mean Average Precision (MAP) - Discounted cumulative gain (DCG)
Weighted classification tasks	- Cost-sensitive loss metric
Rating tasks	- Cohen’s kappa score - Fleiss' kappa score
Computational biology tasks	- Precision-Recall curve - Matthews correlation coefficient (MCC)

Learning Framework

How to design a learning algorithm?

Standard Approach for Learning Algorithms

Empirical Risk Minimization (ERM) [Vapnik, 1992]

- Assume a family of parametric hypothesis function f (e.g. linear discriminator)
- Find the hypothesis f^* that minimize the empirical risk:

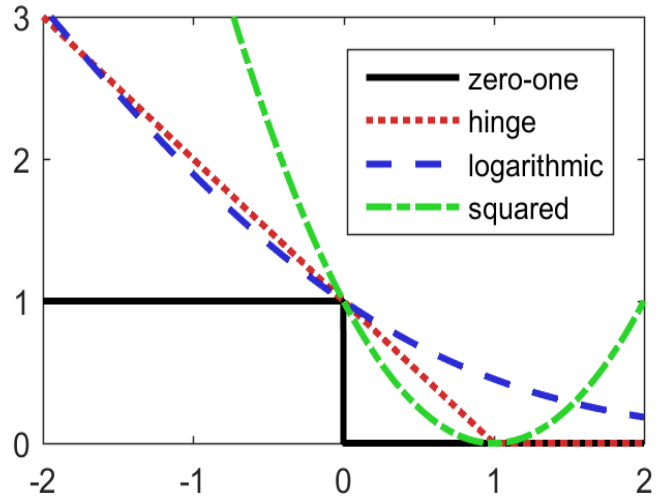
$$\min_f \frac{1}{n} \sum_{i=1}^{n} \text{loss}(f(\mathbf{x}_i), y_i) = \min_f \mathbb{E}_{\tilde{P}(\mathbf{x},y)} \left[\text{loss}(f(\mathbf{x}), y)\right]$$

Intractable optimization!

Since most of loss/performance metrics (e.g. Accuracy, F1-score) are discrete & non-continuous

Surrogate Losses

ERM: prescribes the use of convex surrogate loss to avoid intractability



Example: Binary Classification

Evaluation Metrics: **Accuracy** (Zero-One Loss)

$$\min_f \mathbb{E}_{\tilde{P}(\mathbf{x},y)} \left[\text{ZeroOneLoss}(f(\mathbf{X}), Y)\right]$$

Support Vector Machine (SVM)

$$\min_f \mathbb{E}_{\tilde{P}(\mathbf{x},y)} \left[\text{HingeLoss}(f(\mathbf{x}), y)\right]$$

Logistic Regression (LR)

$$\min_f \mathbb{E}_{\tilde{P}(\mathbf{x},y)} \left[\text{LogLoss}(\hat{P}_f(\hat{y}|\mathbf{x}), y)\right]$$

Adversarial Prediction

A distributionally robust learning framework

Adversarial Prediction (Asif et.al, 2015; Fathony et.al, 2018a)

A Distributionally Robust Approach

Original discrete loss metric:

ERM: e.g. Logistic Regression

$$\min_f \mathbb{E}_{\tilde{P}(\mathbf{x},y)} \left[\text{loss}(f(\mathbf{x}), y)\right] \longrightarrow \min_f \mathbb{E}_{\tilde{P}(\mathbf{x},y)} \left[\text{LogLoss}(\hat{P}_f(\hat{y}|\mathbf{x}), y)\right]$$

Adversarial Prediction:

$$\min_{\mathcal{P}(\hat{y}|\mathbf{x})} \max_{\mathcal{Q}(\check{y}|\mathbf{x})\in\Xi} \mathbb{E}_{\tilde{P}(\mathbf{x})\mathcal{P}(\hat{y}|\mathbf{x})\mathcal{Q}(\check{y}|\mathbf{x})} \left[\text{loss}(\hat{y}, \check{y})\right]$$

Predictor Adversary

Uncertainty set: moment matching on features

$$\Xi \triangleq \{\mathcal{Q} \mid \mathbb{E}_{\tilde{P}(\mathbf{x})\mathcal{Q}(\check{y}|\mathbf{x})}[\phi(\mathbf{x}, \check{y})] = \mathbb{E}_{\tilde{P}(\mathbf{x},y)}[\phi(\mathbf{x}, y)]\}$$

Features Features

→ Operates on the conditional distribution $P(y|\mathbf{x})$ rather than $P(\mathbf{x}, y)$

Adversarial Prediction: Dual Formulation

Primal:

$$\min_{\mathcal{P}(\hat{y}|\mathbf{x})} \max_{\mathcal{Q}(\check{y}|\mathbf{x})\in\Xi} \mathbb{E}_{\tilde{P}(\mathbf{x})\mathcal{P}(\hat{y}|\mathbf{x})\mathcal{Q}(\check{y}|\mathbf{x})} [\text{loss}(\hat{y},\check{y})]$$

$$\Xi \triangleq \{\mathcal{Q} \mid \mathbb{E}_{\tilde{P}(\mathbf{x})\mathcal{Q}(\check{y}|\mathbf{x})}[\phi(\mathbf{x},\check{y})] = \mathbb{E}_{\tilde{P}(\mathbf{x},y)}[\phi(\mathbf{x},y)]\}$$

Lagrange duality, minimax duality

Dual:

discrete loss metric

$$\min_{\theta} \mathbb{E}_{\tilde{P}(\mathbf{x},y)} \left[\max_{\mathcal{Q}(\check{y}|\mathbf{x})} \min_{\mathcal{P}(\hat{y}|\mathbf{x})} \mathbb{E}_{\mathcal{P}(\hat{y}|\mathbf{x})\mathcal{Q}(\check{y}|\mathbf{x})} [\text{loss}(\hat{y},\check{y}) + \theta^\intercal (\phi(\mathbf{x},\check{y}) - \phi(\mathbf{x},y))] \right]$$

Lagrange multiplier for the constraints

Convex w.r.t. θ

Decomposable Metrics

(Asif et.al, 2015; Fathony et.al, 2016, 2017, 2018a)

Examples: Multiclass, Ordinal, Taxonomy-based, and Cost-sensitive Classification

Decomposable metrics: $\text{loss}(\hat{\mathbf{y}}, \mathbf{y}) = \frac{1}{n} \sum_i \text{loss}(\hat{y}_i, y_i)$

$$\min_{\theta} \mathbb{E}_{\tilde{P}(\mathbf{x},y)} \left[\max_{\mathcal{Q}(\check{y}|\mathbf{x})} \min_{\mathcal{P}(\hat{y}|\mathbf{x})} \mathbb{E}_{\mathcal{P}(\hat{y}|\mathbf{x})\mathcal{Q}(\check{y}|\mathbf{x})} \left[\text{loss}(\hat{y}, \check{y}) + \theta^\intercal \left(\phi(\mathbf{x}, \check{y}) - \phi(\mathbf{x}, y) \right) \right] \right]$$

Simple loss metrics: Analytical solution

(e.g. Zero-One, Absolute Loss); by analyzing the equilibrium solution of zero-sum game.

More complex losses: Reformulation as a Linear Program

$$\max_{\mathbf{q}, v} \; v + \mathbf{f}^\intercal \mathbf{q} - f_y$$

$$\text{s.t.:} \; \mathbf{L}_{(i,:)} \mathbf{q} \geq v \quad \forall i \in [k]$$

$$q_i \geq 0 \quad \forall i \in [k]$$

$$\mathbf{q}^\intercal \mathbf{1} = 1,$$

where:

$$p_i = \hat{P}(\hat{Y} = i|\mathbf{x})$$

$$q_i = \check{P}(\check{Y} = i|\mathbf{x})$$

$$f_i = \theta^\intercal \phi(\mathbf{x}, i)$$

$\mathbf{L}$ is the loss in a matrix form, e.g:

$$\mathbf{L} = \begin{bmatrix} 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{bmatrix}$$

for a 4-class zero-one loss

# of variable = $k + 1$, where k = # of class

Non-Decomposable Metrics

Non-Decomposable Metric

Example:

Binary Classification with F1-score metric

$$\text{F1-score}(\hat{\mathbf{y}}, \mathbf{y}) = \frac{2\ \text{TP}}{\text{PP} + \text{AP}} = \frac{2\sum_i \hat{y}_i y_i}{\sum_i \hat{y}_i + \sum_i y_i}$$

		Actual: Positive	Actual: Negative	
Pred.	Positive	True Pos. (TP)	False Pos. (FP)	Predicted Pos. (PP)
Pred.	Negative	False Neg. (FN)	True Neg. (TN)	Predicted Neg. (PN)
		Actual Pos. (AP)	Actual Neg. (AN)	All Data (ALL)

Dual:

$$\max_{\theta} \mathbb{E}_{\tilde{P}(\mathbf{x},\mathbf{y})} \left[\min_{\mathcal{Q}(\check{\mathbf{y}}|\mathbf{x})} \max_{\mathcal{P}(\hat{\mathbf{y}}|\mathbf{x})} \sum_{\hat{\mathbf{y}},\check{\mathbf{y}}} \mathcal{P}(\hat{\mathbf{y}}|\mathbf{x}) \mathcal{Q}(\check{\mathbf{y}}|\mathbf{x}) \left(\frac{2\sum_i \hat{y}_i \check{y}_i}{\sum_i \hat{y}_i + \sum_i \check{y}_i} - \theta^\intercal \sum_i^n [\phi(\mathbf{x}, \check{y}_i) - \phi(\mathbf{x}, y_i)] \right) \right]$$

Size: 2^n Intractable!

Dual | Marginal Formulation:

$$\max_{\theta} \mathbb{E}_{\tilde{P}(\mathbf{x},\mathbf{y})} \left[\min_{\mathcal{Q}(\check{\mathbf{y}}|\mathbf{x})} \max_{\mathcal{P}(\hat{\mathbf{y}}|\mathbf{x})} \sum_{k\in[0,n]} \sum_{l\in[0,n]} \frac{2}{k+l} \sum_i \mathcal{P}(\hat{y}_i = 1, \textstyle\sum_i \hat{y}_i = k|\mathbf{x}) \mathcal{Q}(\check{y}_i = 1, \textstyle\sum_i \check{y}_i = l|\mathbf{x}) \right.$$

$$\left. - \theta^\intercal \sum_i^n [\mathcal{Q}(\check{y}_i = 1|\mathbf{x}) \phi(\mathbf{x}, \check{y}_i) - \phi(\mathbf{x}, y_i)] \right]$$

Size: n^2

Generic Non-Decomposable Performance Metrics

Fathony & Kolter (in-submission)

		Actual		
		Positive	Negative	
Pred.	Positive	True Pos. (TP)	False Pos. (FP)	Predicted Pos. (PP)
	Negative	False Neg. (FN)	True Neg. (TN)	Predicted Neg. (PN)
		Actual Pos. (AP)	Actual Neg. (AN)	All Data (ALL)

More complex performance metric

$$\text{metric}(\hat{\mathbf{y}}, \mathbf{y}) = \sum_j \frac{f_j(\ \mathsf{TP}\ ,\ \mathsf{TN}\ ,\ \mathsf{PP}\ ,\ \mathsf{AP}\ )}{g_j(\ \mathsf{PP}\ ,\ \mathsf{AP}\ )}$$

$$= \sum_j \frac{f_j(\sum_i \hat{y}_i y_i, \sum_i (1-\hat{y}_i)(1-y_i), \sum_i \hat{y}_i, \sum_i y_i)}{g_j(\sum_i \hat{y}_i, \sum_i y_i)}$$

f is a linear function over TP and TN

Cover most popular metrics:
e.g. Precision, Recall, F_β-score, Balanced Accuracy, Specificity, Sensitivity, Informednes, Kappa score, etc…

Dual | Marginal Formulation:

$$\max_\theta \mathbb{E}_{\tilde{P}(\mathbf{x},\mathbf{y})} \Bigg[\min_{\mathcal{Q}(\check{\mathbf{y}}|\mathbf{x})} \max_{\mathcal{P}(\hat{\mathbf{y}}|\mathbf{x})} \sum_{k\in[0,n]} \sum_{l\in[0,n]} \sum_j \frac{1}{g_j(k,l)} f_j\Big(\textstyle\sum_i \mathcal{P}(\hat{y}_i = 1, \sum_i \hat{y}_i = k)\mathcal{Q}(\check{y}_i = 1, \sum_i \check{y}_i = l),$$

$$\textstyle\sum_i (\mathcal{P}(\hat{y}_i = 0, \sum_i \hat{y}_i = k))(\mathcal{Q}(\check{y}_i = 0, \sum_i \check{y}_i = l)),\ k,\ l\Big) - \theta^\intercal \textstyle\sum_i^n [\mathcal{Q}(\check{y}_i = 1|\mathbf{x})\phi(\mathbf{x}, \check{y}_i) - \phi(\mathbf{x}, y_i)] \Bigg]$$

Size: $2n^2$

Integration with Deep Learning Pipeline

Fathony & Kolter (in-submission)

Programming Interface

Enable programmers to easily incorporate custom performance metric into their deep learning pipeline

```
model = Chain(
  Conv((5, 5), 1=>20, relu),
  MaxPool((2,2)),
  Conv((5, 5), 20=>50, relu),
  MaxPool((2,2)),
  x -> reshape(x, :, size(x, 4)),
  Dense(4*4*50, 500),
  Dense(500, 1), vec
)

objective(x, y) = mean(
  logitbinarycrossentropy(model(x), y))

opt = ADAM(1e-3)
Flux.train!(objective, params(model),
  train_set, opt)
```

Leaning using binary cross entropy

```
model = Chain(
  Conv((5, 5), 1=>20, relu), MaxPool((2,2)),
  Conv((5, 5), 20=>50, relu), MaxPool((2,2)),
  x -> reshape(x, :, size(x, 4)),
  Dense(4*4*50, 500), Dense(500, 1), vec
)

@metric F2Score
function define(::Type{F2Score}, C::ConfusionMatrix)
    return ((1 + 2^2) * C.tp) / (2^2 * C.ap + C.pp)
end
f2_score = F2Score()
enforce_special_case_positive!(f2_score)

objective(x, y) = ap_objective(model(x), y, f2_score)
Flux.train!(objective, params(model), train_set, ADAM(1e-3))
```

F_β score definition

$$\frac{(1+\beta^2)\ \mathrm{TP}}{\beta^2\ \mathrm{AP} + \mathrm{PP}}$$

Leaning using AP formulation for F2-metric

*) The codes are written in Julia

Integration with Deep Learning Pipeline

Fathony & Kolter (in-submission)

Code examples for other performance metrics:

```
@metric GM_PrecRec      # Geometric Mean of Prec and Rec
function define(::Type{GM_PrecRec}, C::ConfusionMatrix)
    return C.tp / sqrt(C.ap * C.pp)
end
gpr = GM_PrecRec()
enforce_special_case_positive!(gpr)
```

Geometric mean of precision and recall

```
@metric Kappa
function define(::Type{Kappa}, C::ConfusionMatrix)
    pe = (C.ap * C.pp + C.an * C.pn) / C.all^2
    num = (C.tp + C.tn) / C.all - pe
    den = 1 - pe
    return num / den
end
kappa = Kappa()
enforce_special_case_positive!(kappa)
enforce_special_case_negative!(kappa)
```

Cohen's kappa score metric

NAME	FORMULA
F_β-score	$\frac{(1+\beta^2)\ \mathrm{TP}}{\beta^2\ \mathrm{AP} + \mathrm{PP}}$
Geom. mean of Prec. & Recall	$\frac{\mathrm{TP}}{\sqrt{\mathrm{PP} \cdot \mathrm{AP}}}$
Balanced Accuracy	$\frac{1}{2}\left(\frac{\mathrm{TP}}{\mathrm{AP}} + \frac{\mathrm{TN}}{\mathrm{AN}}\right)$
Bookmaker Informedness	$\frac{\mathrm{TP}}{\mathrm{AP}} + \frac{\mathrm{TN}}{\mathrm{AN}} - 1$
Markedness	$\frac{\mathrm{TP}}{\mathrm{PP}} + \frac{\mathrm{TN}}{\mathrm{PN}} - 1$
Cohen's kappa score	$\frac{(\ \mathrm{TP} + \mathrm{TN}\ )/\ \mathrm{ALL} - (\ \mathrm{AP} \cdot \mathrm{PP} + \mathrm{AN} \cdot \mathrm{PN}\ )/\ \mathrm{ALL}\ ^2}{1-(\ \mathrm{AP} \cdot \mathrm{PP} + \mathrm{AN} \cdot \mathrm{PN}\ )/\ \mathrm{ALL}\ ^2}$

```
objective(x, y) = ap_objective(model(x), y, gpr)
Flux.train!(objective, params(model),
            train_set, ADAM(1e-3))
```

```
objective(x, y) = ap_objective(model(x), y, kappa)
Flux.train!(objective, params(model),
            train_set, ADAM(1e-3))
```

*) The codes are written in Julia

Conclusion

Conclusion

Adversarial Prediction Framework

- A distributionally robust learning framework
  with uncertainty set defined over the conditional distributions
- Align with the loss/performance metric
  by incorporating the metric into its learning objective
- Computationally efficient
  via marginalization technique
- Easy to integrate with deep learning pipeline
- Perform well in practice

References

- **Adversarial Cost-Sensitive Classification**
  Kaiser Asif, Wei Xing, Sima Behpour, and Brian D. Ziebart.
  Conference on Uncertainty in Artificial Intelligence (UAI), 2015.

- **Adversarial Multiclass Classification: A Risk Minimization Perspective**
  Rizal Fathony, Anqi Liu, Kaiser Asif, Brian D. Ziebart.
  Advances in Neural Information Processing Systems 29 (NeurIPS), 2016.

- **Adversarial Surrogate Losses for Ordinal Regression**
  Rizal Fathony, Mohammad Bashiri, Brian D. Ziebart.
  Advances in Neural Information Processing Systems 30 (NeurIPS), 2017.

- **Consistent Robust Adversarial Prediction for General Multiclass Classification**
  Rizal Fathony, Kaiser Asif, Anqi Liu, Mohammad Bashiri, Wei Xing, Sima Behpour, Xinhua Zhang, Brian D. Ziebart.
  ArXiv preprint, 2018.

- **AP-Perf: Incorporating Generic Performance Metrics in Differentiable Learning**
  Rizal Fathony and Zico Kolter
  In submission, 2019.

Thank You