Structured Prediction Energy Networks

David Belanger

Andrew McCallum

Overview

- Structured Prediction Background
  - Multi-label classification
  - Structured prediction approaches
  - Expressivity-tractability tradeoffs
  - Chicken and egg problem
- Deep Learning and Structured Prediction
  - Prior attempts to improve expressivity in structured prediction
  - Motivation for our work
- Structured Prediction Energy Networks
  - Very generic prediction technique: gradient descent
  - Model label interactions using a deep architecture
- Experiments

STRUCTURED PREDICTION BACKGROUND

Multi-Label Classification

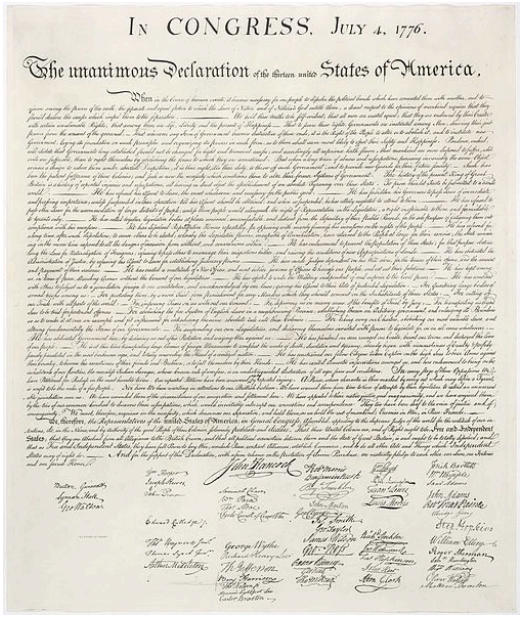
Predict $y = \{y_1, \ldots, y_L\} \in \{0, 1\}^L$

True labels:

animal, daytime, action, outside, …

False labels:

nighttime, inside, person, …



True labels:

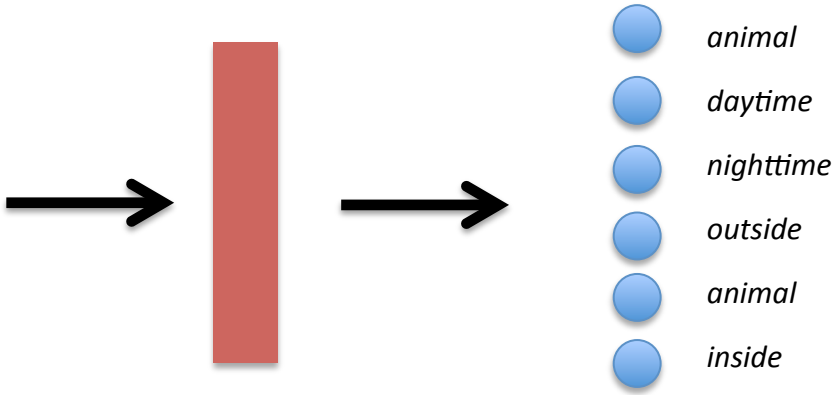
politics, signed, historical_document,…

False labels:

sports, cooking, dialog, …

Images: Wikimedia Commons

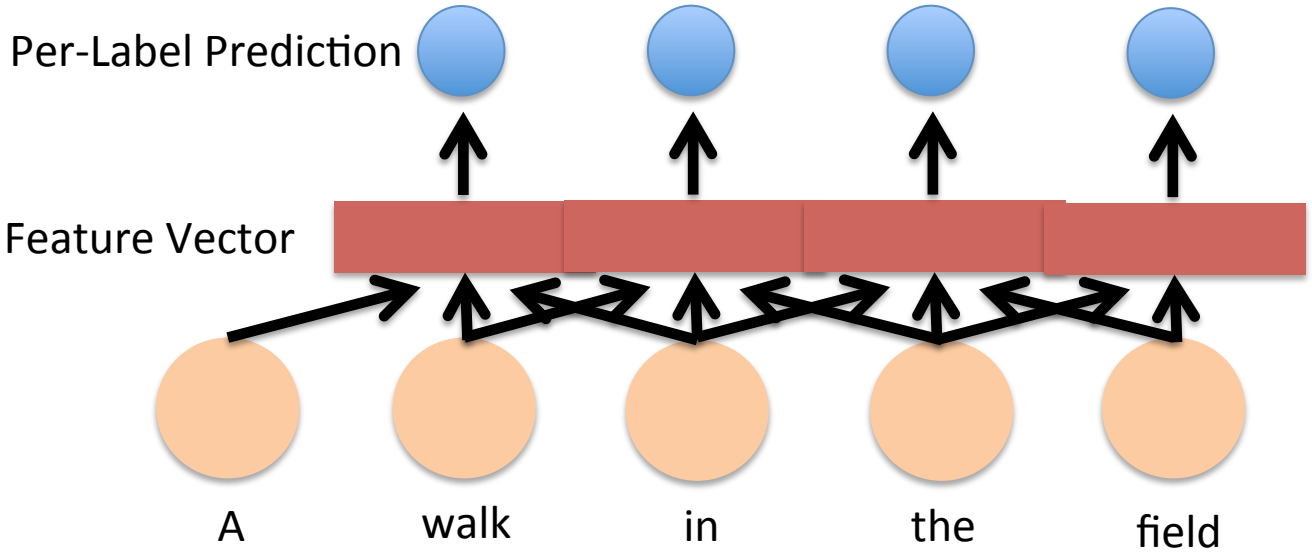
Independent Prediction



Input

Feature Vector

Independent Per-Label Predictors



Bad model, but it's fast!

Joint Prediction

Input

Feature Vector

$E_x(y)$

Energy Function

$$\arg\min_{y \in \mathcal{Y}} E_x(y)$$

Test-Time Optimization

CRF, HMM, etc.

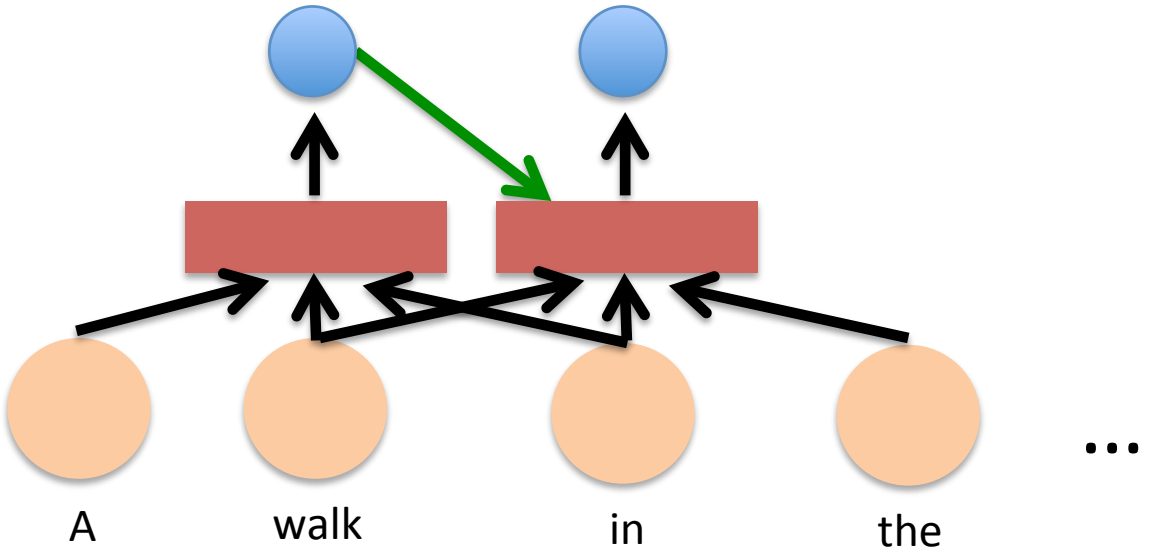
Independent Prediction: models interactions among inputs

Joint Prediction: models interaction among inputs and among labels

Joint Structured Prediction Examples

Model Structure	Prediction Method
Sequence	Viterbi, Forward-Backward
Grid	Loopy BP, Graph Cuts, Gibbs Sampling
Parse Tree	Maximum Spanning Tree, Eisner Algorithm

Search-Based Joint Prediction



Eg. Hall et al., 2006; Daume et al., 2009; Chang et al., 2015

Why Test-Time Optimization? Isn't Everything 'Feed Forward?'

$$y^* = F(x)$$

$$y^* = \arg\min_{y \in \mathcal{Y}} E_x(y) := G(x)$$

vs.

Answer: Yes, but this perspective isn't helpful.

Overall, you want a predictor that is:

1) Parametrized compactly
2) Provides opportunities to inject domain knowledge.
3) Differentiable

Optimization-based prediction often satisfy these criteria.

Images: amazon.com and whitewaterwest.com

Tradeoffs in Structured Prediction

Statistical

Expressivity vs. Parsimony

Computational

Complexity of Prediction

Learning Joint Prediction Models

$$\min_{\theta} \sum_{i} L\left(y_i, F_\theta(x_i)\right)$$

$$\min_{\theta} \sum_{i} L\left(y_i, \arg\min_{y} E_\theta(y; x_i)\right)$$

exact optimization may be intractable

$$\min_{\theta} \sum_{i} L\left(y_i, \text{Algorithm}_{\Psi(\theta)}(x_i)\right)$$

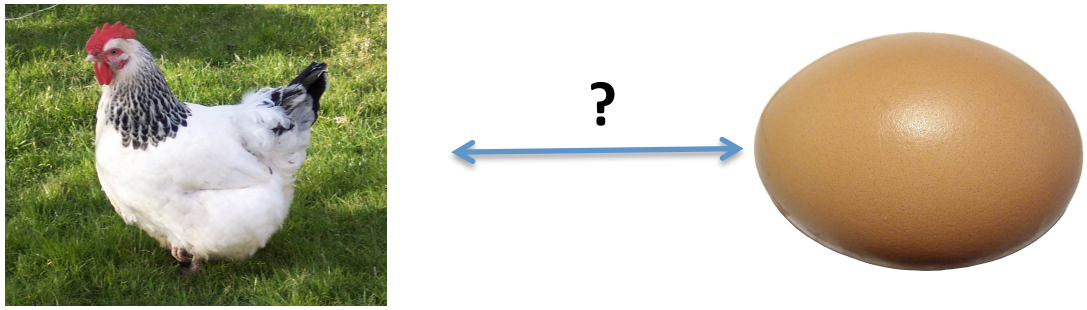
Structured Prediction in Practice

- Goal 1:

  Model the important interactions between x and y.

- Goal 2:

  Choose a model that provides efficient combinatorial optimization over y.



Model

Inference Algorithm

Structure Learning

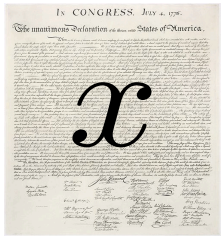
- Input: multivariate data
- Output: graphical model (directed or undirected)
- Goal: use the structure to study the data

- Challenge: search in space of structures

STRUCTURED PREDICTION ENERGY NETWORKS

Our Experiments

Multi-Label Classification



y

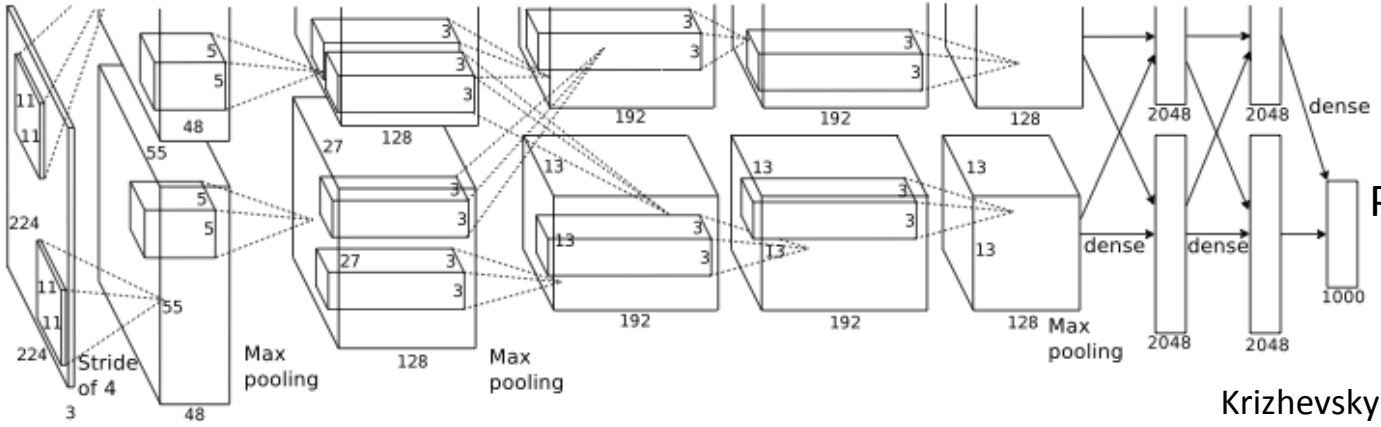
- *politics*
- *sports*
- *historical_document*
- *cooking*
- *signed*
- ...

Difficult Modeling Task

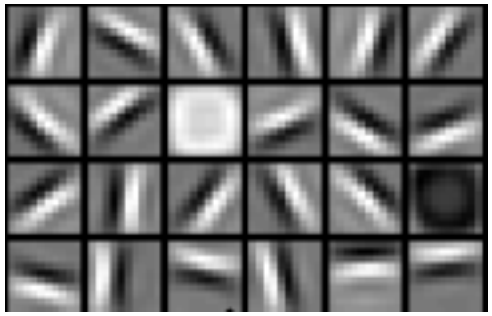
- Labels may be mutually exclusive, imply each other, etc.
- Would like to automatically discover the interactions between labels
- Candidate models (eg loopy CRFs) are computationally expensive

Deep Learning

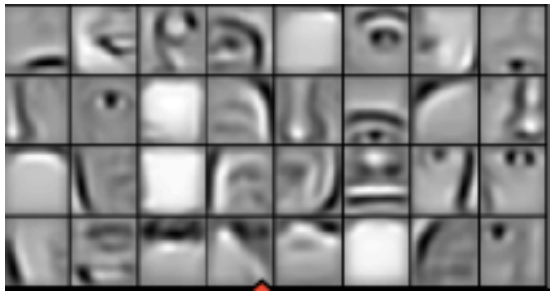
Input



Krizhevsky et al., 2012



Edge Detectors

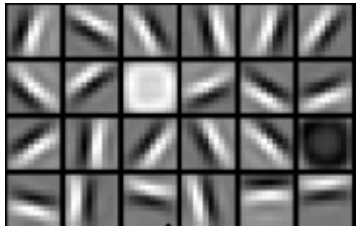


Object Part Detectors

Ng., CS229 lecture

Motivation for Our Work

Neural networks are good at feature learning for the input x.
Can we also use them to do structure learning for the output y?

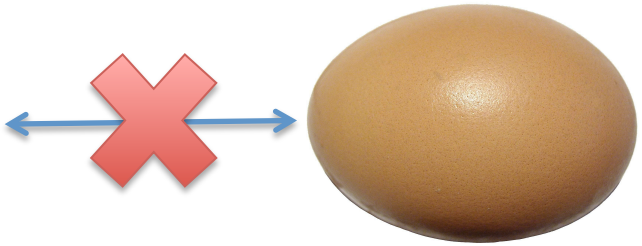


x

?

y

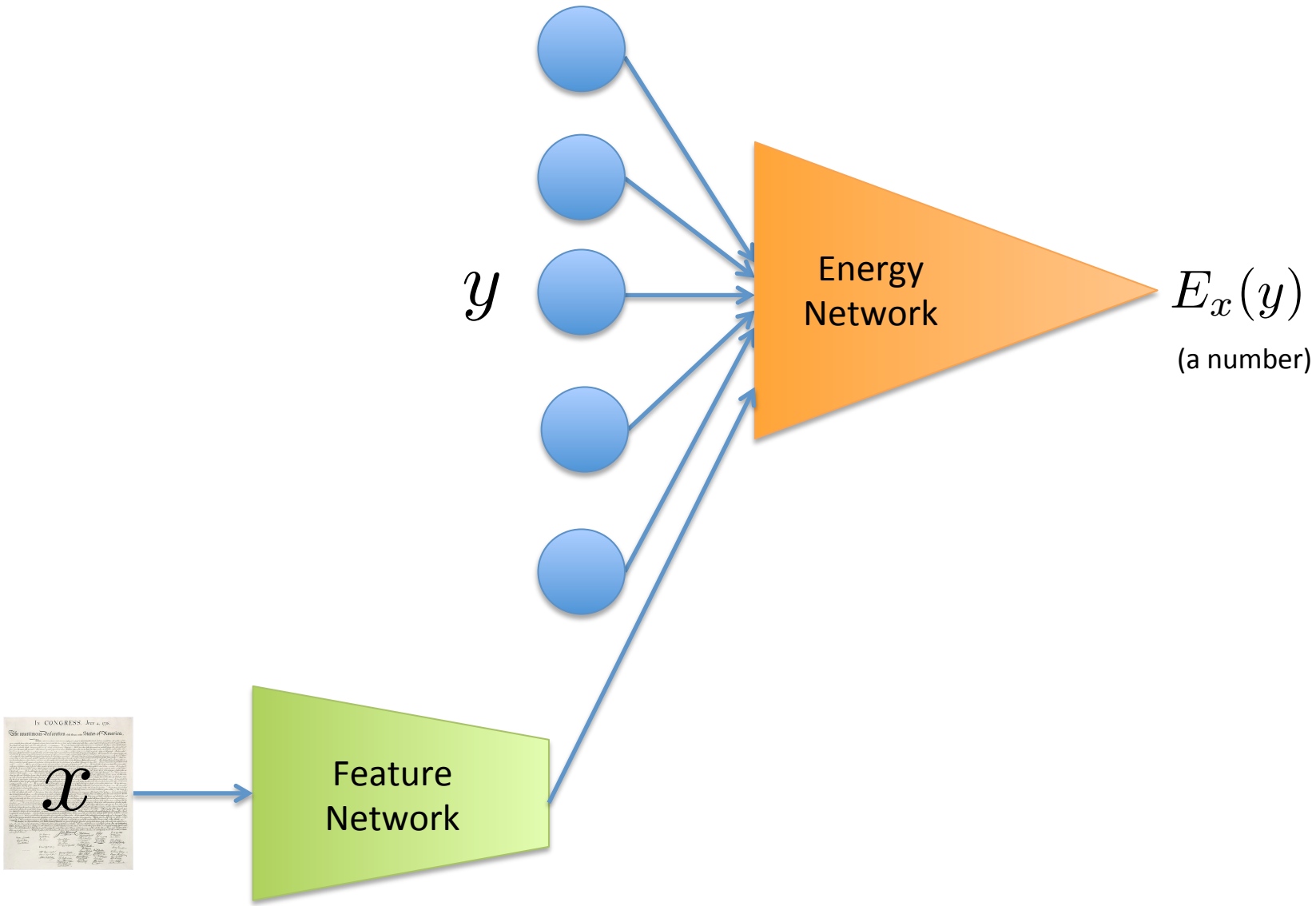
Gradient descent is an extremely generic optimization algorithm.
Let's use it at test time to produce predictions.



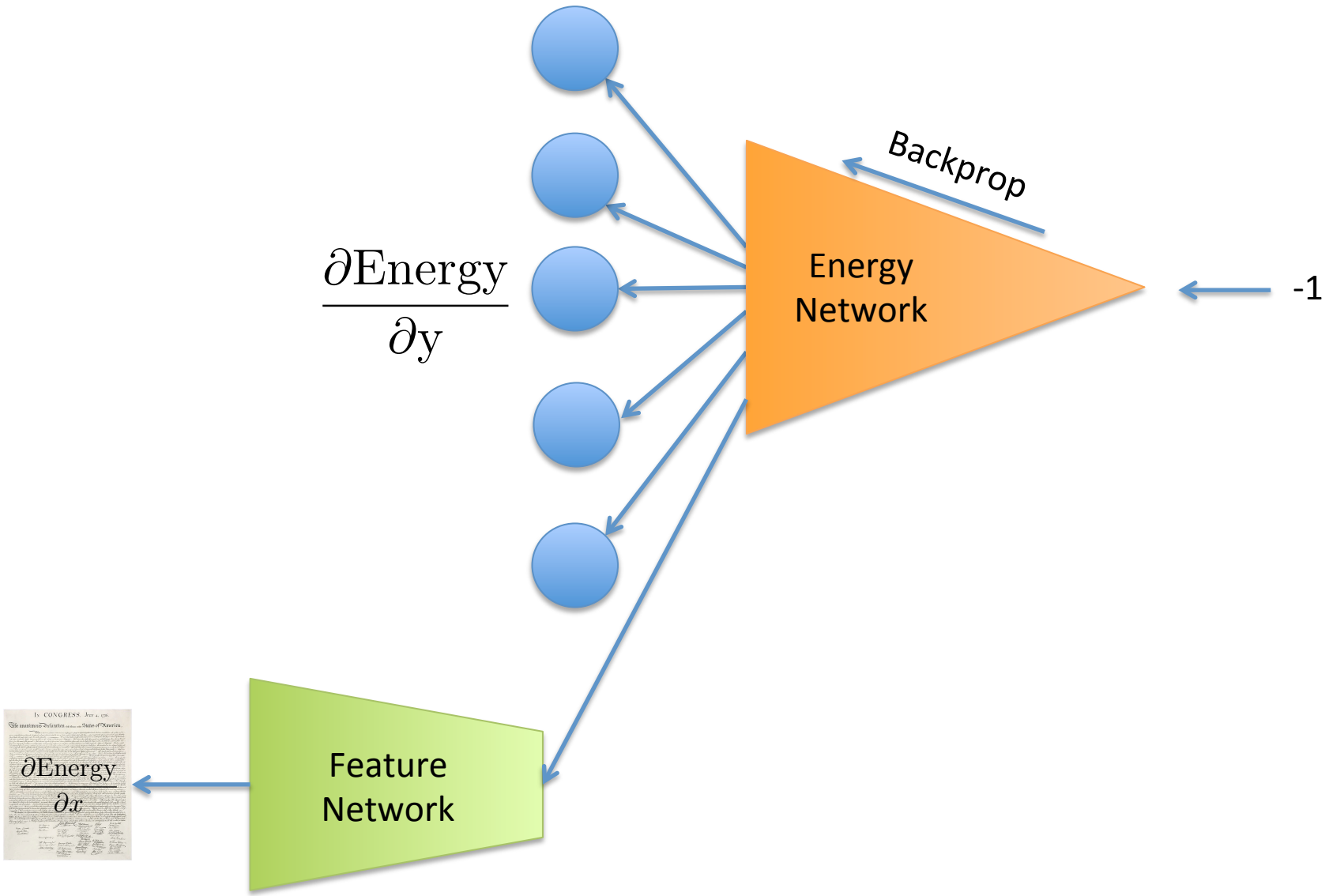
Model

Inference Algorithm

Structured Prediction Energy Network



Differentiating the Energy



SPEN

Key Simplification: Convex Relaxation of y to $\bar{y}$

$$\{0,1\}^L \longrightarrow [0,1]^L$$

Deep Energy Function

Feature Network $F(x)$ (returns a vector)

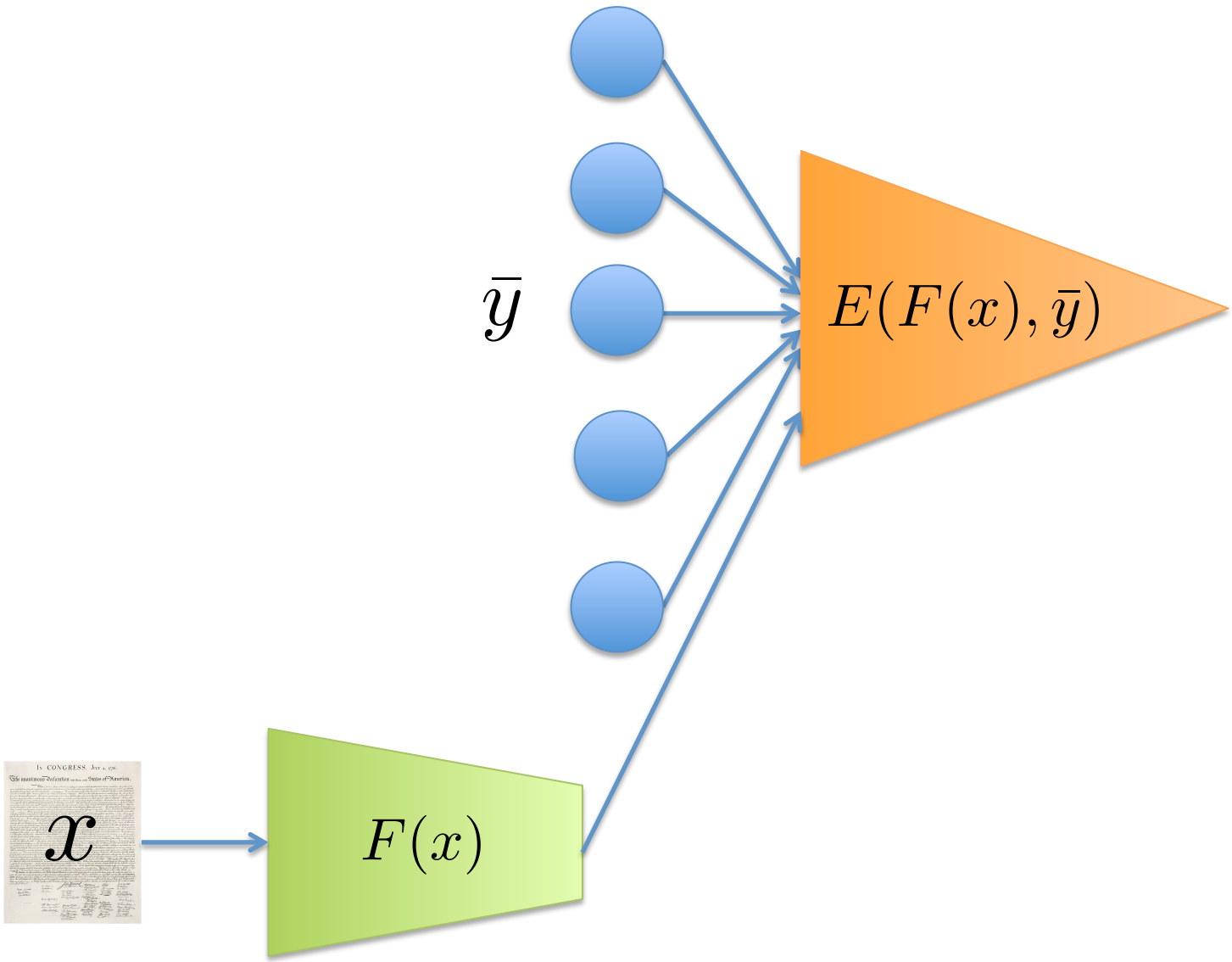
Energy Network $E(F(x), \bar{y})$

'Soft' Prediction $\bar{y}^* = \underset{\bar{y} \in [0,1]^L}{\arg\min}\, E_x(\bar{y})$

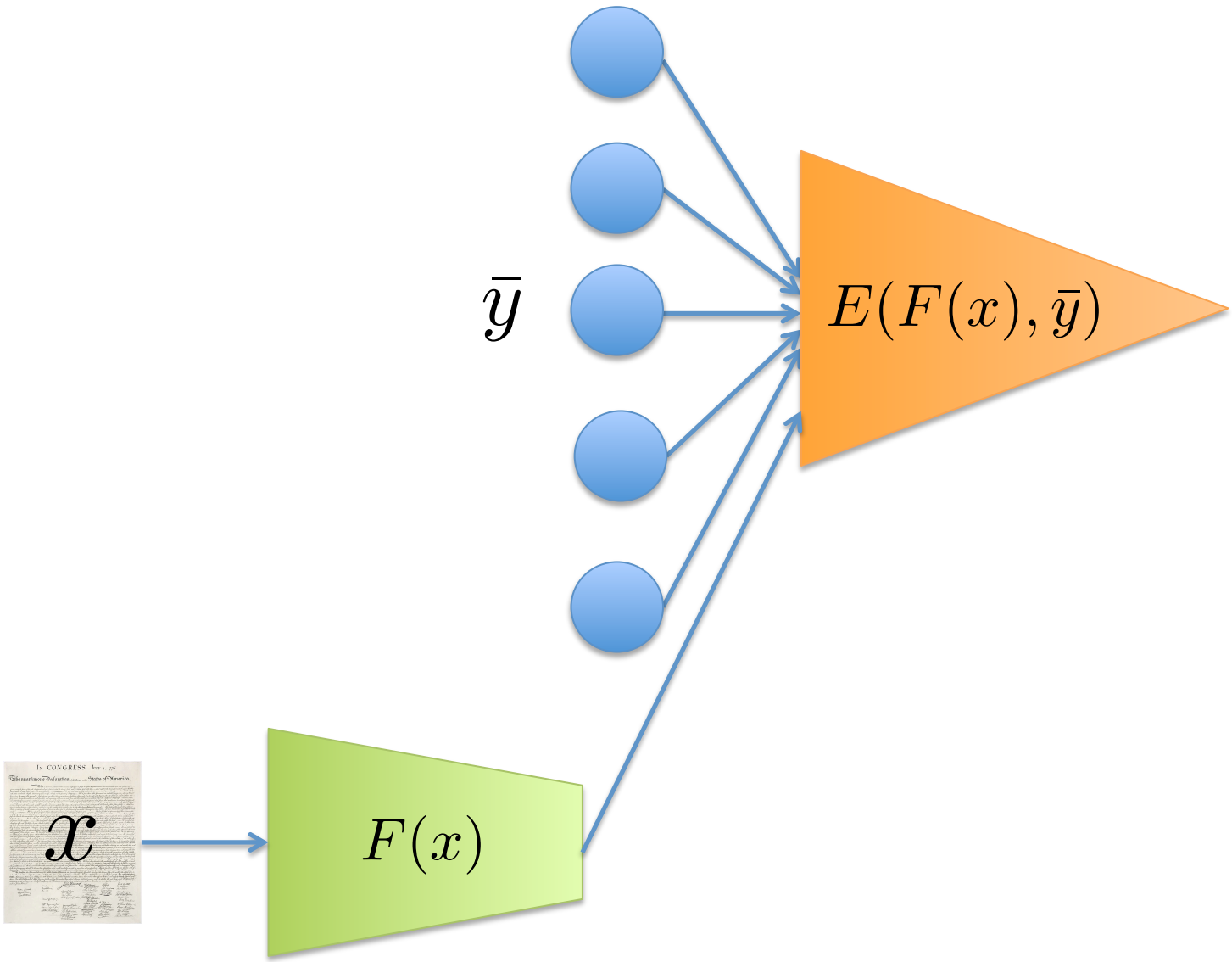
Solve using projected gradient descent

Final Prediction: either use soft prediction, or round

SPEN



SPEN



Prediction

initialize $\bar{y}_0$ uniformly
$f \leftarrow F(X)$ %precompute features
while *not converged* **do**
$g_t \leftarrow \frac{\partial}{\partial \bar{y}} E(f, \bar{y}_t)$
$\bar{y}_{t+1} \leftarrow \bar{y}_t \circ \exp(g_t)$
$\bar{y}_{t+1} \leftarrow \text{Normalize}(\bar{y}_{t+1})$
end

entropic mirror descent
(aka exponentiated gradient)

(Beck and Teboulle, 2003)

SPEN Architecture used In Our Experiments

Feature Network: Multi-Layer Perceptron

$$F(x) = g(A_2(g(A_1(x)))$$

Energy Network: Two Terms

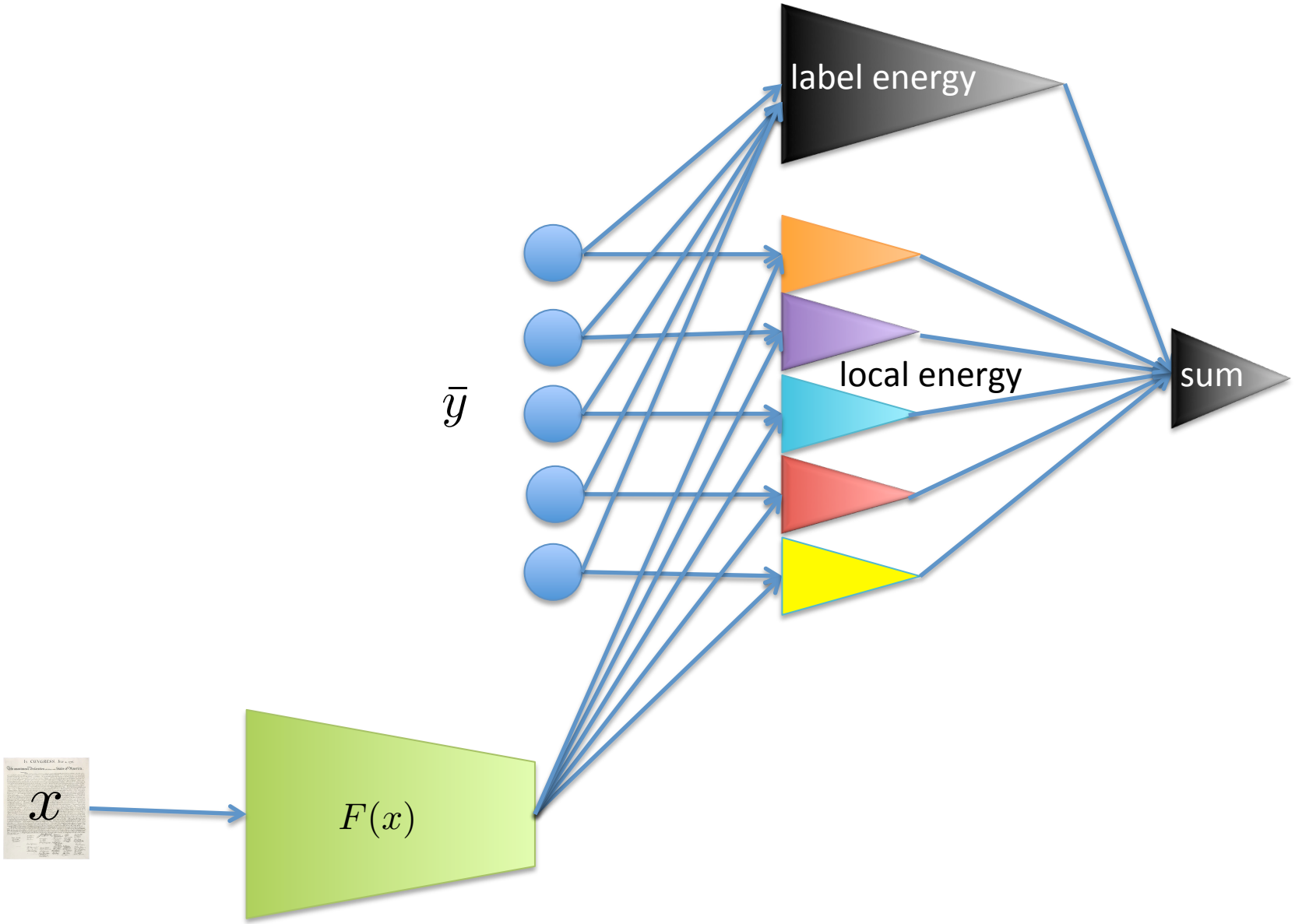
$$E_x^{\text{local}}(\bar{y}) = \sum_{i=1}^{L} \bar{y}_i b_i^\top f(x)$$

per-label linear model

$$E_x^{\text{label}}(\bar{y}) = c_2^\top g(C_1 \bar{y})$$

sometimes deeper

Architecture in Our Experiments



Structure Learning using SPENs

$$E_x^{\text{label}}(\bar{y}) = c_2^\top g(C_1 \bar{y})$$

Measurement Matrix

Computation scales linearly with number of labels.

Captures high-arity interactions without exponential blowup.

Measurement matrix may reveal interpretable structure.

Learning

Structured SVM Loss (Taskar et al., 2004; Tsochantaridis et al., 2004)

$$\sum_{\{x_i, y_i\}} \max_y \left[\Delta(y_i, y) - E_{x_i}(y) + E_{x_i}(y_i)\right]_+$$

Loss-Augmented Inference

$$\arg\min_y \left[-\Delta(y_i, y) + E_{x_i}(y)\right]$$

we require differentiable surrogate

Learning: max-min problem with inexact optimization in inner loop.

Comparison to CRFs

$$E_x^{\text{crf}}(y) = s_2^\top \text{vec}(yy^\top)$$

	SPEN	CRF
Dependence on # labels	O(L)	$O(L^2)$ or worse
Convex prediction problem	No	Sometimes
Automatic Structure Learning	Easy	Yes, but Complicated
Max Likelihood Training	No	Yes
'Soft' Predictions	Kind Of	Yes
LP Relaxation	No	Yes

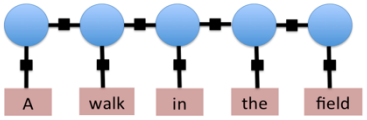
Inexact prediction in inner loop of SSVM is more benign for CRFs than for SPENS (Kulesza & Pereira, 2007; Finley & Joachims, 2008)

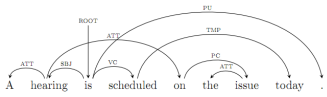
Details

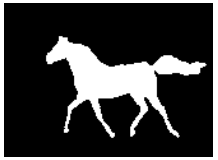
- Cache features F(x) for $E(F(x), \bar{y})$.
- Aggressively parallelize prediction using GPUs.
- Pre-train feature network F(x) using independent per-label loss.
- Use more sophisticated optimization for y (eg., Nesterov Acceleration).

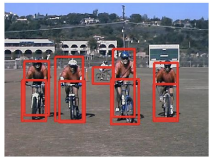
RELATED WORK

Comparing Structured Prediction Tasks





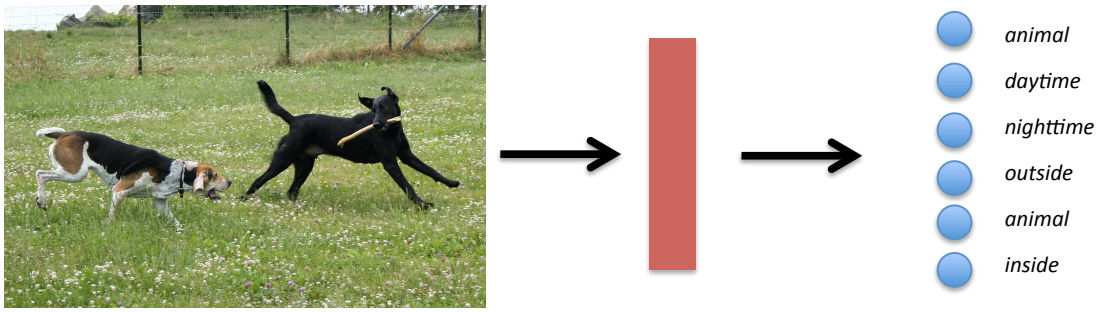




Task	Constrained Outputs	Strong Prior Knowledge about Likely Structures
Multi-Label Classification	no	no
Sequence Tagging	no	yes
Dependency Parsing	yes	yes
Image Segmentation	no	yes
Object Detection	no	no

Deep Learning & Independent Prediction

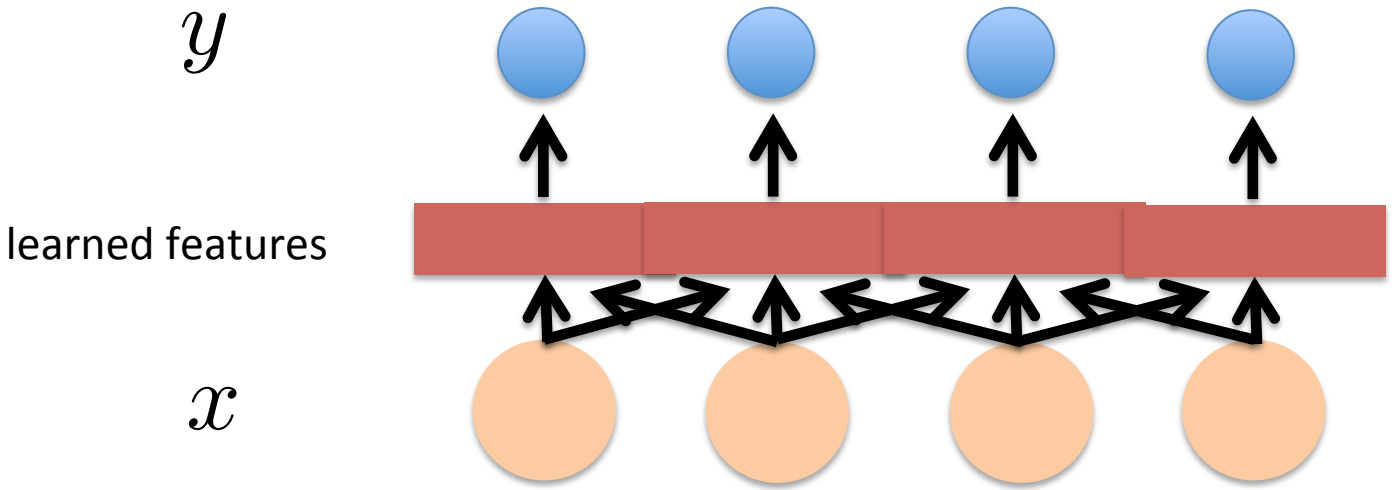
Just make it deep!



Input

Feature Vector

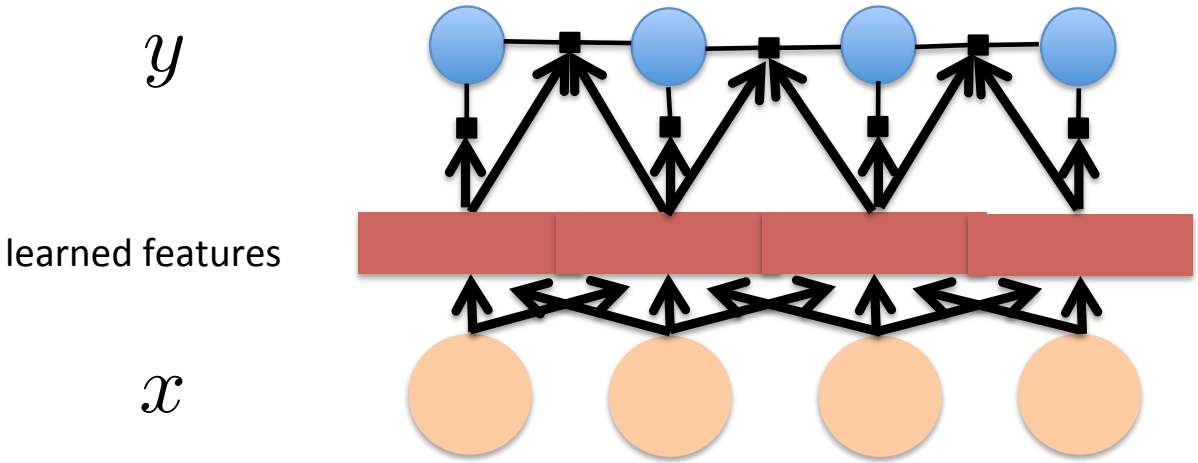
Independent Per-Label Predictors



Deep Learning & Joint Prediction

Conditional Random Field:
mapping from x to a Markov Random Field over y

Prior work: use deep features, but keep same graphical model for y



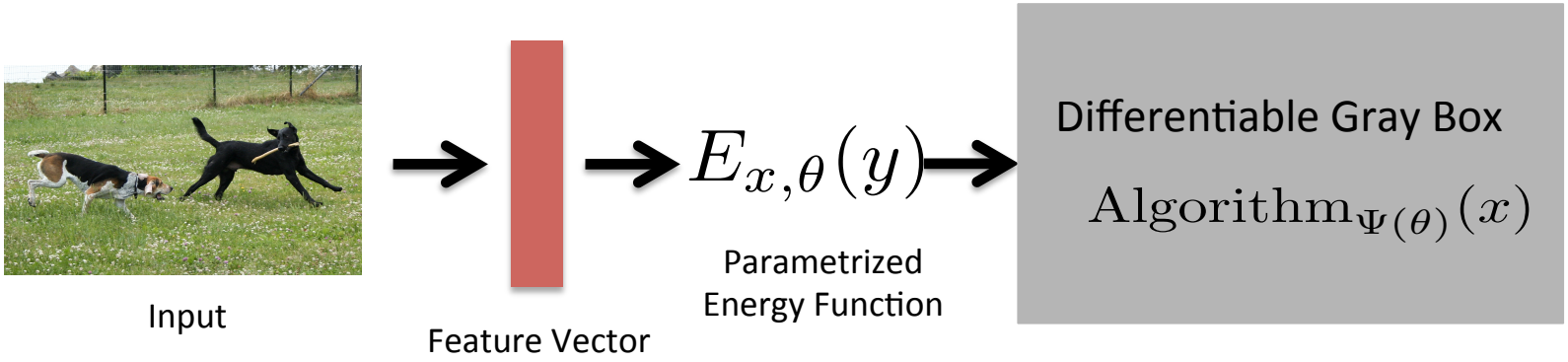
Pro: use existing prediction algorithms

Con: no structure learning; only performs feature learning on x

Eg., LeCun et al., 2006; Collobert et al., 2011; Huang et al., 2015

Deep 'Unrolling' of Structured Prediction Algorithms

$$\min_\theta \sum_i L\left(y_i, \arg\min_y E_\theta(y; x_i)\right) \longrightarrow \min_\theta \sum_i L\left(y_i, \text{Algorithm}_{\Psi(\theta)}(x_i)\right)$$



- Step 1: choose a model family
- Step 2: choose an (approximate) prediction technique
- Step 3: unroll the prediction technique into a deep computation graph with free parameters Ψ
- Step 4: train Ψ directly with backprop

Eg., Stoyanov et al., 2011; Domke, 2013; Hershey et al., 2014; Zheng et al., 2015

Example of 'Unrolling'

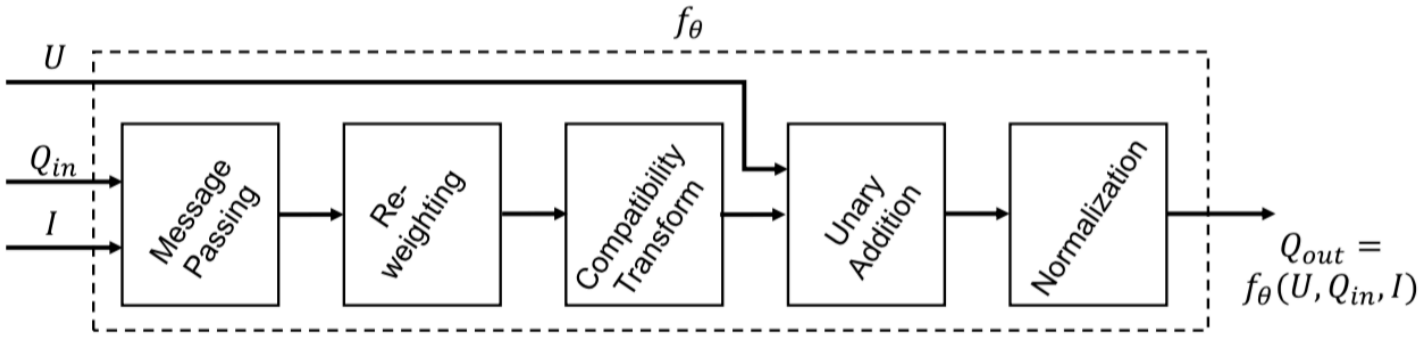
Zheng et al., 2015 "Conditional Random Fields as Recurrent Neural Networks"

Goal: image segmentation

Model: fully-connected CRF

Baseline Prediction Algorithm: mean-field inference for fixed # iterations.

Neural Network Representation:



Every step in mean-field is differentiable, so you can train directly wrt the downstream loss.

Iterative Prediction using Neural Networks

- Siamese networks

  Bromley et al., 1993, Le and Mikolov, 2014

- Generating adversarial examples

  Szegedy et al., 2014, Goodfellow et al., 2014

- Image generation and texture synthesis

  Mordvintsev et al., 2015; Gatys et al., 2015a;b

- Generating prediction increments directly using a NN: $y_{t+1} = y_t + \Delta(x, y_t)$

  Carreira et al., 2015

Multi-Label Classification

- Better losses

  Elisseeff & Weston, 2001; Godbole & Sarawagi, 2004; Zhang & Zhou, 2006; Bucak et al.,2009

- Label Embeddings (low-rank parameters)

  Ji & Ye, 2009; Cabral et al., 2011; Yu et al.,2014; Xu et al., 2014; Bhatia et al., 2015

- Pairwise CRFs

  Ghamrawi & McCallum, 2005; Finley & Joachims, 2008; Meshi et al., 2010; Petterson & Caetano, 2011

- Structured Prediction with Simple, Pre-Supposed Structure

  Read et al., 2011; Jasinska & Dembczyski, 2015; Niculescu-Mizil & Abbasnejad, 2015

- Compressive Sensing / Error Correcting Codes

  Hsu et al., 2009; Hariharan et al., 2010; Kapoor et al., 2012

EXPERIMENTS

Multi-Label Classification Benchmarks

- 3 NLP tasks
- Hundreds of labels
- Hundreds to thousands of sparse features
- Very sparse label vectors
- Some have 'positive only' annotation
  - Missing data modeling (Bucak et al., 2011; Agrawal et al., 2013; Lin et al., 2014)

	#labels	#features	# train	% true labels
Bibtex	159	1836	4880	2.40
Delicious	983	500	12920	19.02
Bookmarks	208	2150	60000	2.03

Multi-Label Classification Benchmarks

BR: Per-label logistic regression
LR: logistic regression with low-rank weights
MLP: 3-layer ReLU NN trained with logistic loss
SPEN: same feature network as MLP

	BR	LR	MLP	SPEN
Bibtex	37.2	39.0	38.9	**42.2**
Delicious	26.5	35.3	**37.0**	35.2
Bookmarks	30.7	31.0	33.8	**34.4**

Modeling missing data:

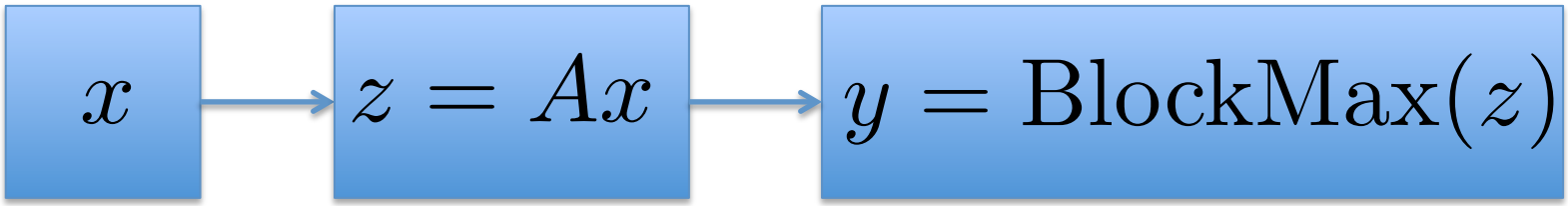
Provides improvements on some tasks (depends on how they were annotated).
Eg., Lin et al. get 44.2 on Bibtex, but only 33.3 on Delicious.

Structure Learning with SPENs

Experiment:

1. Generate data with known interactions between labels
2. Fit a SPEN
3. Inspect the measurement matrix and see if we recovered the interactions

Synthetic Data



64 dim

16 dim

16 dim

BlockMax function:

1) Split z into blocks

$$z = \left[\cdot\cdot\cdot\cdot \,\middle|\, \cdot\cdot\cdot\cdot \,\middle|\, \cdot\cdot\cdot\cdot \,\middle|\, \cdot\cdot\cdot\cdot\right]$$

2) Identify the max value in each block

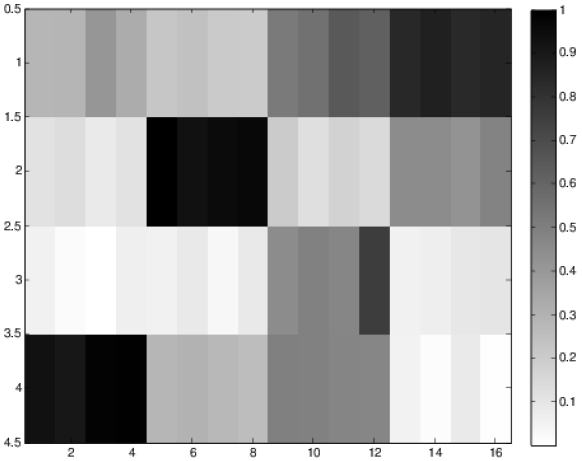
$$y = [\,0\,0\,1\,0 \,|\, 1\,0\,0\,0 \,|\, 0\,0\,1\,0 \,|\, 0\,1\,0\,0\,]$$

Measurement Matrix

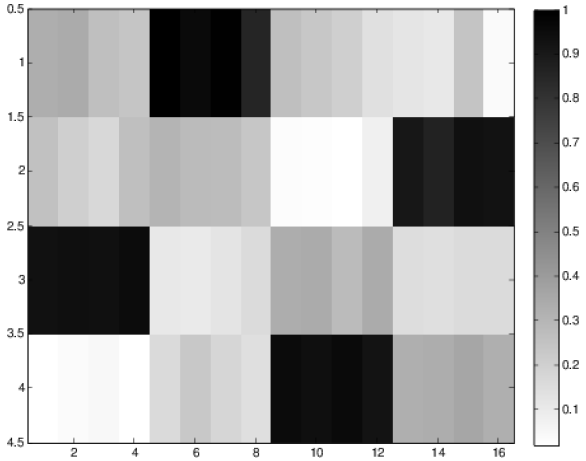
$$E_x^{\text{local}}(\bar{y}) = \sum_{i=1}^{L} \bar{y}_i b_i^\top f(x)$$

linear

$$E_x^{\text{label}}(\bar{y}) = c_2^\top g(C_1 \bar{y})$$



ReLU



HardTanh

Accuracy on the Synthetic Task

Questions:

1. Is it even possible for a feed-forward predictor to capture this block structure?
2. What's the most parsimonious way to model this data?

# train examples	Linear	3-Layer MLP	SPEN w/ Linear Local Energy
1.5k	80.0	81.6	**91.5**
15k	81.8	96.3	**96.7**

Observations:

1. SPEN can generalize better from much less data.
2. The MLP can capture the data constraints, given enough examples.

Conclusion

Prior Structured Prediction Work

- steep tradeoff between tractability and model expressiveness.
- model selection is replaced by algorithm selection.
- structure learning avoided, since it can interfere with tractability
- deep learning features for x, but not for y

Structured Prediction Energy Networks

- simple prediction algorithm
- per-iteration complexity doesn't depend on treewidth
- structure learning using deep architecture
- deep learning features for x
- fewer guarantees than CRFs
- Applicable to wide variety of structured prediction tasks

Future Applications

- Entity type tagging
- Vision problems
  - denoising, segmentation, etc.
- Structure learning for time series

More Future Work

- New training techniques
  - Use the ‘unrolling’ approach
- Avoid ‘curse of last reducer’
- How to make measurement matrix most interpretable
- Speed up prediction by pushing the feature representation to capture more structure.

THANKS!