Deep Learning meets Structured Prediction

Alexander G. Schwing

in collaboration with T. Hazan, M. Pollefeys and R. Urtasun
in collaboration with L.-C. Chen, A. L. Yuille and R. Urtasun

University of Toronto

Big Data and Statistical Machine Learning

Large scale problems according to the program:

- input dimensionality
- number of training samples
- number of categories

Scene understanding

$x =$ image

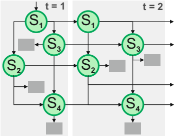
$s \in \mathcal{S}$: room layout

Scene understanding

$x =$ image

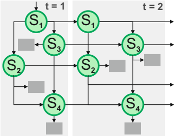
$s \in \mathcal{S}$: room layout

Tutoring systems



$x =$ responses

$s \in \mathcal{S}$: student skills

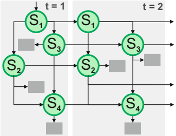
Scene understanding	Tutoring systems	Tag prediction
		
$x =$ image	$x =$ responses	$x =$ image
$s \in \mathcal{S}$: room layout	$s \in \mathcal{S}$: student skills	$s \in \mathcal{S}$: tags

Scene understanding

x = image

$s \in \mathcal{S}$: room layout

Tutoring systems



x = responses

$s \in \mathcal{S}$: student skills

Tag prediction

x = image

$s \in \mathcal{S}$: tags

Large scale problems

- input dimensionality — x is large
- number of training samples — $|\mathcal{D}| = |\{(x, s)\}|$ is large
- number of categories — $|\mathcal{S}|$ is large

Why is large scale a challenge

Inference:

$$s^* = \arg\max_{s \in \mathcal{S}} F(s, x, w)$$

Why is large scale a challenge

Inference:

$$s^* = \arg\max_{s \in \mathcal{S}} F(s, x, w)$$

- Search over output space $\mathcal{S}$
- Computation of $F(s, x, w)$ from data x

Why is large scale a challenge

Inference:

$$s^* = \arg\max_{s\in\mathcal{S}} F(s,x,w)$$

- Search over output space $\mathcal{S}$
- Computation of $F(s,x,w)$ from data x

Learning:

$$w^* = \arg\max_{w} \sum_{(x,s)\in\mathcal{D}} \left(F(s,x,w) - \ln \sum_{\hat{s}\in\mathcal{S}} \exp F(s,x,w) \right)$$

- Summation over output space $\mathcal{S}$
- Summation over dataset $\mathcal{D}$
- Computation of $F(s,x,w)$ from data x

How we deal with the challenges?

- Inference:

  How to find the maximizer of $F(s, x, w)$ given x, w?
- Learning:

  How to find the parameters w of $F(s, x, w)$ given $\mathcal{D}$?

Inference

$$s^* = \arg\max_{s \in \mathcal{S}} F(s, x, w)$$

The domain size $|\mathcal{S}|$ is potentially large

- ImageNet challenge: $|\mathcal{S}| = 1000$
- Layout prediction: $|\mathcal{S}| = 50^4$
- Tutoring systems: $|\mathcal{S}| = 2^{\text{Number of modeled skills}}$
- Tag prediction: $|\mathcal{S}| = 2^{\text{Number of tags}}$

Computation of $F(s, x, w)$ for all possible $s \in \mathcal{S}$ in general often intractable.

But: Interest in jointly predicting multiple variables $s = (s_1, \ldots, s_n)$

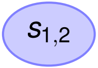
Assumption: function/model decomposes additively

$$F(s, x, w) = F(s_1, \dots, s_n, x, w) = \sum_r f_r(s_r, x, w)$$

- Restriction r: $s_r = (s_i)_{i \in r}$
- Discrete domain:

$$f_{\{1,2\}}(s_{\{1,2\}}) = f_{\{1,2\}}(s_1, s_2) = \left[f_{\{1,2\}}(1,1), f_{\{1,2\}}(1,2), \dots\right]$$

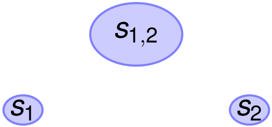
- Visualization



[Werner‘07; Koller&Friedman‘09]

Example

$$s^* = \arg\max_s \sum_r f_r(s_r)$$

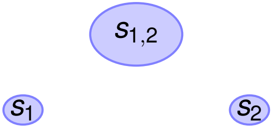


Integer Linear Program (LP) equivalence: variables $b_r(s_r)$

$$\max_{b_1,b_2,b_{12}} \begin{bmatrix} b_1(0) \\ b_1(1) \\ b_2(0) \\ b_2(1) \\ b_{12}(0,0) \\ b_{12}(1,0) \\ b_{12}(0,1) \\ b_{12}(1,1) \end{bmatrix}^\top \begin{bmatrix} f_1(0) \\ f_1(1) \\ f_2(0) \\ f_2(1) \\ f_{12}(0,0) \\ f_{12}(1,0) \\ f_{12}(0,1) \\ f_{12}(1,1) \end{bmatrix}$$

Example

$$s^* = \arg\max_s \sum_r f_r(s_r)$$



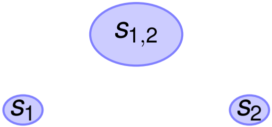
Integer Linear Program (LP) equivalence: variables $b_r(s_r)$

$$\max_{b_1,b_2,b_{12}} \begin{bmatrix} b_1(0) \\ b_1(1) \\ b_2(0) \\ b_2(1) \\ b_{12}(0,0) \\ b_{12}(1,0) \\ b_{12}(0,1) \\ b_{12}(1,1) \end{bmatrix}^\top \begin{bmatrix} f_1(0) \\ f_1(1) \\ f_2(0) \\ f_2(1) \\ f_{12}(0,0) \\ f_{12}(1,0) \\ f_{12}(0,1) \\ f_{12}(1,1) \end{bmatrix} \quad \text{s.t.} \quad b_r(s_r) \in \{0,1\}$$

[Werner'07; Koller&Friedman'09]

Example

$$s^* = \arg\max_s \sum_r f_r(s_r)$$

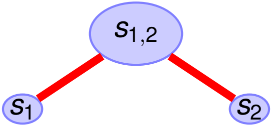


Integer Linear Program (LP) equivalence: variables $b_r(s_r)$

$$\max_{b_1,b_2,b_{12}} \begin{bmatrix} b_1(0) \\ b_1(1) \\ b_2(0) \\ b_2(1) \\ b_{12}(0,0) \\ b_{12}(1,0) \\ b_{12}(0,1) \\ b_{12}(1,1) \end{bmatrix}^\top \begin{bmatrix} f_1(0) \\ f_1(1) \\ f_2(0) \\ f_2(1) \\ f_{12}(0,0) \\ f_{12}(1,0) \\ f_{12}(0,1) \\ f_{12}(1,1) \end{bmatrix} \quad \text{s.t.} \quad \begin{array}{c} b_r(s_r) \in \{0,1\} \\ \sum_{s_r} b_r(s_r) = 1 \end{array}$$

Example

$$s^* = \arg\max_s \sum_r f_r(s_r)$$



Integer Linear Program (LP) equivalence: variables $b_r(s_r)$

$$\max_{b_1,b_2,b_{12}} \begin{bmatrix} b_1(0) \\ b_1(1) \\ b_2(0) \\ b_2(1) \\ b_{12}(0,0) \\ b_{12}(1,0) \\ b_{12}(0,1) \\ b_{12}(1,1) \end{bmatrix}^\top \begin{bmatrix} f_1(0) \\ f_1(1) \\ f_2(0) \\ f_2(1) \\ f_{12}(0,0) \\ f_{12}(1,0) \\ f_{12}(0,1) \\ f_{12}(1,1) \end{bmatrix} \quad \text{s.t.} \quad \begin{array}{l} b_r(s_r) \in \{0,1\} \\ \sum_{s_r} b_r(s_r) = 1 \\ \sum_{s_p \backslash s_r} b_p(s_p) = b_r(s_r) \end{array}$$

$$\hat{s} = \arg\max_s \sum_r f_r(s_r)$$

Integer linear program:

$$\max_{b_1,b_2,b_{12}} \begin{bmatrix} b_1(1) \\ b_1(2) \\ b_2(1) \\ b_2(2) \\ b_{12}(1,1) \\ b_{12}(2,1) \\ b_{12}(1,2) \\ b_{12}(2,2) \end{bmatrix}^\top \begin{bmatrix} f_1(1) \\ f_1(2) \\ f_2(1) \\ f_2(2) \\ f_{12}(1,1) \\ f_{12}(2,1) \\ f_{12}(1,2) \\ f_{12}(2,2) \end{bmatrix} \quad \text{s.t.} \quad \begin{array}{l} b_r(s_r) \in \{0,1\} \\ b_r(s_r) \geq 0 \\ \sum_{s_r} b_r(s_r) = 1 \\ \sum_{s_p \setminus s_r} b_p(s_p) = b_r(s_r) \end{array}$$

$$\hat{s} = \arg\max_s \sum_r f_r(s_r)$$

Integer linear program:

$$\max_{b_r} \quad \sum_{r,s_r} b_r(s_r) f_r(s_r) \quad \text{s.t.} \quad \begin{array}{l} b_r(s_r) \in \{0,1\} \\ b_r(s_r) \geq 0 \\ \sum_{s_r} b_r(s_r) = 1 \\ \sum_{s_p \setminus s_r} b_p(s_p) = b_r(s_r) \end{array}$$

$$\hat{s} = \arg\max_{s} \sum_{r} f_r(s_r)$$

Integer linear program:

$$\max_{b_r} \quad \sum_{r,s_r} b_r(s_r) f_r(s_r) \quad \text{s.t.} \quad \begin{array}{l} b_r(s_r) \in \{0,1\} \\ b_r(s_r) \geq 0 \\ \displaystyle\sum_{s_r} b_r(s_r) = 1 \end{array}$$

Marginalization

$$\hat{s} = \arg\max_s \sum_r f_r(s_r)$$

Integer linear program:

$$b_r(s_r) \in \{0, 1\}$$

$$\max_{b_r} \quad \sum_{r,s_r} b_r(s_r) f_r(s_r) \quad \text{s.t.} \quad \text{Local probability } b_r$$

Marginalization

$$\hat{s} = \arg\max_s \sum_r f_r(s_r)$$

LP relaxation:

$$\max_{b_r} \sum_{r,s_r} b_r(s_r) f_r(s_r) \quad \text{s.t.} \quad \cancel{b_r(s_r) \in \{0,1\}}$$

Local probability b_r

Marginalization

$$\hat{s} = \arg\max_s \sum_r f_r(s_r)$$

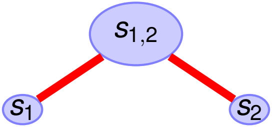
LP relaxation:

$$\max_{b_r} \sum_{r,s_r} b_r(s_r) f_r(s_r) \quad \text{s.t.} \quad \begin{array}{l} \cancel{b_r(s_r) \in \{0,1\}} \\ \text{Local probability } b_r \\ \text{Marginalization} \end{array}$$

Standard LP solvers are **slow** because of many variables and constraints. Specifically tailored algorithms...

[Weiss et al.'07, Globerson&Jaakkola'07, Johnson'08, Jojic et al.'10, Hazan&Shashua'10, Savchynskyy et al.'12, Ravikumar et al.'10, Martins et al.'11, Meshi&Globerson'11, Komodakis et al.'10'12, Schwing et al.'11'12'14]

Graph structure defined via marginalization constraints



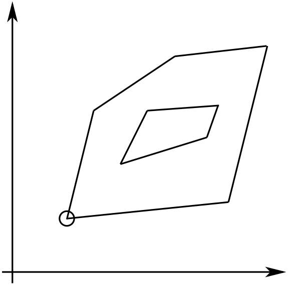
- Message passing solvers
  Advantage: Efficient due to analytically computable sub-problems
  Problem: Special care required to find global optimum
- Subgradient methods
  Advantage: Guaranteed globally convergent
  Problem: Special care required to find fast algorithms

Block-coordinate ascent/message passing solvers

[Weiss et al.'07, Globerson&Jaakkola'07, Johnson'08, Jojic et al.'10, Hazan&Shashua'10, Savchynskyy et al.'12, Ravikumar et al.'10, Martins et al.'11, Meshi&Globerson'11]

Optimize w.r.t. subset of variables

Advantage: Efficient due to analytically computable sub-problems
Problem: Getting stuck in corners



Smoothing, proximal updates, augmented Lagrangian methods

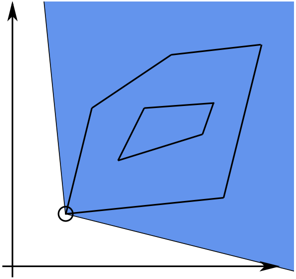
[Komodakis et al.'10'12, Kappes et al.'12]

Subgradient Methods

Update Lagrange multipliers via any subgradient direction

Advantage: Globally convergent

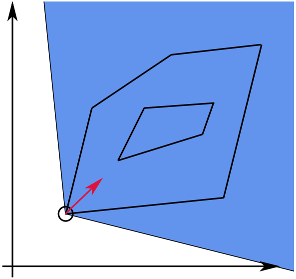
Problem: Slow and non-monotone convergence



Subgradient Methods

Update Lagrange multipliers via any subgradient direction

Advantage: Globally convergent
Problem: Slow and non-monotone convergence



What we like: steepest subgradient ascent direction

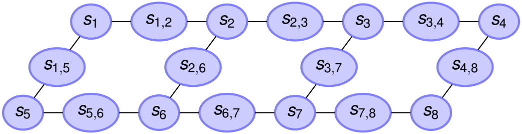
Distributed Inference for Graphical Models

Goal:

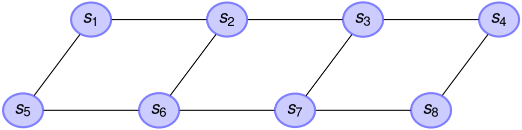
- Optimize the LP relaxation objective
- Leverage the problem structure
- Distribute memory and computation requirements
- **Maintain convergence and optimality guarantees**

Dual decomposition extension of LP relaxation solvers via partitioning of variables.

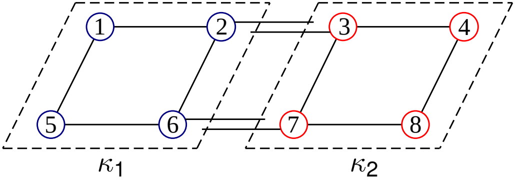
- Dual decomposition intuition:



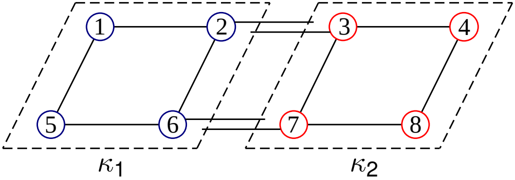
- Dual decomposition intuition:



- Dual decomposition intuition:



- Dual decomposition intuition:

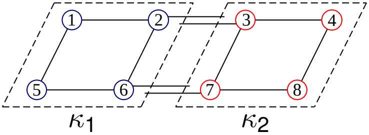


- Unique set of variables:

$$b_r^\kappa(s_r)$$

- Marginalization constraints and local beliefs $\forall\kappa$
- Consistency constraints:

$$b_r^\kappa(s_r) = b_r(s_r)$$



Distributed LP Relaxation

$$\max_b \sum_\kappa \left(\sum_{r\in\kappa, s_r} b_r^\kappa(s_r)\hat{f}_r(s_r) \right)$$

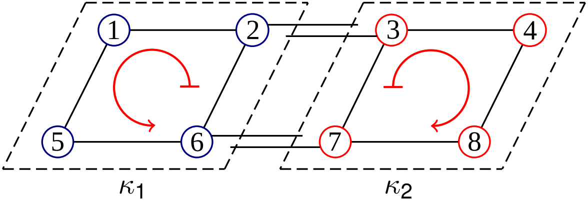
s.t. $\forall\kappa$ Local probabilities b_r^κ

$\forall\kappa$ Marginalization constraints

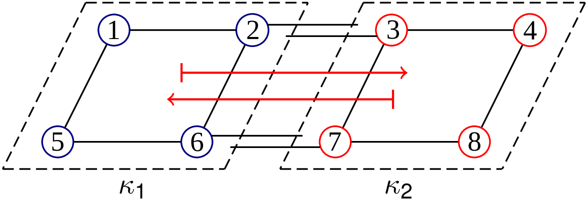
$\forall\kappa$ Consistency

Algorithm:

- Some message passing iterations in parallel $\forall \kappa$



- Exchange of information between different κ

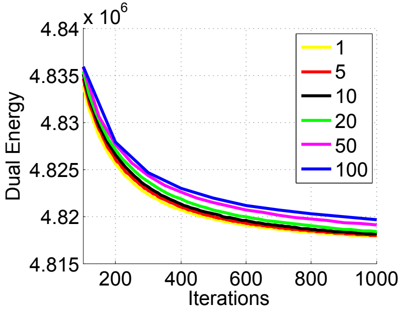


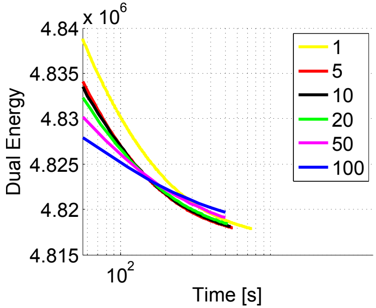
State-of-the-art

- libDAI 0.2.7 [Mooij 2010]
- graphLAB [Low et al. 2010]

Method	Runtime [s]	Efficiency [nodes/μs]	primal
BP (libDAI)	2617/4	0.04	1.0241
BP (graphLAB RR)	1800	0.06	1.0113
SplashBP (graphLAB)	689	0.06	1.0121
Ours (General)	371	0.29	**1.0450**
Ours (Ded.)	113	0.94	**1.0450**
Ours dist. (Ded.)	18	5.8	1.0449

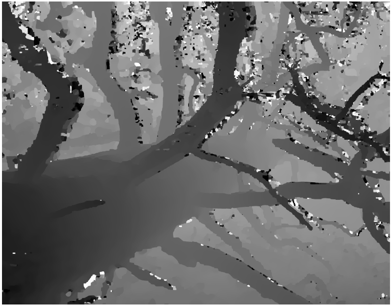
Inter-machine communication





Large-scale setting

- x is large (> 12 MPixel image)
- $|\mathcal{S}|$ is large ($280^{12,000,000}$)
- s_r is large (> 12 million regions with 280 states)
- s_r is large (> 24 million regions with 80k states)



Sources publicly available on http://alexander-schwing.de

How we deal with the challenges?

- Inference:

  How to find the maximizer of $F(s, x, w)$ given x, w?
- Learning:

  How to find the parameters w of $F(s, x, w)$ given $\mathcal{D}$?

Learning

good parameters from annotated examples

$$\mathcal{D} = \{(x, s)\}$$

- Log-linear models (CRFs, structured SVMs):

$$F(s, x, w) = w^\top \tilde{F}(s, x)$$

- Non-linear models, e.g., CNNs (this talk):

$$F(s, x, w)$$

Inference:

$$s^* = \arg\max_{s \in \mathcal{S}} F(s, x, w)$$

Probability of a configuration s:

$$p(s \mid x, w) = \frac{1}{Z(x, w)} \exp F(s, x, w)$$

$$Z(x, w) = \sum_{\hat{s} \in \mathcal{S}} \exp F(\hat{s}, x, w)$$

Inference alternatively:

$$s^* = \arg\max_{s \in \mathcal{S}} p(s \mid x, w)$$

Probability of a configuration s:

$$p(s \mid x, w) = \frac{1}{Z(x,w)} \exp F(s,x,w)$$

$$Z(x,w) = \sum_{\hat{s}\in\mathcal{S}} \exp F(\hat{s},x,w)$$

Maximize the likelihood of training data via

$$\begin{aligned} w^* &= \arg\max_w \prod_{(x,s)\in\mathcal{D}} p(s|x,w) \\ &= \arg\max_w \sum_{(x,s)\in\mathcal{D}} \left(F(s,x,w) - \ln \sum_{\hat{s}\in\mathcal{S}} \exp F(s,x,w) \right) \end{aligned}$$

Maximum likelihood is equivalent to maximizing cross-entropy:

- Target distribution: $p_{(x,s),\text{tg}}(\hat{s}) = \delta(\hat{s} = s)$
- Cross-Entropy:

$$\begin{aligned}
& \max_w \sum_{(x,s)\in\mathcal{D},\hat{s}} p_{(x,s),\text{tg}}(\hat{s}) \ln p(\hat{s} \mid x; w) \\
= \quad & \max_w \sum_{(x,s)\in\mathcal{D}} \ln p(s \mid x; w) \\
= \quad & \max_w \ln \prod_{(x,s)\in\mathcal{D}} p(s \mid x; w)
\end{aligned}$$

Program of interest:

$$\max_{w} \sum_{(x,s)\in\mathcal{D},\hat{s}} p_{(x,s),\text{tg}}(\hat{s}) \ln p(\hat{s} \mid x; w)$$

Optimize via gradient ascent

$$\begin{aligned}
\frac{\partial}{\partial w} & \sum_{(x,s)\in\mathcal{D},\hat{s}} p_{(x,s),\text{tg}}(\hat{s}) \ln p(\hat{s} \mid x; w) \\
&= \sum_{(x,s)\in\mathcal{D},\hat{s}} \left(p_{(x,s),\text{tg}}(\hat{s}) - p(\hat{s} \mid x; w)\right) \frac{\partial}{\partial w} F(\hat{s}, x, w)
\end{aligned}$$

- Compute predicted distribution $p(\hat{s} \mid x; w)$
- Use chain rule to pass back difference between prediction and observation

Algorithm: Deep Learning

Repeat until stopping criteria

1. Forward pass to compute $F(s, x, w)$
2. Compute $p(s \mid x, w)$
3. Backward pass via chain rule to obtain gradient
4. Update parameters w

Why is large scale data a challenge?

Algorithm: Deep Learning

Repeat until stopping criteria

1. Forward pass to compute $F(s, x, w)$
2. Compute $p(s \mid x, w)$
3. Backward pass via chain rule to obtain gradient
4. Update parameters w

Why is large scale data a challenge?

- How do we even represent $F(s, x, w)$ if $\mathcal{S}$ is large?
- How do we compute $p(s \mid x, w)$?

Domain size of typical applications:

- ImageNet challenge: $|\mathcal{S}| = 1000$
- Layout prediction: $|\mathcal{S}| = 50^4$
- Tutoring systems: $|\mathcal{S}| = 2^{\text{Number of modeled skills}}$
- Tag prediction: $|\mathcal{S}| = 2^{\text{Number of tags}}$

Solution:

- Interest in jointly predicting multiple variables $s = (s_1, \ldots, s_n)$
- Assumption: function/model decomposes additively

$$F(s, x, w) = F(s_1, \ldots, s_n, x, w) = \sum_r f_r(s_r, x, w)$$

Every $f_r(s_r, x, w)$ is an arbitrary function, e.g., a CNN

How to compute gradient:

$$\frac{\partial}{\partial w} \sum_{(x,s)\in\mathcal{D},\hat{s}} p_{(x,s),\text{tg}}(\hat{s}) \ln p(\hat{s} \mid x; w)$$

$$= \sum_{(x,s)\in\mathcal{D},\hat{s}} \left(p_{(x,s),\text{tg}}(\hat{s}) - p(\hat{s} \mid x; w)\right) \frac{\partial}{\partial w} F(\hat{s}, x, w)$$

$$= \sum_{(x,s)\in\mathcal{D},r,\hat{s}_r} \left(p_{(x,s),r,\text{tg}}(\hat{s}_r) - p_r(\hat{s}_r \mid x; w)\right) \frac{\partial}{\partial w} f_r(\hat{s}_r, x, w)$$

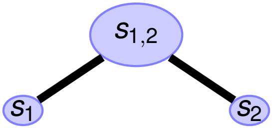
How to obtain marginals $p_r(\hat{s}_r|x, w)$?

Approximation of marginals via:

- Sampling methods
- Inference methods

Inference approximations:

$$\underbrace{\max_{b\in\mathcal{L}} \sum_{r,s_r} b_r(s_r \mid x, w) f_r(s_r, x, w)}_{\textit{Inference}} + \sum_r c_r H(b_r)$$



Typically employed variational algorithms:

- Convex Belief Propagation (distributed)
- Tree-reweighted message passing
- (Generalized) Loopy Belief Propagation
- (Generalized) double loop Loopy Belief Propagation

Approximated Deep Structured Learning

Repeat until stopping criteria

1. CNN Forward pass to compute $f_r(\hat{s}_r, x, w)\ \forall r$
2. Compute approximate beliefs $b_r(\hat{s}_r \mid x, w)$
3. Backward pass via chain rule to obtain gradient g
4. Update parameters w

$$g = \sum_{(x,s)\in\mathcal{D},r,\hat{s}_r} \left(p_{(x,s),r,\text{tg}}(\hat{s}_r) - b_r(\hat{s}_r \mid x, w)\right) \frac{\partial}{\partial w} f_r(\hat{s}_r, x, w)$$

Dealing with large number $|\mathcal{D}|$ of training examples:

- Parallelized across samples (any number of machines and GPUs)
- Usage of mini batches

Dealing with large input dimension x:

- Usage of standard CNNs
- GPU and CPU implementation

Dealing with large output spaces $\mathcal{S}$:

- Variational approximations
- Blending of learning and inference

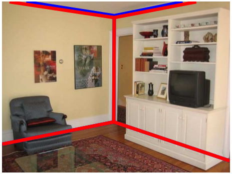
ImageNet dataset

- $|\mathcal{S}| = 1000$
- 1.2 million training examples
- 50,000 validation examples

Model	Validation set error [%]
AlexNet	19.95
DeepNet16	10.29
DeepNet19	10.37

Different from reported results because of missing averaging, different image crops, etc.

Layout dataset
Given a single image x, predict a 3D parametric box that best describes the observed room layout



- $|\mathcal{S}| = 50^4$
- Linear model
- 205 training examples
- 104 test examples

Pixel-wise prediction errors [%] on layout dataset:

	OM	GC	OM + GC	Others
[Hoiem07]	-	28.9	-	-
[Hedau09]	-	21.2	-	-
[Wang10]	22.2	-	-	-
[Lee10]	24.7	22.7	18.6	-
[Pero12]	-	-	-	16.3
Ours	**18.63**	**15.35**	**13.59**	-

Flickr dataset

- $|\mathcal{S}| = 2^{38}$
- 10000 training examples
- 10000 test examples

Training method	Prediction error [%]
Unary only	9.36
Piecewise	7.70
Joint (with pre-training)	7.25

Visual results

female/indoor/portrait
female/indoor/portrait

sky/plant life/tree
sky/plant life/tree

water/animals/sea
water/animals/sky

animals/dog/indoor
animals/dog

indoor/flower/plant life
$\emptyset$

Learnt class correlations:

	female	people	indoor	portrait	sky	plant life	male	clouds	tree
female	0.00	0.68	0.04	0.24	-0.01	-0.05	0.07	-0.01	0.01
people	0.68	0.00	0.06	0.36	-0.05	-0.12	0.74	-0.04	-0.03
indoor	0.04	0.06	0.00	0.07	-0.35	-0.34	0.02	-0.15	-0.21
portrait	0.24	0.36	0.07	0.00	-0.02	-0.01	0.12	0.02	0.05
sky	-0.01	-0.05	-0.35	-0.02	0.00	0.24	-0.00	0.44	0.30
plant life	-0.05	-0.12	-0.34	-0.01	0.24	0.00	-0.07	0.09	0.68
male	0.07	0.74	0.02	0.12	-0.00	-0.07	0.00	0.00	-0.02
clouds	-0.01	-0.04	-0.15	0.02	0.44	0.09	0.00	0.00	0.11
tree	0.01	-0.03	-0.21	0.05	0.30	0.68	-0.02	0.11	0.00

Distributed Inference

- Maintaining convergence guarantees
- Parallelized across computers

Deep Nonlinear Structured Prediction

- Nonlinearity, e.g., via CNNs in every factor
- Unifying structured prediction

Future directions

- Applications
- Effects of approximations
- Latent variable models
- Can we include the optimization of the model hyper-parameters
- Time series data