Introduction to Graphical Models

Alexander Shekhovtsov

2018

https://cw.fel.cvut.cz/wiki/courses/ucuss18
Contents

• Introduction
  • Probabilistic models
    • Decisions under uncertainty
    • Different inference tasks
  • What are graphical models
    • Dependance diagrams, conditional independence
    • Types of graphical models
    • Applications
  • Hidden Markov Model
    • Markov chain, different factorizations
    • MAP problem — Viterbi algorithm
    • Marginals problem — forward-backward algorithm
    • Generalized dynamic programming
    • Further topics
  • Markov Random Field
    • Markov Properties and factorization
    • Conditional Random Filed
    • Examples
  • MAP in MRF
    • Energy minimization, solvable classes, graph cuts, relaxations
  • Marginals in MRF
    • Forward and Reverse KL Divergence
    • Mean Field
    • Dense (fully connected) CRF
  • Graphical models as neural networks
    • Unrolled inference
    • Examples of NN + CRF applications
  • Neural Networks as graphical models
    • Sigmoid Belief network
    • Uncertainties, noises
    • Variance propagation methods
  • Bayesian Learning
    • Variational Bayesian learning
    • Stochastic gradient variational Bayesian learning
Introduction

Bayesian Networks

NNs as GMs

HMM

Bayesian Learning

MRF

GMs as NNs

MRF Marginals: Mean Field

GMs as NNs

MRF MAP

Bayesian Learning

MRF MAP
Introduction: What are Graphical Models
Basic Classification Problems

Two-class \{0, 1\} \quad \text{Multi-class} \{1,..K\}

Classification Using Discriminant Functions

- **SVMs**
  - Design measurements, represent them as a feature vector
  - Learn the best discriminant function

- **Deep NNs (simplified view)**
  - Learn deep feature vectors
  - Apply SVM

\[ f(x, w, b) = \text{sign}(h \cdot x + b) = \text{sign}(w^T x + b) \]

R \subseteq \mathbb{R}^m \quad \text{VC dimension (capacity) depends on}

\[ m \leq \frac{R^2}{2} m^2 + 1 \]

\[ R \text{ is given by the data itself.} \]

\[ \text{Margin} \quad m \text{ can be optimized in the classifier design.} \]

Conclusion: separation hyperplanes with larger margin have lower VC dimension, lower value of the upper bound.
Motivation for Graphical Models I: Structured Predictions

- Text Recognition
- Optical Structure Recognition
- Body Parts Segmentation

{space of text sentences}
- Image Segmentation
- Landmarks and Parts Detection
Motivation for Graphical Models II: Probabilistic Reasoning

• Example: Medical Diagnosis
• Knowing the observed variables and conditional probabilities find the likely cause

[Lauritzen and Speigelhalter 1988]

• Originally, such diagrams and methods were used by experts with pen and paper...
• When do probabilities occur?
  • As a result of randomness such as thermal noise, but not only…
  • A way to represent information

• Example 1: information about population height
  • Average human height is 162 cm (single number)
  • Human height is from 54 to 272 cm (interval)
  • Fraction of population of a given height
    • contains more information
    • more information => better solutions
    • defines a probability distribution
• We represent the information with probabilities $p(x)$

• Some new fact(s) need to be taken into account, e.g. male / female

• Refine the available information, $p(x|A)$

• Suppose also person weight is known

Statistical Models

random person in the world

Suppose also person weight is known

weight

hight

Refine further
Statistical Models

- Example 2: non-functional dependencies

Hidden state = patient’s brain

MRI 1 — function

MRI 2 — function, different exposure time

Dependence of MRI 1 on MRI 2 is not a function! $Y = f(X)$?

Can be described as conditional probability distribution $p(Y | X)$
\( \omega \in \Omega \) – elementary event
\( A \subset \Omega \) – event
\( P : \Omega \to [0, 1] \) – probability measure
\( X : \Omega \to \mathcal{X} \) – random variable
\( x \in \mathcal{X} \) – a value that r.v. \( X \) may take
\( X = x \) – all elementary events that map to \( x \):
\[ \{ \omega \in \Omega \mid X(\omega) = x \} \] – an event
\( P(A) \) ✓
\( P(X) \) ✗
\( P(X = x) \) ✓
\( p_X : \mathcal{X} \to [0, 1] \) – density (or p.m.f.) of \( X \)
\( p_X(x) \) – density at point \( x \)

This lecture:
\( P(X=x, Y=y, Z=z) \) is abbreviated as \( P(x, y, z) \)
\( p_{X,Y|Z}(x, y|z) \) is abbreviated as \( p(x, y|z) \) or as \( p(X=1, y|z) \) when ambiguous
\( p(X) \) will denote \( p_X(X) \) – the “whole density function”
(technically a composition: \( \Omega \xrightarrow{X} \mathcal{X} \xrightarrow{p_X} [0, 1] \))
Two classes to recognize: $k \in \{0,1\}$  
- take some measurement $x$, for example thickness

Observe $X=5$, which fish is it?
- What if salmons are extremely rare in your lake?
- Need to know probabilities $p(K)$ of fish occurrence
  - $p(K=0) = 0.15$, $p(K=1) = 0.85$
  - So what do we do with these numbers?

Simple Classification Example

Known statistics $p(X|k)$ for $k=0,1$

$p(\text{measurement} \mid \text{knowing the class})$
For simple classification the denominator does not matter:

\[ p(K=0 \mid x) \geq p(K=1 \mid x) \iff \frac{p(K=0 \mid x)}{p(K=1 \mid x)} \geq 1 \iff \frac{p(x \mid K=0)}{p(x \mid K=1)} \geq \frac{p(K=1)}{p(K=0)} = \theta \]

If we have utilities (risks) or want to quantify uncertainty, need posterior probabilities:

\[ p(K=0 \mid x) = 0.52, \quad p(K=1 \mid x) = 0.47 \]
• Experiment: flipping a coin

\[ K \in \{\text{Heads, Tails}\} \]

\[ P(K=\text{Heads}) = p \]

\[ P(K=\text{Tails}) = 1 - p \]

\( p \) is unknown

• Suppose you tried 20 times and observed: 18 H and 2 T

• What you can say about \( p \)?
  • \( 0 < p < 1 \) (strictly)
  • it is more likely that \( p \) is closer to 0.9
  • but other values of \( p \), including 1/2 are not excluded...

• Bayes has proposed to assign probabilities to \( p \) considered as beliefs (the information that we have about \( p \))
• Recall axioms of the probability theory:

**Axiom 1:** $0 \leq P(A) \leq 1$, with $P(A) = 1$ if $A$ is certain

**Axiom 2:** If events $(A_i)$, $i = 1, 2, \ldots$ are pairwise incompatible (exclusive) then $P(\bigcup_i A_i) = \sum_i P(A_i)$

**Axiom 3:** $P(A \cap B) = P(B \mid A)P(A)$

• Proofs exist that these rules are necessary
  “if we want to assign numerical values to represent degrees of rational belief in a set of propositions” (Cox 1946).
Axiom 1: $0 \leq P(A) \leq 1$, with $P(A) = 1$ if $A$ is certain

Axiom 2: If events $(A_i)$, $i = 1, 2, \ldots$ are pairwise incompatible (exclusive) then $P(\bigcup_i A_i) = \sum_i P(A_i)$

Axiom 3: $P(A \cap B) = P(B \mid A)P(A)$

Exercise: prove the Bayes’ theorem:

$$P(A \mid B) = \frac{P(B \mid A)P(A)}{P(B)}$$

Now prove it without axioms?
Back in 1760s...

Richard Price

An Essay towards solving a Problem in the Doctrine of Chances, 1763

“...in the constitution of things fixt laws according to which things happen...
...and thus to confirm the argument taken from final causes for the existence of the Deity”

Richard Price

...Bicycle invented about 50 years later
Exercise

- Suppose we have a test for cancer with the following statistics:
  - The test was positive in 98% of cases when subjects had cancer
  - The test was negative in 97% of cases when subjects did not have cancer
  - Suppose that 0.1% of the entire population has this disease

- A patient takes a test. Compute
  - The probability that a person who test positive has this disease?
  - The probability that a person who test negative does not have this disease?

Variables: $C \in \{y, n\}$, $T \in \{+, -\}$
Probabilistic Models

- Observed variables:
  \( X_1, X_2, \ldots, X_n \); represented by vector \( X = (X_i \mid i = 1, \ldots n) \); Event \( X = x \) is denoted as \( x \)

- Hidden variables:
  \( K_1, K_2, \ldots, K_m \); represented by vector \( K = (K_i \mid i = 1, \ldots m) \)

(The naming / roles may differ depending on the context)

**Definition (Model)**

A probabilistic *model* is the joint probability distribution over a set of random variables. We assume the density \( p(X, K) \).

- Models describe how a part of the world works. Are always approximations or simplifications.

- Posterior inference task: Given \( X = x \), compute \( p(K \mid x) \)

- Maximum a posterior task (recognition): \( \arg\max_K p(K \mid x) \)

- Statistical decision making: \( \arg\min_d \sum_k \text{Risk}(d, k)p(k \mid x) \)
Example of Tasks

Model: $p(X, K)$
- Observation: $x = \text{(yes, yes, yes, no)}$
- Tasks:
  - Posterior: $p(K_3=\text{yes} \mid x)$ (belief in bronchitis)
  - MAP: most likely explanation: $\max_k p(k \mid x)$
  - Decision making: \{do nothing, heal 1,2,3, new analysis\}
- More general queries:
  - Suppose result of X-ray is not yet available,
    - what in the belief in bronchitis versus more serious problems?
    - what is the prediction for X-ray?
    - how much the belief in bronchitis depends on X-ray?
The Promise and The Catch

• Promises of probabilistic models:
  • A sound formulation for a system that can answer different kinds of queries:
    • recognition (likely cause)
    • handling missing data
    • prediction (likely symptoms)
    • “what if” queries
  • semi-supervised learning (parameters are random variables)
  • ...

• Obstacles:
  • Model representation
  • The problems that we can formulate mathematically are not necessarily solvable

• Looks like the right way to go, a major part in AI research
• With some hard work we get subclasses and approximations that are useful
• Probabilistic models are useful

• To represent the model in the example we need probabilities for all combinations of 8 Boolean variables:
  \[ p(X_1, X_2, X_3, X_4, K_1, K_2, K_3, K_4) \]
  \[ 2^8 = 256 \text{ numbers} \]
  • Becomes quickly intractable
  • to store / to learn

Trivial observation: If all variables are independent, the distribution factors as:
\[ p(X, K) = p(X_1)p(X_2)p(X_3)p(X_4)p(K_1)p(K_2)p(K_3)p(K_4) \]
Can be described by just 8 parameters. Something in between?
Conditional Independence

- Example: smoke, fire, alarm
- all 3 correlated, but
- given smoke $\Rightarrow$ fire and alarm are independent

$X$ and $Y$ with density $p(X, Y)$ are independent iff $p(x, y) = p(x)p(y)$ for all $x \in X$, $y \in Y$

- Conveniently represented with a graph diagram

\[
p(X_2, X_3|X_1) = p(X_2|X_1)p(X_3|X_1)
\]

- Factorization: $p(X_1, X_2, X_3) = p(X_2, X_3|X_1)p(X_1) = p(X_2|X_1)p(X_3|X_1)p(X_1)$

- A directed graphical model (Bayes Network)
Example:

- $X_i$ — weather state on day $i$
- Simplifying assumption: the weather on day $i$ depends only on the state on day $i-1$, but not $i-2$, ...

$$p(X_1, X_2, X_3, \ldots) = p(X_1)p(X_2 | X_1)p(X_3 | X_2) \ldots$$

State transition diagram: 0.5 (rain) 0.9 (sun)
Hidden Markov Model

- **Example:**
  - $S_i$ — letter in a sequence (hidden)
  - $X_i$ — observed images

- **Factorization:**
  \[
  p(X, S) = p(S_1) \prod_{i=2}^{n} p(S_i \mid S_{i-1}) \prod_{i=1}^{n} p(X_i \mid S_i)
  \]
• A region is independent of the rest given some neighborhood
Markov Random Field

- Example: 2D spin glass:
  - $X_i$ — spin orientation $\{-1,1\}$
  - Neighboring states “like” to be the same

- Local Markov Property w.r.t. $G$:
  - Given neighbors of $X_i$, it is independent of the rest.

- Pairwise Markov Property w.r.t. $G$:
  - Absent edge $(i,j)$ iff $X_i$ and $X_j$ are conditionally independent given the rest.

- Factorization:
  $$p(x) = \prod_{c \in \mathcal{C}(G)} g_c(x_c)$$
  (over cliques of $G$, more on this later)
Factor Graphs

- Factorization is another constructive way to define joint probability distribution than conditional independence

\[
p(X) = \frac{1}{Z} f_1(X_1) f_2(X_1, X_2) f_3(X_1, X_2) f_4(X_2, X_3)
\]

\(Z\) is the normalization factor, such that \(\sum_X p(X) = 1\)

- It is more general
- Inference algorithms often work directly with the factorization
- But:
  - more difficult to learn
    (c.f. conditional probabilities we could measure directly from the data)
Factor Graphs
• Coding
  • Sending $N$ bits over a noisy channel to decode $n$ bits
  • Shannon limit: codes exist with $n/N < \text{channel capacity}$ for arbitrary small error rate

• LDPCs: proposed by Robert Gallager in 1962
  • Good decoding algorithms found in 90’s
  • Appeared to be instances of Belief Propagation
  • Motivated a lot of research on BP

• Turbo Codes and LDPCs
  • 3G and 4G mobile standards
  • Digital video broadcasting
  • Satellite communication systems
  • …

• Current codes coming closer and closer to Shannon limit

[Low Density Parity Check Codes]

 received signal

Gaussian noise

Message bits

Parity bits

received

Reduced factor graph

[Daphne Coller, Coursera]
Example: joint probability $p(X,Y)$:
- $p(A,A) = 0.4$
- $p(A,B) = 0.1$
- $p(B,A) = 0.3$
- $p(B,B) = 0.2$

Goal: decide whether $X$ is $A$ or $B$ (say we win $1$ if we guess right)
- Approach 1: the most probable joint state is $AA$ -> decide for $A$
- Approach 2: compute marginal distribution $p(X)$ -> decide based on that

Continuous example:
- $X$ - face position, $Y$ - arm position
- Want to know face position

In practice, however we deal with approximation algorithms that behave poorly at high levels of uncertainty, anyhow.
Conclusion

- Summary
  - Probabilistic models describe how some part of world works
  - Well suited for reasoning with uncertainty and posing many recognition problems
  - Graphical Models are probabilistic models
    - Have an underlying graph-like structure
    - The structure is a way of simplification and is related to the structure of an application
    - Modeling is needed to come up with a good structure
    - The space complexity is tractable
  - Solving the recognition problems (the time complexity) may be difficult
  - But still often possible, areas of applications of GMs:
    - Computer Vision
    - Bioinformatics
    - Communications
    - ...
Hidden Markov Model
• Good for Classical Education
• Illustration of MAP and marginals problems that can be solved without hacks
• A very good starting point for understanding methods that work in general graphs (MRFs)
• In fact many methods are only understood as an extension of exact algorithms on trees
• There are actually many applications
Markov Chain

Directed GM

\[ p(x) = p(x_1) \prod_{i=2}^{n} p(x_i | x_{i-1}) \]

Equivalent directed GM

\[ p(x) = p(x_n) \prod_{i=1}^{n-1} p(x_i | x_{i+1}) \]

For converting between these forms, we will need an algorithm for computing marginals.

Undirected GM

Given \( X_3, X_2 \) and \( X_4 \) are independent...

Factorization:

\[ \prod_{ij}^{n} g(x_i, x_j) \]

Factorization in marginals:

\[ p(x) = \prod_{ij}^{n} \frac{p(x_i, x_j)}{p(x_i)p(x_j)} \prod_{i} p(x_i) \]
Hidden Markov Model

Directed GM

Undirected GM

\[ p(x, y) = p(x_1) \prod_{i=2}^{n} p(x_i | x_{i-1}) \prod_{i=1}^{n} p(y_i | x_i) \]

- Sequences (text, grammars)
- Time dependencies (speech, tracking, DNA)
- Good for understanding many things
- Basis for generalization of several algorithms

Observe that: \[ p(x) = p(x_1) \prod_{i=2}^{n} p(x_i | x_{i-1}) - \text{Markov chain} \]
Maximum a posteriori (MAP): given observation $y$ we want to find the most probable hidden configuration $x$: $\max_x p(x \mid y)$

Recall $p(x \mid y) = \frac{p(x, y)}{p(y)}$

For fixed $y$, pdf $p(x \mid y)$ is a Markov chain on $x$:

$$p(x \mid y) = \frac{p(y \mid x)p(x)}{p(y)} = \frac{1}{p(y)} p(x) \prod_i p(y_i \mid x_i) = \frac{1}{p(y)} \prod_{ij} g_{ij}(x_i, x_j) \prod_i g_i(x_i)$$

(We’ll need marginalization computations to recover a directed or marginals factorization)

To find the MAP solution $x$ we don’t need to know $p(y)$:

$$\arg\max_{x} \prod_i g_i(x_i) \prod_{ij} g_{ij}(x_i, x_j)$$

data prior
Energy Minimization

\[ \text{argmax}_x \prod_i g_i(x_i) \prod_{ij} g_{ij}(x_i, x_j) = \text{argmax}_x \log \left( \prod_i g_i(x_i) \prod_{ij} g_{ij}(x_i, x_j) \right) \]

\[ \text{log is monotone, all factors non-negative} \]

\[ f_a(x_a) = - \log g_a(x_a) \]

\[ \text{argmin}_x \left[ E(x) = \sum_i f_i(x_i) + \sum_{ij} f_{ij}(x_i, x_j) \right] \]

- Need to find a minimum of a function which is a sum of functions of one variable (unary terms) and two variables (pairwise terms)
As Shortest Path

\[
\text{argmin}_x \left[ E(x) = \sum_i f_i(x_i) + \sum_{ij} f_{ij}(x_i, x_j) \right]
\]

(Construction known as Trellis graph)

- Paths map one to one to labelings \( x \); cost of a path equals \( E(x) \)
- Shortest path \( \implies \) MAP solution
Problem:

\[
\min_x \sum_{i \in \mathcal{V}} f_i(x_i) + \sum_{ij \in \mathcal{E}} f_{ij}(x_i, x_j)
\]

Use distributivity:

\[
\min (a + c, b + c) = \min (a, b) + c
\]

\[
\min_{x_1, \ldots, x_n} \left[ f_{1,2}(x_1, x_2) + f_1(x_1) + \ldots \right] = \min_{x_2, \ldots, x_n} \left[ \min_{x_1} \left[ (f_{1,2}(x_1, x_2) + f_1(x_1)) + \ldots \right] \right]
\]

Recurrent update:

\[
\varphi_1(x_1) = 0
\]

\[
\varphi_j(x_j) = \min_{x_i} \left( f_{ij}(x_i, x_j) + f_i(x_i) + \varphi_i(x_i) \right)
\]

Viterbi Algorithm:
Forward pass: computes best path from the left
Backward pass: backtrack the minimizer

Shortest path from the left to every state. Core of all message passing algorithms
Given factorization $p(x) = \frac{1}{Z} \prod_i g_i(x_i) \prod_{ij} g_{ij}(x_i, x_j)$

Compute $p(x_i), p(x_i, x_j)$:

\[
p(x_i) = \sum_{x_V \setminus \{i\}} p(x) = \sum_{x_1, \ldots, x_{i-1}, \square, x_{i+1} \ldots x_n} p(x);
\]

\[
p(x_i, x_j) = \sum_{x_V \setminus \{i, j\}} p(x)
\]

\[
p(x_i) \propto \hat{M}_i(x_i) g_i(x_i) \hat{M}_i(x_i)
\]

\[
\sum_{x_i} p(x_i) = 1
\]

\[
p(x_i, x_j) \propto \hat{M}_i(x_i) g_i(x_i) g_{ij}(x_i, x_j) g_j(x_j) \hat{M}_j(x_j)
\]

\[
\sum_{x_i, x_j} p(x_i, x_j) = 1
\]
Marginals

Given factorization $p(x) = \frac{1}{Z} \prod_i g_i(x_i) \prod_{ij} g_{ij}(x_i, x_j)$

Compute $p(x_i), p(x_i, x_j)$:

$$p(x_i) = \sum_{x_V \setminus \{i\}} p(x) = \sum_{x_1, \ldots, x_{i-1}, \square, x_{i+1} \ldots x_n} p(x); \quad p(x_i, x_j) = \sum_{x_V \setminus \{i, j\}} p(x)$$

- **Use distributivity**: $a \cdot c + b \cdot c = (a + b) \cdot c,$

$$\sum_{x_1, \ldots, x_{i-1}} \left[ g_{12}(x_1, x_2) \cdot g_1(x_1) \cdot (\ldots) \right] = \sum_{x_2, \ldots, x_{i-1}} \left[ \sum_{x_1} \left[ (g_{12}(x_1, x_2) \cdot g_1(x_1)) \cdot (\ldots) \right] \right]$$

- **Recurrent update**: 

$$\vec{M}_1(x_1) = 1$$

$$\vec{M}_j(x_j) = \sum_{x_i} (g_{ij}(x_i, x_j) \cdot g_i(x_i) \cdot \vec{M}_i(x_i))$$

Note: this is matrix-vector product
Forward-Backward Algorithm

- **Forward**: compute left marginals recurrently: $\hat{M}_i(x_i)$
- **Backward**: compute right marginals recurrently $\hat{M}_i(x_i)$
- Compose marginals as $p(x_i) = \hat{M}_i(x_i) g_i(x_i) \hat{M}_i(x_i)$
Exercise: Extend to Trees

\[ p(x_i) \propto \tilde{M}_i(x_i) g_i(x_i) \tilde{M}_i(x_i) \]
Generalized Algorithms

- Did you notice the similarity of computations in MAP and marginals problems?

Actually, for any semi-ring \((R, \oplus, \otimes)\) there holds distributivity:

\[
a \otimes (b \oplus c) = (a \otimes b) \oplus (a \otimes c)
\]

\[
(b \oplus c) \otimes a = (b \otimes a) \oplus (c \otimes a)
\]

We can write a generalized algorithm for the problem of \(\oplus\otimes\) marginals on a chain (tree):

\[
m_i(x_i) = \bigoplus_{x \neq x_i} \bigotimes_{ij} g_{ij}(x_i, x_j)
\]

For example: \((\mathbb{B}, \vee, \wedge)\), \(([0, 1], \min, \max)\), \((\mathbb{R}, \logsumexp, +) \sim (\mathbb{R}_+, +, \times)\)

[Schlesinger M.I. Ten lectures in statistical and structural pattern recognition]
Example: Scan-line Stereo

- **Input**
  - Two images from a calibrated camera pair
  - Rectified: epipolar lines correspond to image rows

- **Problem**
  - For each pixel in the left image find the corresponding pixel in the right image

- **Output**
  - Dense depth (disparity) map

![Input Pair](image1.png)  ![Disparity Map (GT)](image2.png)
Example: Scan-line Stereo

\( i \) - pixel
\( x_i \) - chosen disparity label
\( x = (x_i \mid i \in \mathcal{V}) \) - labeling

\( f_i(x_i) \) - matching cost

\( f_{ij}(x_i, x_j) \) - smoothness cost

\[
\min_x \sum_{i \in \mathcal{V}} f_i(x_i) + \sum_{ij \in \mathcal{E}} f_{ij}(x_i, x_j)
\]
Here, the data term shows how our algorithm combines differences. We start by defining our energy function (sec- 
more than one direction. Concretely our strategy injects information from the 2D problem within, but not across horizontal scanlines. The corre-
As a result of this multiple recursion, the belief at a given pixel is influenced by its entire. . .
As a result of this multiple recursion, the belief at a given pixel is influenced by its entire. . .

Figure 2: (a) Star-shaped graphs associated in SGM to two adjacent pixels. (b) Depiction of (Hirschm¨uller, 2005) to derive the disparity of pixel. The algorithm proposed in this paper performs a sep-
subsequent work (Deng and Lin, 2005) to derive the disparity of pixel. Although the optimization is exact over the star-shaped graph, the graphs for two adjacent pixels need to be computed. Although our trees prove to be smaller subset of the whole image's pixels. This repre-

Tree-based Heuristics for Stereo

full graph

scanline DP

Bleyer & Gelautz-VISSAP-08
+ own tree for each pixel
+ larger coverage

Veksler-05
Psota et al. ICCV-15

+ connect similar colors first
+ learned potentials

Hirschmüller-05 (SGM)
+ own tree for each pixel
+ reuse messages in DP
Conclusion

- Hidden Markov Model is very similar to Markov Chain
- All problems seem to be solvable with a kind of dynamic programming (but e.g. unsupervised learning isn’t)
- In fact, trees seem to be important
• Junction Tree Algorithm
• Unsupervised learning (hidden states not observed) — Baum-Welsche algorithm
• Parallel algorithms $O(n \log(K))$ time with $K$ processors:
  • sum-product: Fourier transform
  • min-sum: lower envelopes, distance transform
• Kalman Filter
• Markov Chain Monte Carlo
  • Ergodicity and stationary distribution
• Finale state automata
• Markov Decision Processes
Given $x_i$, the optimal solution consists of optimal solution (s to $x_i$) and ($x_i$ to t)

Variables ($X_1, \ldots, X_{i-1}$) and ($X_{i+1}, \ldots, X_n$) are conditionally independent given $X_i$
One minimization of the form

$$\varphi_j(x_j) = \min_{x_i} (\varphi_i(x_i) + f_i(x_i) + f_{ij}(x_i, x_j))$$

is the problem of finding a lower envelope of a set of functions well studied in geometry / graphics.

- Lower envelope (distance transform)
  
  $$f_{ij}(x_i, x_j) = w_{ij} \rho(x_i - x_j)$$

  $O(nL^2)$ - naive approach, $n$ variables, $L$ labels

  $O(nL)$ - efficient sequential algorithms

  $O(n \log L)$ - efficient parallel algorithms, using $L$ processors

[Hirata’96, Meijster’02] [Felzenszwalb & H.’06] [Goodrich’86, Chen’02]
Max-Product BP, Tree-Reweighted

- Can Run Message passing in parallel
  \[d(i, j) := \min_k (d(i, k) + d(k, j))\]
  c.f. all shortest paths in a graph
  \(O(n)\) time, \(O(n)\) processors

- Can apply on graphs with loops (loopy BP)
  - Over-counting
  - May oscillate
  - May diverge (unbounded)

- Tree-Reweighted [Wainwright'05]
  - Decomposition into trees
  - Connection to LP relaxation and its dual
  - Parallel algorithm may still oscillate
Markov Random Fields
Goals

- Definitions
- Examples in Computer Vision
- Overview on MAP problem, one technique in detail
- Marginals problem — variational approach in detail
• Collection of discrete random variables

\[ X_1, X_2, \ldots, X_n, \quad X_i \in D \]

Definition

\[ p : D^n \rightarrow \mathbb{R} \text{ is a random field if } p(x) > 0 \ \forall x, \ \sum_x p(x) = 1. \]

• Non-negativity is important for existence of conditional probabilities and other good reasons. Practically not a limitation.

Definition

Random field \( p \) is a Markov random field if it satisfies some conditional independence (Markov) properties.

(Book: Lauritzen S.L., "Graphical Models", 1996)
MRF w.r.t. a Graph

- Graph $G = (V, E)$
- Set of nodes $V$; random variables $X_i, i \in V$
- Set of edges $E$

- **Local Markov Property w.r.t. $G$:**
  - Given the neighbors of $X_i$, it is independent of the rest:
  $$p(X_i \mid X_{V_i}) = p(X_i \mid X_{N(i)}), \forall i \in V$$

- **Pairwise Markov Property w.r.t. $G$:**
  - Absent edge $(i, j)$ in $G$ iff $X_i$ and $X_j$ are conditionally independent given the rest of variables.

**Theorem (Lauritzen 96)**

*Local and Pairwise Markov Properties are equivalent.*

**Definition**

MRF w.r.t. graph $G$ is a random field satisfying Markov property w.r.t. $G$
Definition

$p$ is a **Gibbs Random field** if it factors as $p(x) = \prod_{c \subseteq S} f_c(x_c)$,

- Here we do not need $c$ to be a clique in some graph
- Knowing factorization is more than knowing conditional independencies
- The factorization is what matters for the representation tractability and inference

MRF factorization

- Conditional independencies help to structure and simplify the distribution

Theorem (Hammersley-Clifford, 1971)

MRF $p$ w.r.t. graph $G$ factors over cliques of $G$: $p(x) = \prod_{c \subseteq C} f_c(x_c)$,

- $C$ is the set of cliques – maximal fully connected subgraphs
Two-class Segmentation

MRF Model

Observations: \( p(y \mid x) = \prod_i p(y_i \mid x_i) \)

Prior: \( p(x) = \prod_{ij} \exp(-\lambda|x_i - x_j|) \) same neighbors are more probable

Samples from the prior for varied lambda:

Input Image

binary segmentation

\( p(X) \)
Conditional Random Field

- $x_i$, $i \in V$ - hidden random variables (segmentation)
- $y_j$, $j \in V'$ - observed random variables (Image)

Definition (Lafferty et al. 01)

$p(x \mid y)$ is a conditional random field if it satisfies Markov properties w.r.t. $x$ given $y$.

Generative: $p(y) = \sum_x p(x, y)$

Recognition is the same: $\arg\min_x p(x, y) = \arg\min_x p(x \mid y)$

MRF $p(x, y)$

hidden variables $x$

observed variables $y$

CRF $p(x \mid y)$

more flexible for recognition
Two-class Segmentation

CRF Model

\[ p(x|y) \text{ is an MRF} \]

CRF model: \[ p(y | x) = \prod_i g_i(y | x_i) \]

\[ g_i(y|x_i) - \text{could be a logistic model, decision tree, boosted classifier, etc.} \]
MAP of MRF — Energy Minimization
MAP of MRF

- Given the model \( p(x) = \prod_{c \in S} g_c(x_c) \) find the most probable state:

  \[
  \max_x p(x)
  \]

- Joint maximization in all variables

- Take negative logarithm:

  \[
  \min_x \sum_{c \in S} -\log g_c(x_c) = \min_x E(x)
  \]

- Partially separable minimization problem, called Energy minimization

- Belongs to discrete optimization domain (combinatorial optimization, graph theory, ILP, relaxations, etc.)

- Many optimization techniques specifically suitable for computer vision
Common scenario: only pairwise interactions:

$$\min_x \sum_{i \in \mathcal{V}} f_i(x_i) + \sum_{ij \in \mathcal{E}} f_{ij}(x_i, x_j)$$

$(\mathcal{V}, \mathcal{E})$ - graph

$\mathcal{V}$ - set of nodes

$\mathcal{E}$ - set of edges

$x = (x_i \mid i \in \mathcal{V})$ - labeling

- NP-hard (includes MAX-CUT, vertex packing, etc.)
- Two large groups of methods used in CV:
  - minimum cut (graph cuts)
  - LP relaxation / message passing
- There are much more
Common scenario: only pairwise interactions:

$$\min_x \sum_{i \in \mathcal{V}} f_i(x_i) + \sum_{ij \in \mathcal{E}} f_{ij}(x_i, x_j)$$

$(\mathcal{V}, \mathcal{E})$ - graph
$\mathcal{V}$ - set of nodes
$\mathcal{E}$ - set of edges
$x = (x_i \mid i \in \mathcal{V})$ - labeling

- NP-hard (includes MAX-CUT, vertex packing, etc.)
- Two large groups of methods used in CV:
  - minimum cut (graph cuts)
  - LP relaxation / message passing
- There are much more
Example: Potts Model for Object Class Segmentation

- \( \mathcal{V} \) - set of pixels; \( \mathcal{E} \subset \mathcal{V} \times \mathcal{V} \) neighboring pixels;
- \( \mathcal{X}_s = \{1, \ldots K\} \) – class label;
- \( E_f(x) = \sum_{s \in \mathcal{V}} f_s(x_s) + \sum_{st \in \mathcal{E}} \lambda_{st}[x_s \neq x_t] \).

Image Ground Truth

(MSRC object class segmentation)
Cannot guarantee \( f(x) \leq P(n)f(x^*) \)

\[ f(x) \leq Cf(x^*) \]

\[ f(x) = f(x^*) \]

As Integer Linear Program

- Energy minimization: \( \min_x \sum_i f_i(x_i) + \sum_{ij} f_{ij}(x_i, x_j) \)
- For each \( i \) encode \( x_i \) with \( \mu_i(k) \in \{0, 1\}, k \text{ -- label} \)
- For each \( ij \) encode \((x_i, x_j)\) with \( \mu_{ij}(k, k') \in \{0, 1\} \)
- The objective linearizes
- \( \mu \) need to respect constraints

\[
\min_{\mu} \sum_i \sum_k f_i(k)\mu_i(k) + \sum_{ij} \sum_{k,k'} f_{ij}(k, k')\mu_{i,j}(k, k')
\]

\( \mu \geq 0; \quad \mu \in \{0, 1\}^I \quad \rightarrow \mu \in [0, 1] - \text{LP relaxation} \)

\[
\sum_k \mu_i(k) = 1
\]

\[
\sum_{k,k'} \mu_{ij}(k, k') = 1
\]

\[
\sum_{k'} \mu_{ij}(k, k') = \mu_i(k)
\]

\[
\sum_k \mu_{ij}(k, k') = \mu_j(k')
\]

\[
\mu_{ij}(k, k')
\]

\[
\mu_i(k)
\]

\[
\mu_j(k')
\]
The Power of Basic LP Relaxation

- Consider a class $C$ of problems specified by unrestricted graph structure and pairwise potentials from some set $F$.

**Theorem (Thapper and Zivny 2012, Kolmogorov 2013)**

*(Roughly)* Class $C$ has a polynomial time algorithm iff the Basic LP relaxation is tight for $C$.

- This means LP relaxation is a rather universal tool
- It is also tight for many practical individual instances or provides a good approximation

**Theorem (Prusa, Werner, 2017)**

LP Relaxation of MAP MRF is as hard as any linear program. (Already for Potts model with 3 labels on a planar graph).

- It means it is very unlikely to come up with an algorithm better than $O(n^{3.5}L)$
- Many approximate methods developed in Computer Vision
Minimum s-t Cut

Capacitated network
\[ G = (V, E, c), \]
\[ c(u, v) \geq 0 \text{ - arc capacities} \]

Cut cost:
\[ \sum_{(u,v) \in E \atop u \in S \atop v \notin S} c(u, v) \rightarrow \min_{S \atop s \in S \atop t \notin S} \]

Source set \( S \)

Cut \((S, V \setminus S)\)

Sink set \( T = V \setminus S \)

- Problem history: 30+ years
- Active research for better algorithms:
  - theoretical (Orlin’12: \( O(mn) \) algorithm), parallel algorithms
  - practical, esp. in computer vision
Reduction to Minimum s-t Cut

- Let \( x_i \in \{0, 1\} \)
- Energy minimization: \( \min_x \sum_{i \in V} f_i(x_i) + \sum_{ij \in E} f_{ij}(x_i, x_j) \)
- Expand as polynomial:
  \[
  f_i(x_i) = f_i(1)x_i + f_i(0)(1 - x_i) = c_0 + c_i x_i; \\
  f_{ij}(x_i, x_j) = \ldots = c_0' + c_i' x_i + c_j' x_j + c_{ij} x_i(1 - x_j).
  \]
- Minimum cut: \( \min_{S \subset V} \sum_{ij \in (S, V \setminus S)} c_{ij} \)
- Solvable in polynomial time if \( c_{uv} \geq 0 \)
Segmentation as Mincut
Recall the segmentation model: $f_{ij}(x_i, x_j) = \lambda|x_i - x_j|, \; x_i x_j \in \{0, 1\}$

Derive $c_{ij}$ such that $f_{ij}$ expresses as

$c_0 + ax_i + bx_j + c_{ij}x_i(1 - x_j)$
Applications of min-cut

Multiview Reconstruction
Lempitsky et al. 2006
Boykov and Lempitsky 2006

Surface Fitting
Lempitsky and Boykov 2007

3D Segmentation
Boykov and Joly 2001
Boykov and Funka-Lea 2006
Boykov and Kolmogorov 2003

(More with further extensions)
Just few more…
Example: Joint Segmentation and Parameter Estimation

- **Input:**
  - Image
  - FG / BG brush

- **Output:**
  - Complete segmentation

Rother, Kolmogorov, Blake: “GrabCut” — Interactive Foreground Extraction using Iterated Graph Cuts
Model

- Markov random field (generative) model:
- Segmentation $x : \Omega \rightarrow \{0, 1\}$
  - Model: $p(x)$ - neighboring pixels are more likely to take the same segment
- Color clusters: $k : \Omega \rightarrow \{1, \ldots, K\}$
  - Model: $p(k|x)$ - conditionally independent for all pixels
- Image: $I : \Omega \rightarrow \mathbb{R}^3$ - color drawn from a color cluster
  - Model: $p(I|k)$ - conditionally independent for all pixels
Method

- Given appearance model find best segmentation (min-cut)
- Given segmentation refit the appearance model

- Problem: fitting a Gaussian mixture is not closed form, may oscillate or get stuck
- Solution: Expectation Maximization algorithm
Stereo as Mincut

Sequence Alignment problem (bioinformatics), Needleman–Wunsch algorithm (1970)
Also good for scan-line stereo!

Shortest Path

Minimum Cut — extends to surfaces

Hard to construct directly (one CV paper did)
Many other Problems Solvable with Min-cut

$f_{i,j}(x_i, x_j) = V(x_i - x_j)$, convex

More generally, submodular

... Class of graph-cut representable problems

Multi-class segmentation for a hierarchy of nested candidate regions

Optimized Crossover

Current best solution

Proposal solution

Crossover (fusion problem)

Local Search in some combinatorial locality
Expansion Move

[Boykov, Veksler, and Zabih: "Fast Approximate Energy Minimization via Graph Cuts", 1999]

Current best solution $x$

Proposal solution $y$

Crossover (fusion problem)

Minimum Cut
Space of Possible Expansion of One Label
Expansion Move

- Start with initial solution $x$
- For each label $a$
  - Consider the Expansion-Move to $a$:
  - $x_i$ stays or switches to $a$ -> reduce to graph cut and solve
- Iterate until $x$ stops changing

Semi-metric $f_{ij}(\alpha, \beta)$:

- $f_{ij}(\alpha, \beta) = 0$ iff $\alpha = \beta$
- $f_{ij}(\alpha, \beta) = f_{ij}(\beta, \alpha) \geq 0$
- $f_{ij}(\alpha, \beta) \leq f_{ij}(\alpha, \gamma) + f_{ij}(\gamma, \beta)$

Theorem (Boykov, Veksler, Zabich, 1999)

For semi-metric problems, the expansion-move algorithm finds a solution with an approximation ratio:

$$2c = 2c \max_{ij} \frac{\max_{\alpha \neq \beta} f_{ij}(\alpha, \beta)}{\min_{\alpha \neq \beta} f_{ij}(\alpha, \beta)}$$

Truncated Quadratic

"robust" potentials: outliers not over penalized
Applications of graph cuts

Stereo
Boykov et al. 1998
Kolmogorov and Zabih 2001

A general and fast technique
In 2011 received
Helmholtz Prize (Test of Time) Award
MRF Marginals — Mean Field Approximation
Computing Marginals

\[ p(x \mid y) \propto \exp \left( \sum_i -\phi_i(x_i, y_i) - \sum_{(i,j)} \phi_{ij}(x_i, x_j) \right) \]

\[ \phi_{ji}(x_j, x_i) \equiv \phi_{ij}(x_i, x_j) \]

Posterior of the states given image

Want to estimate marginals \( p(x_i \mid y) \)

\[ p(x_i \mid y) = \mathbb{E}_{x_N \setminus \{i\}} [p(x \mid y)] \propto \sum_{x_N \setminus \{i\}} \exp \left( \sum_i -\phi_i(x_i, y_i) - \sum_{(i,j)} \phi_{ij}(x_i, x_j) \right) \]
Example of Marginal Probabilities
Factorized Approximation of the Posterior

\[ p(x_i \mid y) = \propto \sum_{x_{\mathcal{V}\setminus\{i\}}} \exp \left( \sum_i -\phi_i(x_i, y_i) - \sum_{(i,j)} \phi_{ij}(x_i, x_j) \right) \]

Posterior of the states given image
Want to estimate marginals \( p(X_i \mid I) \)

\[ q(x) = \prod_i q_i(x_i) \]

Approximation of the posterior
(assume posterior distribution is concentrated around one configuration)
Let \( p(X) \) and \( q(X) \) be two probability distributions.

### Definition

Kullback–Leibler divergence (1951) of \( p \) and \( q \) is:

\[
KL(p(X) \| q(X)) = \sum_{x \in X} p(x) \log \frac{p(x)}{q(x)}
\]

In the definition above, \( 0 \log \frac{0}{0} = 0 \log \frac{0}{q} = 0 \) and \( p \log p0 = \infty \).

For continuous variables:

\[
KL(p(X) \| q(X)) = \int p(x) \log \frac{p(x)}{q(x)} \, dx
\]

The expected number of extra bits required to code samples from \( p \) using a code optimized for \( q \)

The amount of information lost when \( q \) is used to approximate \( p \)

Non-negative, \( KL(p \| q) = 0 \) iff \( p = q \)
Assume $p(x) > 0$, $q(x) > 0$, $\sum_x p(x) = 1$, $\sum_x q(x) = 1$

Statement: $\sum_x p(x) \log \frac{p(x)}{q(x)} \geq 0$

Proof

Denote $y(x) = \frac{q(x)}{p(x)}$, the inequality reads:

$\sum_x p(x)(-\log y(x)) \geq 0$

Observe that $\log$ is a convex function, apply Jensen’s inequality:

$\sum_x p(x)(-\log y(x)) \geq -\log \sum_x p(x)y(x) = -\log 1 = 0$

From strictly convexity: equality iff all $y(x)$ are equal
Asymmetry

Minimizing forward KL divergence:

$$\min_q \text{KL}(p\|q) \left( \int p(x) \log \frac{p(x)}{q(x)} dx \right)$$

Well on average in the expectation over $p$

Minimizing reverse KL divergence:

$$\min_q \text{KL}(q\|p) \left( \int q(x) \log \frac{q(x)}{p(x)} dx \right)$$

Well on average in the expectation over $q$ — concentrating around a mode of $p$

Example:

$p$ - bimodal
$q$ - Gaussian

• This gives rise to two families of variational methods
Reverse KL — Mean Field

\[
KL(q\|p) = \sum_x q(x) \log \frac{q(x)}{p(x)} = -\sum_x q(x) \log p(x) + \sum_x q(x) \log q(x)
\]

Cross-entropy / Evidence - Entropy

Entropy of independent variables is additive:

\[
\sum_x q(x) \log q(x) = \sum_x \prod_{i'} q_i'(x_i') \sum_i \log q_i(x_i) = \sum_x \sum_i \prod_{i'} q_i'(x_i') \log q_i(x_i)
\]

\[
= \sum_i \sum_i \prod_{i'} q_i'(x_i') \log q_i(x_i) = \sum_i \sum_i q_i(x_i) \log q_i(x_i) = \sum_i -H(q_i).
\]

Cross-entropy decouples over pairwise terms:

\[
\sum_x q(x) \log p(x) = -\sum_x \prod_{i'} q_i'(x_i')(\sum_i \phi_i(x_i) + \sum_{ij} \phi_{ij}(x_i, x_j))
\]

\[
= -\sum_i \sum_i \phi_i(x_i)q_i(x_i) - \sum_{ij} \sum_{x_i, x_j} \phi_{ij}(x_i, x_j)q_i(x_i)q_j(x_j)
\]
Mean Field

\[ \min_q \sum_i \sum_{x_i} q_i(x_i) \left( \log q_i(x_i) + \phi_i(x_i) + \sum_{j \in \mathcal{N}(i)} \sum_{x_j} q_j(x_j) \phi_{ij}(x_i, x_j) \right) \]

s.t. \( q_i \geq 0; \quad \sum_{x_i} q_i(x_i) = 1 \quad \forall i \quad | \quad \text{Lagrange multiplier } \lambda_i \)

\[ 0 = \frac{\partial}{\partial q_i(x_i)} = \log q_i(x_i) + \phi_i(x_i) + 1 + \sum_{j \in \mathcal{N}(i)} \sum_{x_j} q_j(x_j) \phi_{ij}(x_i, x_j) - \lambda_i \]

\[ \log q_i(x_i) = -\phi_i(x_i) - \sum_{j \in \mathcal{N}(i)} \sum_{x_j} q_j(x_j) \phi_{ij}(x_i, x_j) - \lambda'_i \]

\[ q_i(x_i) \propto \exp(-\phi_i(x_i)) \prod_{j \in \mathcal{N}(i)} \exp \left( - \sum_{x_j} q_j(x_j) \phi_{ij}(x_i, x_j) \right) \]

Algorithms:
- sequential coordinate-wise minimization (convergent)
- parallel coordinate-wise (may oscillate)

Non-convex because of \( q_i q_j \)
Fully Connected (Dense) CRFs

Assume potentials have the following structure: \( \phi_{ij}(x_i, x_j) = \rho(x_i, x_j)k(i - j) \)

\[
\log q_i(x_i) = \phi_i(x_i) + \sum_{j \neq i} \sum_{x_j} q_j(x_j)\rho(x_i, x_j)k(i - j) - \lambda'
\]

Parallel update can be implemented efficiently:

- For all labels \( l \):
  - \( s(j) := \sum_{l'} q_j(l')\rho(l, l') \)
  - \( \log q_i'(l) := \phi_i(l) + \sum_{j \neq i} s(j)k(i - j) = \phi_i(l) + s \ast k - s(i)k(0) \)
- Renormalize all \( q_i' \)

[Kraehenbuehl and Koltun: Efficient Inference in Fully Connected CRFs with Gaussian Edge Potentials, 2012]

Potentials of the form: \( \phi_{ij}(x_i, x_j) = \rho(x_i, x_j) \sum_m w_m k^m(f_i - f_j), \)

\( f \) -some features \( \rightarrow \) bilateral filtering

Convergence with some assumptions, better algorithms than parallel coordinate-descent, other relaxations
[Krahenbuehl and Koltun: Efficient Inference in Fully Connected CRFs with Gaussian Edge Potentials, 2012]
Forward KL

\[ KL(p\|q) = \sum_x p(x) \log \frac{p(x)}{q(x)} = \sum_x p(x) \log p(x) - \sum_x p(x) \log q(x) \]

- Entropy of p

Cross-entropy

\[-\mathbb{E}_{p(X)} \log q(X)\]

When minimizing in \(q\), \(H(p)\) does not matter

Cross-entropy simplifies using factorization of \(q\):

\[ \sum_x p(x) \log q(x) = \sum_x p(x) \sum_i \log q_i(x_i) = \sum_i \sum_{x_1, \ldots, x_i, \ldots, x_n} p(x) \log q_i(x_i) = \sum_i \sum_{x_i} p(x_i) \log q_i(x_i) \]

Turns out that we need to know marginals \(p(X_i)\). But then:

\[ \min_q - \sum_i \sum_{x_i} p(x_i)q_i(x_i) \]

\[ \Rightarrow q_i(x_i) = p(x_i) \]

s.t. \( \sum_i q_i = 1 \)

Forward divergence was the “right one” but we did not get a simplification
Mean Field as Approximation and Forward KL

\[ q'_i(x_i) \propto \exp(-\phi_i(x_i)) \prod_{j \in \mathcal{N}(i)} \exp\left( - \sum_{x_j} q_j(x_j) \phi_{ij}(x_i, x_j) \right) \]

Terms from the original distribution \( \text{terms from current estimate} \)

The iterative algorithm can be understood as follows. At each iteration

- Approximate \( p(x) \approx \hat{p}(x) = p(x_i \mid x_{V \setminus \{i\}})q(x_{V \setminus \{i\}}) \)

- Minimize \( KL(\hat{p} \parallel q) \)

Note, the second step efficiently means \( q_i := \hat{p}(x_i) = \sum_{x_{\mathcal{N}(i)}} p(x_i \mid x_{\mathcal{N}(i)})q(x_{\mathcal{N}(i)}) \)
Graphical Models as Neural Networks

Semantic Segmentation

$X_1 \in \{\text{bg, cat, dog, person}\}$

Pixels/locations

$X_1 = \text{bg} \quad X_4 = \text{cat}$

Classifier for each pixel

$X_2 = \text{dog}$

Enforce consistence with CRF
Gradual “Neuralization” of CRF approaches
Mean Field CRF inference as common CNN operations

\[
Q_u(l) \leftarrow \frac{1}{\sum_{l'} \exp(U_u(l'))} \exp(U_u(l)) \quad \text{Initialization}
\]

\[Q_u^{(m)}(l) \leftarrow \sum_{v \neq u} k^{(m)}(f_u, f_v) Q_v(l) \quad \text{for all } m \quad \text{Message Passing}
\]

\[\tilde{Q}_u(l) \leftarrow \sum_m w^{(m)} \tilde{Q}_u^{(m)}(l) \quad \text{Weighting Filter Outputs}
\]

\[\tilde{Q}_u(l) \leftarrow \sum_{l' \in L} \mu(l, l') \tilde{Q}_u(l') \quad \text{Compatibility Transform}
\]

\[Q_u(l) \leftarrow U_u(l) - \tilde{Q}_u(l) \quad \text{Adding Unary Potentials}
\]

\[Q_u(l) \leftarrow \frac{1}{\sum_{l'} \exp(Q_u(l'))} \exp(\tilde{Q}_u(l)) \quad \text{Normalizing}
\]

end while

Conditional random fields as recurrent neural networks (Zheng et al., 2015)
Another Example: CRF with Learned Potential Structure

Improved results compared to DenseCRF, based on Gibbs sampling (training and test time)
Knöbelreiter et al. End-to-End Training of Hybrid CNN+CRF Models for Stereo, 2017

Adaptive regularizer: \( w_{ij} \rho(\|x_i - x_j\|) \)
Commonly applied in segmentation, stereo
Contrast sensitive (gradient) model: \( w_{ij} = \exp(\alpha |I_i - I_j|^\beta) \)

Contrast Sensitive / Pairwise CNN

I_0 \rightarrow Unary CNN \rightarrow Correlation \rightarrow CRF \rightarrow D

I_1 \rightarrow Unary CNN

Generalizes engineered features
Occurs in many matching problems:
image retrieval, optical flow, stereo

Replaces post-processing such as
Cost aggregation / filtering, SGM, etc.
## Effect of Joint Training

<table>
<thead>
<tr>
<th>Layer</th>
<th>Unary CNN</th>
<th>Unary CNN + CRF</th>
<th>Full Joint</th>
</tr>
</thead>
<tbody>
<tr>
<td>3 layer</td>
<td>24.67</td>
<td>4.25</td>
<td>3.84</td>
</tr>
<tr>
<td>7 layer</td>
<td>17.83</td>
<td>3.11</td>
<td>2.69</td>
</tr>
</tbody>
</table>

5 iterations of DMM
• CRF could improve the results
• But also, we practically implemented it with CNN-like elements
• It means that in fact we have designed specialized CNN layers with a special structure
  • allowing for more spatial interactions
  • enforcing clustering of neighboring predictions
  • adjusting to image edges
• Does it matter that these layers were derived from MAP CRF?

• Further Topics
  • Deep Boltzmann machine, Deep Bayesian network
Bayesian Networks
Directed Graphical Model (Bayesian Network)

- Directed Acyclic Graph
  - Graph $G = (V, E)$
  - Set of nodes $V$; random variables $X_i$, $i \in V$
  - Set of directed edges $E \subset V \times V$
  - There are no directed loops in $G$
  - Parents of $i$ is the set $\text{Pa}(i) = \{j \in V \mid (j, i) \in E\}$

Edges encode “direct dependencies”

Definition

*Bayesian network* w.r.t. graph $G$ is a random field that factorizes as

$$p(X) = \prod_{i \in V} p(X_i \mid X_{\text{Pa}(i)})$$
Logistic conditional probabilities:
- the probability model that has linear discriminant function
- can be also derived assuming the factorization

Same conditional probabilities in:
- restricted Boltzmann machine, deep Boltzmann machine, deep Bayesian network

As considered by Neal (1992)
- Binary variables
- Conditional probabilities using logistic model:

\[ p(Y_j = 1 | X) = \frac{1}{1 + \exp(-\sum_i w_i X_i)} \]

\[ p(Y | X) = \prod_j p(Y_j | X) \]

Logistic conditional probabilities:
- the probability model that has linear discriminant function
- can be also derived assuming the factorization
- Same conditional probabilities in:
  - restricted Boltzmann machine, deep Boltzmann machine, deep Bayesian network
The optimal Bayesian classifier is given by

\[
p(K = 1 | x) \leq \frac{p(K = 0 | x)}{\theta}
\]

Equivalently, with log-odds:

\[
f(x) := \log p(K = 1 | x) - \log p(K = 0 | x) \leq \eta
\]

What is the form of conditional distribution \( p(K | X) \) such that \( f(x) \) is linear: \( f(x) = w^T x \)?

- \( X \) - observed feature vector
- \( K \) in \( \{0,1\} \) - hidden class label (face / not face)
Consider a joint model $p(X, Y) = p(Y \mid X)p(X)$.

Conditional distribution $p(Y \mid X)$ is *strongly conditionally independent* if it factors as:

$$p(y \mid x) = \frac{1}{Z(x)} \prod_{i,j} g_{ij}(x_i, y_j)$$

$$p(y \mid x) = \frac{1}{Z(x)} \exp \sum_{i,j} u_{ij}(x_i, y_j) = \prod_j \frac{1}{Z_j(x)} \exp \sum_i u_{ij}(x_i, y_j) = \prod_j p(y_j \mid x)$$

Any function $u_{ij}(x_i, y_j)$ of binary variables can be written as $u_{ij}(x_i, y_j) = y_j W_{ij} x_j + b_j y_j + c_i x_i + d$

Terms $c_i x_i + d$ cancel in the normalization of $p(Y \mid X)$

$$p(Y_j = 1 \mid x) = \frac{1}{Z_j(x)} \exp(\sum_i W_{ij} x_j + b_j), \quad p(Y_j = 0 \mid x) = \frac{1}{Z_j(x)} \exp(0) = \frac{1}{Z_j(x)}$$

$$p(Y_j = 1 \mid x) = \frac{1}{1 + \exp\left\{-\left(\sum_i W_{ij} x_j + b_j\right)\right\}}$$
Further Topics

- Global conditional independencies — Markov Blanket
- Local conditional independencies — Moral Graph
- Optimal approximations by trees — Chow-Liu trees

- Other names for BN:
  - belief network,
  - directed graphical model
  - (probabilistic network, causal network, knowledge map)
Neural Networks as Graphical Models

Materials: Shekhovtsov, Flach, Busta: “Feed-forward Uncertainty Propagation in Belief and Neural Networks”, 2018
Recall: Sigmoid Belief Network

\[ p(Y_j=1 | X) = \frac{1}{1 + \exp(-\sum_i w_i X_i)} \]

\[ p(Y | X) = \prod_j p(Y_j | X) \]

Assume input \( X^0 = x^0 \) is given,

Model: \( p(X^n, X^{n-1}, \ldots, X^1 | x^0) \)

First level posterior: \( p(X^1 = 1 | x^0) = S(W^1 x^0) \)

Second level posterior: \( p(X^2 = 1 | x^0) = \sum_{x_1} p(X^2 = 1 | x^1)p(x^1 | x^0) \)

\[ \ldots \]

Network output: \( p(X^n | x^0) = \mathbb{E}_{X_1, X_2, \ldots, X_{n-1}} p(X^n, X^{n-1}, \ldots, X^1 | x^0) \)
• Sigmoid output is often interpreted as probability (e.g. part detectors, hierarchy of logistic models)
• NNs do not compute the expectation (substitute it inside)

Use cases for computing the expectation:
• Improve stability (robustness) of neural networks
• Training networks with binary activations / weights
For two consecutive layers $X, Y$

Apply the first order Taylor approximation for the moments of functions of random variables:

$$p(Y = 1 \mid x^0) = \mathbb{E}_{X \sim P(X \mid x^0)}[S(w^T X)] \approx S(\mathbb{E}_X[w^T X]) = S(w^T \mathbb{E}_X[X])$$

Note that for Bernoulli variables $E_Y[Y] = p(Y = 1 \mid x^0)$.

We obtained standard NN propagation rules where activations are the "means".

- Is this difference important?
Example: Logic Gates

• For example, composition of parts:
  \( X_1 = 1 \) if seeing “car mirror”
  \( X_2 = 1 \) if seeing “car stop light”

• If \( X_1 = 1 \) with probability 0.3 and \( X_2 = 1 \) with probability 0.2 what is the probability that both are present: \( X_1 \& X_2 \)?

Let us fit logistic model
\[
p(Y=1 \mid X) = S(a(X_1 + X_2) + b)
\]
And compare \( \mathbb{E}_X S(a(X_1 + X_2) + b) \)
with \( \text{AP1} = S(a\mathbb{E}_X[X_1 + X_2] + b) \)

<table>
<thead>
<tr>
<th>( p(X_1=1) )</th>
<th>( p(X_2=1) )</th>
<th>( \mathbb{E}[X_1 &amp; X_2] )</th>
<th>( \mathbb{E}[Y] )</th>
<th>( \text{AP1} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0.00015</td>
<td>0.00015</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0.05</td>
<td>0.05</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0.95</td>
<td>0.95</td>
</tr>
<tr>
<td>0.25</td>
<td>0.25</td>
<td>0.0625</td>
<td>0.077</td>
<td>0.0027</td>
</tr>
<tr>
<td>0.5</td>
<td>0.5</td>
<td>0.25</td>
<td>0.26</td>
<td>0.05</td>
</tr>
<tr>
<td>0.75</td>
<td>0.75</td>
<td>0.56</td>
<td>0.55</td>
<td>0.5</td>
</tr>
</tbody>
</table>

Parameters \( a, b \) are set such that: \( S(a(1+1) + b) > 0.95, S(a(0+0) + b) < 0.05 \)

Logistic model is ok, but NN severely underestimates the probability of \( X_1 \text{ AND } X_2 \).
Similarly, for \( X_1 \text{ OR } X_2 \), NN overestimates the probabilities.
Could it Be One of the Reasons for Instability?


CNNs are sensitive to random noise and to adversarial attacks (structured noise optimized to compromise a given network)

The other reasons could be:
Lack of regularization (overfitting)? CNN structure?
Uncertain / Missing Inputs

- Uncertain input may be:
  - Sensor noise (noisy image, lidar, computational sensors, etc.)
  - an input from other networks

A circle denotes an uncertain value

What is the average output when input is random with the given uncertainty?
• Known\(^1\) to improve generalization of NNs
• Usually sampled at training time and replaced with means at test time

\[ Z_i \sim \text{Bernoulli}(0.3) \]

Equivalence of Injected Noise and Probabilistic Models

More generally, let $Y = f(X, Z)$

Then c.d.f. of $Y$ given $X$, $F_Y(y | X) = \mathbb{E}_Z[f(X, Z) \leq y]$

We can in principle reconstruct $p(Y | X)$

In dropout training objective we have something like:

$$\mathbb{E}_Z \left[ \log \text{softmax}(W^n \text{ReLU}(W^{n-1} \text{ReLU}(\ldots W^1 x^0 \ldots)Z_{n-2})Z_{n-1})) \right]$$
NN as Bayesian / Belief Network

- All neurons are random variables
- Feed-forward network = directed graphical model

\[
X^0 = \text{constant}
\]

\[
X^k = WX^{k-1}
\]

\[
X^k = f(X^{k-1} + Z), \ Z \sim \mathcal{N}(0,1)
\]

Infer: \( p(X^k_i | X^0) \)
Goal: if we take into account all stochastic components, we should be able:

- in classification: compute better likelihoods (confidence estimates)
- in regression: output with uncertainty

Sampling techniques [Some paper]

Several methods exist, but not widely used and many open research questions
Feed-forward Uncertainty Propagation

General diagram for all layers

\[ (\mu_i, \sigma_i^2) \quad \text{Layer} \quad (\mu_j', \sigma_j'^2) \]

Linear: \( Y = w^T X \)

\[
\mu' = \mathbb{E}[Y] = w^T \mathbb{E}[X] = w^T \mu, \\
\sigma'^2 = \sum_{ij} w_i w_j \text{Cov}[X] \approx \sum_i w_i^2 \sigma_i^2,
\]

ReLU: \( Y = \max(X, 0) \)

Assume \( X \sim \mathcal{N}(\mu, \sigma^2) \)

\[
\mu' = \int_{-\infty}^{\infty} p(X) f(X) \, dx \\
\sigma' = \int_{-\infty}^{\infty} p(X) f(X)^2 \, dx - \mu'^2
\]

- Also supporting: sigmoid, softmax, max-pooling, maxOut, dropout, …
• Expectations are always smooth
Propagation Methods: Example

- AP1: take clean image and propagate with standard rules
- MC: take several samples of noise and collect statistics from propagating image+noise
- AP2: propagating mean and variance

Under noisy input, estimates may differ
If the output variance is low - standard method is Ok,
But if it is high we would not know about it
Experiments on CIFAR-10

Data: CIFAR10

Network: 9 convolutional layers + last layer: average pooling, softmax
Currently only for shallow networks, working on improving it

**Better Stability**

- **Gaussian Noise**
- **Adversarial (gradient sign)**
Problem:
compute expectations of neurons (mean and variance) over the dataset

Used for: (same as in Batch Normalization)
• initialization (start in a non-saturated regime)
• normalization (a reparametrization better conditioning gradient descent)

Statistics over the Dataset

Poor initialization: all inputs to a neuron are in a saturated part
• Shekhovtsov and Flach: Neural Network Normalization using Analytic Variance Propagation

Statistics for Normalization

Our method

Batch Normalization

Variance Propagation

Standard propagation

dataset statistics

normalized neurons statistics

Train Loss (lr=0.001)
Analytic Dropout

- Can give a better generalization than standard dropout and trains faster
- Related work: Wang and Manning “Fast dropout training”

![Validation Accuracy Graph](image)

- analytic dropout + normalization
- analytic dropout
- dropout(0.2)
- standard

(BN performs better in this plot)
• Lots of things to improve in NNs understanding them as probabilistic models
  • uncertain inputs, stability of NNs under perturbations
  • uncertain outputs for regression
  • initialization and normalization
  • improving training with dropout and other noisy regularizers
  • generative models
  • better learning models
Variational Bayesian Learning
Let $x$ be an input and $y$ the prediction or class label we want to recognize. Consider a conditional model $p(y \mid x; \theta)$ parametrized by $\theta$.

Let $D = \{(x^t, y^t) \mid t = 1, \ldots T\}$ be a set of training samples.

Recall the maximum likelihood approach:

- **Training**: find the maximum conditional likelihood estimate of $\theta$:
  \[
  \hat{\theta} = \arg\max_{\theta} \prod_{t} p(y^t \mid x^t; \theta)
  \]

- **Testing**: recognize new input $x$ using $\hat{\theta}$:
  \[
  y = \arg\max_{y} p(y \mid x; \hat{\theta})
  \]

- The confidence is given by the posterior $p(y \mid x; \hat{\theta})$
Bayesian Learning

Bayesian approach

- Consider $\theta$ as a random variable, with a priori distribution $p(\theta)$
- The conditional model becomes $p(y \mid x, \theta)$
- **Training**: the posterior estimate of $\theta$ given $D$ is:

$$p(\theta \mid D) = \frac{p(D \mid \theta)p(\theta)}{p(D)} = \frac{\prod_t p(y^t \mid x^t, \theta)p(\theta)p^*(x)}{p(D)},$$

where $p^*(x)$ is the true distribution of inputs, which we will not be estimating and assume that $x$ is independent of $\theta$.
- Up to normalization: $p(\theta \mid D) \propto \prod_t p(y^t \mid x^t, \theta)p(\theta)$. Can compute for a given $\theta$ using all the data.
- **Testing**: given $x$, integrate out $\theta$:

$$p(y \mid x) = \int p(y \mid x, \theta)p(\theta \mid D)d\theta \quad \text{— in general intractable}$$
Example: Uniform Distribution

- Let $p(x; \theta)$ be a uniform distribution in $[0, \theta]$.
- Want to estimate $\theta$.
- Suppose we know a priori $\theta \in [0, 10]$, choose $p(\theta)$ uniform in $[0, 10]$.

Given a sample $\mathcal{D} = \{x_1, x_2, \ldots, x_n\}$, compute Bayesian estimate of $p(\theta \mid \mathcal{D})$:

$$
p(\theta \mid \mathcal{D}) \propto \prod_{i=1}^{n} p(x_i \mid \theta) p(\theta) = \prod_{i=1}^{n} \frac{1}{\theta} [x_i \leq \theta] p(\theta).
$$
Example: Uniform Distribution

Prior distribution $p(\theta)$

Measurements
ML plug−in $p(x | \theta)$
Bayesian $p(\theta | D)$
Bayesian $p(x)$
- **Proposition:** compute approximation to $p(\theta \mid D)$ by a simpler distribution $q(\theta)$.

- Let for example $\theta \in \mathbb{R}^d$ and

$$q(\theta) = \prod_{i=1}^{d} p_{\mathcal{N}}(\theta_i; \hat{\theta}, \hat{\sigma}^2)$$

- For each coordinate of $\theta$ we would like to estimate mean and variance.

- Recall the mean field approach:

$$\min_{q} KL(q(\theta) \| p(\theta \mid D))$$

- Only this time $\theta$ is continuous.

Sensible if expect the posterior to be concentrated around some point
Having $q$, the Bayesian posterior is approximated using distribution $q$ in place of $p(\theta \mid D)$:

$$p(y \mid x, D) \approx \int p(y \mid x, \theta) q(\theta) d\theta.$$
Variational Bayesian Learning

Solving the variational problem. Expand KL:

\[
KL(q(\theta) \| p(\theta \mid D)) = \mathbb{E}_{\theta \sim q(\theta)} \log \frac{q(\theta)}{p(\theta \mid D)} = \mathbb{E}_{\theta \sim q(\theta)} \log \prod_t p(y^t \mid x^t, \theta)p(\theta)/p(D)
\]

\[
= \mathbb{E}_{\theta \sim q(\theta)} \left[ -\sum_t \log p(y^t \mid x^t, \theta) \right] + KL(q(\theta) \| p(\theta)) + \log p(D).
\]

log likelihood, expected over parameters, data evidence

data-independent regularization

Special case I:
When we choose \( q \) to be the delta-function at \( \hat{\theta} \) (fix a tiny \( \hat{\sigma} \)) and the prior \( p(\theta) \) as \( \mathcal{N}(0, \sigma_0^2 I) \), the variational optimization becomes, up to constants,

\[
\min_{\hat{\theta}} \left[ -\sum_t \log p(y^t \mid x^t, \hat{\theta}) \right] + \frac{\|\hat{\theta}\|^2}{2\sigma_0^2},
\]

i.e., we recover the maximum likelihood, with a weight regularization.
Variational Bayesian Learning with SGD

\[ KL(q(\theta)\|p(\theta \mid D)) = \mathbb{E}_{\theta \sim q(\theta)} \left[ -\sum_t \log p(y^t \mid x^t, \theta) \right] + KL(q(\theta)\|p(\theta)) + \log p(D). \]

\[ \arg\min_q KL(q(\theta)\|p(\theta \mid D)) = \arg\min_q |D| \mathbb{E}_{\theta \sim q} [ -\log p(y \mid x, \theta) ] + KL(q(\theta)\|p(\theta)) \]

Special case II: \( q(\theta) = q(\theta \mid \phi) \) is Gaussian with parameters \( \phi \)

- \( KL(q(\theta)\|p(\theta)) \) is closed form for several types of priors \( p(\theta) \)

- Gradient in \( q \) of the data evidence expresses as:

\[ \frac{\partial}{\partial \phi} \mathbb{E}_{(x,y) \sim D} \left[ -\log p(y \mid x, \theta) \right] = \mathbb{E}_{(x,y) \sim D} \left[ -\frac{\partial}{\partial \phi} \log p(y \mid x, \theta) \right] \]

A stochastic estimate of the gradient can be made from few samples of the data and parameters — means we can apply SGD.
Stochastic gradient in $q$:

- pick a random training sample $(x^t, y^t)$ (or a batch)
- *sample* parameters $\theta$ from current posterior: $\theta \sim q(\theta)$
- Evaluate usual log likelihood $\log p(y^t | x^t, \theta)$
- add *regularizer*
- back propagate and perform a gradient descent step *in parameters of $q$*

Looks similar to training with dropout, doesn’t it?

References:

- Graves A.: Practical Variational Inference for Neural Networks, 2009
Books

• S. L. Lauritzen. Graphical models., 1996
• Sebastian Nowozin and Christoph H. Lampert. Structured Learning and Prediction in Computer Vision