Structured Prediction and Probabilistic Graphical Models

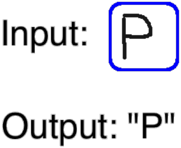
Mark Schmidt

University of British Columbia

August, 2015

Classic Machine Learning vs. Structured Prediction

Classical supervised learning: Output is one a single label.



Classic Machine Learning vs. Structured Prediction

Classical supervised learning: Output is one a single label.

Input: 

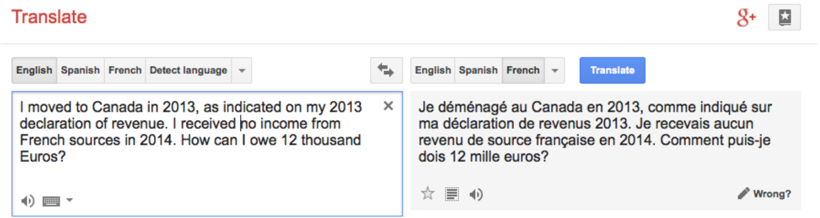
Output: "P"

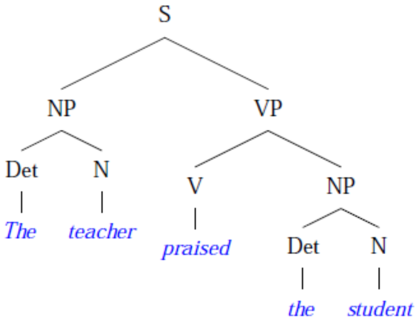
Structured prediction: Output can be a general object.

Input: 

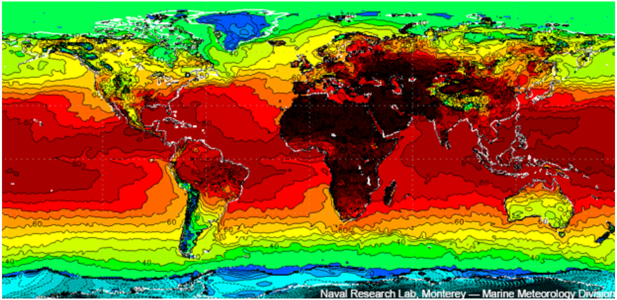
Output: "Paris"

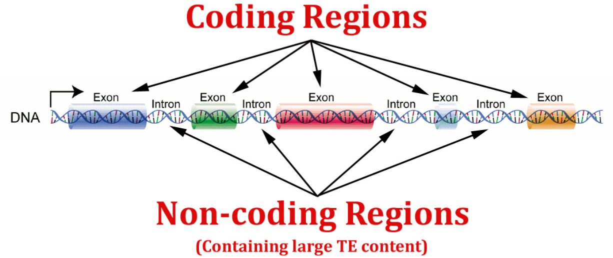
Examples of Structured Prediction



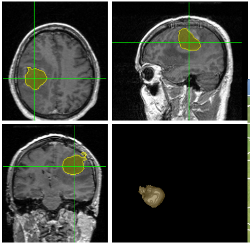


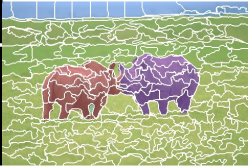
Examples of Structured Prediction

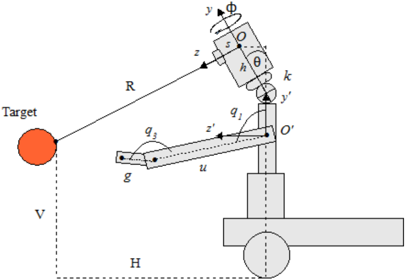




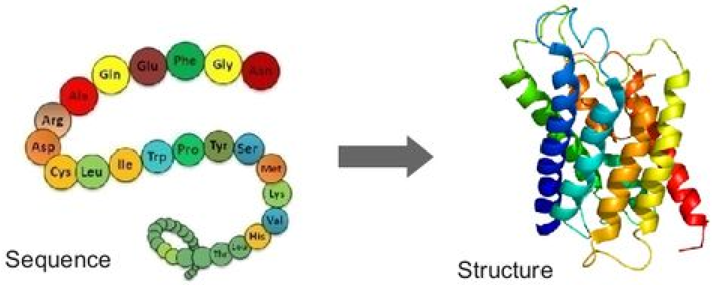
Examples of Structured Prediction







Examples of Structured Prediction

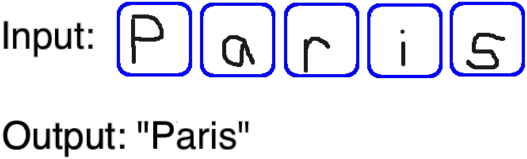


In 1917, Einstein applied the general theory of relativity to model the large-scale structure of the universe. He was visiting the United States when Adolf Hitler came to power in 1933 and did not go back to Germany, where he had been a professor at the Berlin Academy of Sciences. He settled in the U.S., becoming an American citizen in 1940. On the eve of World War II, he endorsed a letter to President Franklin D. Roosevelt alerting him to the potential development of "extremely powerful bombs of a new type" and recommending that the U.S. begin similar research. This eventually led to what would become the Manhattan Project. Einstein supported defending the Allied forces, but largely denounced using the new discovery of nuclear fission as a weapon. Later, with the British philosopher Bertrand Russell, Einstein signed the Russell-Einstein Manifesto, which highlighted the danger of nuclear weapons. Einstein was affiliated with the Institute for Advanced Study in Princeton, New Jersey, until his death in 1955.

Tag colours:

LOCATION TIME PERSON ORGANIZATION MONEY PERCENT DATE

“Classic” ML for Structured Prediction



Two obvious ways to solve this using “classic” machine learning:

“Classic” ML for Structured Prediction

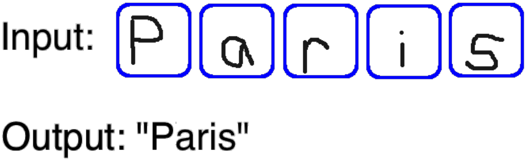
Input: 

Output: "Paris"

Two obvious ways to solve this using “classic” machine learning:

1. Treat each word as a different class label.
   - Problem: there are too many possible words.

“Classic” ML for Structured Prediction



Two obvious ways to solve this using “classic” machine learning:

1. Treat each word as a different class label.
   - Problem: there are too many possible words.
2. Predict each letter individually:
   - Works if you are really good at predicting individual letters.
   - Some tasks don’t have a natural decomposition.
   - Ignores dependencies between letters.

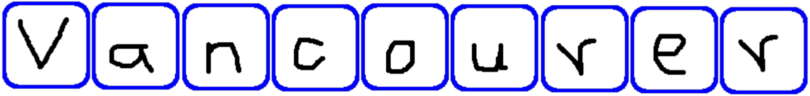
Motivation: Structured Prediction

- What letter is this?

Motivation: Structured Prediction

- What letter is this?

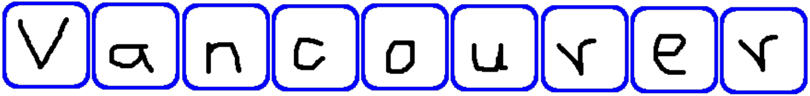
- What are these letters?



Motivation: Structured Prediction

- What letter is this?

- What are these letters?

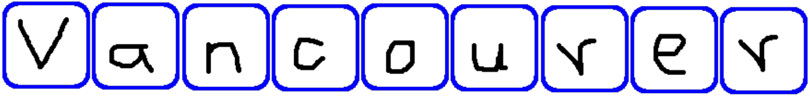


- Predict each letter using “classic” ML and neighbouring images?
    - Shoehorn this into a standard deep learning problem?

Motivation: Structured Prediction

- What letter is this?

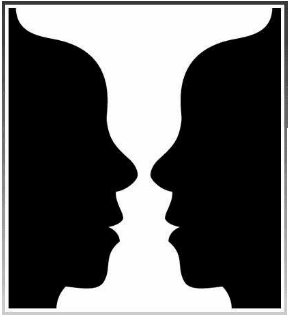
- What are these letters?

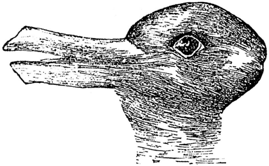


- Predict each letter using “classic” ML and neighbouring images?
  - Shoehorn this into a standard deep learning problem?
- Good or bad depending on loss function:
  - Good if you want to predict individual letters.
  - Bad if goal is to predict entire word.

Does the brain do structured prediction?

Gestalt effect: “whole is other than the sum of the parts”.





What do you see?
By shifting perspective you might see an old woman or a young woman.

Dealing with the Huge Number of Lables

- Structured prediction basic ideas:
  1. Define an energy function $E(Y|X)$:
     - Energy of output object Y given input object X.

(Low energy is better)

Dealing with the Huge Number of Lables

- Structured prediction basic ideas:
  1. Define an energy function $E(Y|X)$:
     - Energy of output object Y given input object X.

       (Low energy is better)
     - But the number of Y is huge: want to share information across Y.

Dealing with the Huge Number of Lables

- Structured prediction basic ideas:
  1. Define an energy function $E(Y|X)$:
     - Energy of output object Y given input object X.

       (Low energy is better)
     - But the number of Y is huge: want to share information across Y.
  2. Make the energy function depends on features $F(Y, X)$:
     - $F(Y_j, X_j)$: features for classifier of individual letter.

Dealing with the Huge Number of Lables

- Structured prediction basic ideas:
  1. Define an energy function $E(Y|X)$:
     - Energy of output object Y given input object X.

       (Low energy is better)
     - But the number of Y is huge: want to share information across Y.
  2. Make the energy function depends on features $F(Y, X)$:
     - $F(Y_j, X_j)$: features for classifier of individual letter.
     - $F(Y_{j-1}, Y_j)$: dependency between adjacent letters ('q-u').

Dealing with the Huge Number of Lables

- Structured prediction basic ideas:
  1. Define an energy function $E(Y|X)$:
     - Energy of output object Y given input object X.

       (Low energy is better)
     - But the number of Y is huge: want to share information across Y.
  2. Make the energy function depends on features $F(Y, X)$:
     - $F(Y_j, X_j)$: features for classifier of individual letter.
     - $F(Y_{j-1}, Y_j)$: dependency between adjacent letters ('q-u').
     - $F(Y_{j-1}, Y_j, X_{j-1}, X_j)$: adjacent letters and image dependency.

Dealing with the Huge Number of Lables

- Structured prediction basic ideas:
  1. Define an energy function $E(Y|X)$:
     - Energy of output object Y given input object X.

       (Low energy is better)
     - But the number of Y is huge: want to share information across Y.
  2. Make the energy function depends on features $F(Y, X)$:
     - $F(Y_j, X_j)$: features for classifier of individual letter.
     - $F(Y_{j-1}, Y_j)$: dependency between adjacent letters ('q-u').
     - $F(Y_{j-1}, Y_j, X_{j-1}, X_j)$: adjacent letters and image dependency.
     - $F(Y_{j-1}, Y_j, j)$: position-based dependency (French: 'e-r' ending).

Dealing with the Huge Number of Lables

- Structured prediction basic ideas:
  1. Define an energy function $E(Y|X)$:
     - Energy of output object Y given input object X.

       (Low energy is better)
     - But the number of Y is huge: want to share information across Y.
  2. Make the energy function depends on features $F(Y, X)$:
     - $F(Y_j, X_j)$: features for classifier of individual letter.
     - $F(Y_{j-1}, Y_j)$: dependency between adjacent letters ('q-u').
     - $F(Y_{j-1}, Y_j, X_{j-1}, X_j)$: adjacent letters and image dependency.
     - $F(Y_{j-1}, Y_j, j)$: position-based dependency (French: 'e-r' ending).
     - $F(Y_{j-2}, Y_{j-1}, Y_j, j)$: third-order and position (English: 'i-n-g' end).

Dealing with the Huge Number of Lables

- Structured prediction basic ideas:
  1. Define an energy function $E(Y|X)$:
     - Energy of output object Y given input object X.

       (Low energy is better)
     - But the number of Y is huge: want to share information across Y.
  2. Make the energy function depends on features $F(Y, X)$:
     - $F(Y_j, X_j)$: features for classifier of individual letter.
     - $F(Y_{j-1}, Y_j)$: dependency between adjacent letters ('q-u').
     - $F(Y_{j-1}, Y_j, X_{j-1}, X_j)$: adjacent letters and image dependency.
     - $F(Y_{j-1}, Y_j, j)$: position-based dependency (French: 'e-r' ending).
     - $F(Y_{j-2}, Y_{j-1}, Y_j, j)$: third-order and position (English: 'i-n-g' end).
     - $F(Y \in \mathcal{D})$: is y in dictionary $\mathcal{D}$?

Dealing with the Huge Number of Lables

- Structured prediction basic ideas:
  1. Define an energy function $E(Y|X)$:
     - Energy of output object Y given input object X.

       (Low energy is better)
     - But the number of Y is huge: want to share information across Y.
  2. Make the energy function depends on features $F(Y, X)$:
     - $F(Y_j, X_j)$: features for classifier of individual letter.
     - $F(Y_{j-1}, Y_j)$: dependency between adjacent letters ('q-u').
     - $F(Y_{j-1}, Y_j, X_{j-1}, X_j)$: adjacent letters and image dependency.
     - $F(Y_{j-1}, Y_j, j)$: position-based dependency (French: 'e-r' ending).
     - $F(Y_{j-2}, Y_{j-1}, Y_j, j)$: third-order and position (English: 'i-n-g' end).
     - $F(Y \in \mathcal{D})$: is y in dictionary $\mathcal{D}$?
  3. Learn the parameters of the energy function from data:
     - Learn parameters so that "correct" labels get low energy.
     - Features let us transfer knowledge to completely new labels.

       (E.g., predict a word you've never seen before.)

Inference in structured prediction

- Week 2 will discuss learning the energy function.
- Week 1 focuses on inference.

Inference in structured prediction

- Week 2 will discuss learning the energy function.
- Week 1 focuses on inference.
  - E.g., the decoding problem:

$$\max_{Y} E(Y|X).$$

  - Trivial in "classic" machine learning, now it can be hard.

    (don't want to measure energy of every possible word)

Inference in structured prediction

- Week 2 will discuss learning the energy function.
- Week 1 focuses on inference.
  - E.g., the decoding problem:
$$\max_Y E(Y|X).$$
  - Trivial in "classic" machine learning, now it can be hard.
(don't want to measure energy of every possible word)
  - We will also do inferences with the Gibbs/Boltzmann distribution:
$$p(Y|X) = \frac{\exp(-E(Y|X))}{Z},$$
where
$$Z = \sum_{Y'} \exp(-E(Y'|X)).$$
  - Z is called the normalizing constant or partition function

Undirected Graphical Models

- We'll focus on pairwise undirected graphical models (UGMs).
  - But basic ideas apply to other graphical models.

Undirected Graphical Models

- We'll focus on pairwise undirected graphical models (UGMs).
  - But basic ideas apply to other graphical models.
- This means our energy functions have the form

$$E(Y|X) = \sum_i f_i(Y_i, X) + \sum_{i,j} f_{i,j}(Y_i, Y_j, X),$$

Undirected Graphical Models

- We'll focus on pairwise undirected graphical models (UGMs).
  - But basic ideas apply to other graphical models.
- This means our energy functions have the form

$$E(Y|X) = \sum_i f_i(Y_i, X) + \sum_{i,j} f_{i,j}(Y_i, Y_j, X),$$

and our Gibbs distribution has the form

$$\begin{aligned}
P(Y|X) &= \frac{\exp(-\sum_i f_i(Y_i, X) - \sum_{i,j} f_{i,j}(Y_i, Y_j, X))}{Z} \\
&\propto \prod_i \exp(-f_i(Y_i, X)) \prod_{i,j} \exp(-f_{i,j}(Y_i, Y_j, X). \\
&= \prod_i \phi_i(Y_i, X) \prod_i \phi_{i,j}(Y_i, Y_j, X),
\end{aligned}$$

where the ϕ functions are called the potentials.

Undirected Graphical Models

- For pairwise UGMs our energy has the form

$$E(Y|X) = \sum_i f_i(Y_i, X) + \sum_{i,j} f_{i,j}(Y_i, Y_j, X).$$

Undirected Graphical Models

- For pairwise UGMs our energy has the form

$$E(Y|X) = \sum_i f_i(Y_i, X) + \sum_{i,j} f_{i,j}(Y_i, Y_j, X).$$

- We may not want a function $f_{i,j}$ between every pair i and j.
- E.g., for sequences we may only want $f_{j-1,j}$.

Undirected Graphical Models

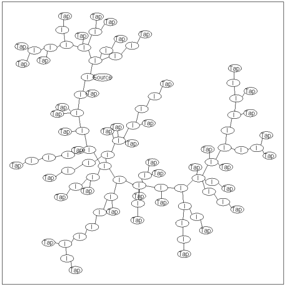
- For pairwise UGMs our energy has the form
$$E(Y|X) = \sum_i f_i(Y_i, X) + \sum_{i,j} f_{i,j}(Y_i, Y_j, X).$$
- We may not want a function $f_{i,j}$ between every pair i and j.
- E.g., for sequences we may only want $f_{j-1,j}$.
- We can draw a graph based on this:
  - Each node corresponds to a variable.
  - We have an edge between i and j if we have an $f_{i,j}$.

Undirected Graphical Models

- For pairwise UGMs our energy has the form

$$E(Y|X) = \sum_i f_i(Y_i, X) + \sum_{i,j} f_{i,j}(Y_i, Y_j, X).$$

- We may not want a function $f_{i,j}$ between every pair i and j.
- E.g., for sequences we may only want $f_{j-1,j}$.
- We can draw a graph based on this:
  - Each node corresponds to a variable.
  - We have an edge between i and j if we have an $f_{i,j}$.
  - E.g., tomorrow, we will consider this tree-structured graph:



3 Tasks in UGMs for Week 1

Week 1: ignore conditioning and consider generic $E(X)$ and $P(X)$.

3 Tasks in UGMs for Week 1

Week 1: ignore conditioning and consider generic $E(X)$ and $P(X)$.

1. Decoding: Compute the optimal configuration,

$$\min_X E(X).$$

3 Tasks in UGMs for Week 1

Week 1: ignore conditioning and consider generic $E(X)$ and $P(X)$.

1. Decoding: Compute the optimal configuration,

$$\min_X E(X).$$

2. Inference: Compute partition function and marginals,

$$Z = \sum_{X'} P(X'), \quad P(X_i = j) = \sum_{X' | X_i = j} p(X').$$

3 Tasks in UGMs for Week 1

Week 1: ignore conditioning and consider generic $E(X)$ and $P(X)$.

1. Decoding: Compute the optimal configuration,

$$\min_X E(X).$$

2. Inference: Compute partition function and marginals,

$$Z = \sum_{X'} P(X'), \quad P(X_i = j) = \sum_{X' | X_i = j} p(X').$$

3. Sampling: Generate X' according to Gibbs distribution:

$$X' \sim P(X).$$

In UGMs, efficiency of these tasks is related to graph structure.

3 Tasks by Hand on a Simple Example

- To illustrate the tasks, let's take a simple 2-variable example,

$$E(x_1, x_2) = -f_1(x_1) - f_2(x_2) - f_{1,2}(x_1, x_2),$$

where

$$f_1(x_1) = \begin{cases} 1 & x_1 = 0 \\ 2 & x_1 = 1 \end{cases}, \quad f_2(x_2) = \begin{cases} 1 & x_1 = 0 \\ 3 & x_2 = 1 \end{cases}, \quad f_{1,2}(x_1, x_2) = \begin{cases} 2 & x_1 = x_2 \\ 1 & x_1 \neq x_2 \end{cases}$$

3 Tasks by Hand on a Simple Example

- To illustrate the tasks, let's take a simple 2-variable example,

$$E(x_1, x_2) = -f_1(x_1) - f_2(x_2) - f_{1,2}(x_1, x_2),$$

  where

$$f_1(x_1) = \begin{cases} 1 & x_1 = 0 \\ 2 & x_1 = 1 \end{cases}, \quad f_2(x_2) = \begin{cases} 1 & x_1 = 0 \\ 3 & x_2 = 1 \end{cases}, \quad f_{1,2}(x_1, x_2) = \begin{cases} 2 & x_1 = x_2 \\ 1 & x_1 \neq x_2 \end{cases}$$

- x_1 wants to be 1, x_2 really wants to be 1, both want to be same.

3 Tasks by Hand on a Simple Example

- To illustrate the tasks, let's take a simple 2-variable example,

$$E(x_1, x_2) = -f_1(x_1) - f_2(x_2) - f_{1,2}(x_1, x_2),$$

where

$$f_1(x_1) = \begin{cases} 1 & x_1 = 0 \\ 2 & x_1 = 1 \end{cases}, \quad f_2(x_2) = \begin{cases} 1 & x_1 = 0 \\ 3 & x_2 = 1 \end{cases}, \quad f_{1,2}(x_1, x_2) = \begin{cases} 2 & x_1 = x_2 \\ 1 & x_1 \neq x_2 \end{cases}$$

- x_1 wants to be 1, x_2 really wants to be 1, both want to be same.
- We can think of the possible states/energies in a big table:

x_1	x_2	f_1	f_2	$f_{1,2}$	$-E(x_1, x_2)$

3 Tasks by Hand on a Simple Example

- To illustrate the tasks, let's take a simple 2-variable example,

$$E(x_1, x_2) = -f_1(x_1) - f_2(x_2) - f_{1,2}(x_1, x_2),$$

  where

$$f_1(x_1) = \begin{cases} 1 & x_1 = 0 \\ 2 & x_1 = 1 \end{cases}, \quad f_2(x_2) = \begin{cases} 1 & x_1 = 0 \\ 3 & x_2 = 1 \end{cases}, \quad f_{1,2}(x_1, x_2) = \begin{cases} 2 & x_1 = x_2 \\ 1 & x_1 \neq x_2 \end{cases}$$

- x_1 wants to be 1, x_2 really wants to be 1, both want to be same.
- We can think of the possible states/energies in a big table:

x_1	x_2	f_1	f_2	$f_{1,2}$	$-E(x_1, x_2)$
0	0				

3 Tasks by Hand on a Simple Example

- To illustrate the tasks, let's take a simple 2-variable example,

$$E(x_1, x_2) = -f_1(x_1) - f_2(x_2) - f_{1,2}(x_1, x_2),$$

where

$$f_1(x_1) = \begin{cases} 1 & x_1 = 0 \\ 2 & x_1 = 1 \end{cases}, \quad f_2(x_2) = \begin{cases} 1 & x_1 = 0 \\ 3 & x_2 = 1 \end{cases}, \quad f_{1,2}(x_1, x_2) = \begin{cases} 2 & x_1 = x_2 \\ 1 & x_1 \neq x_2 \end{cases}$$

- x_1 wants to be 1, x_2 really wants to be 1, both want to be same.
- We can think of the possible states/energies in a big table:

x_1	x_2	f_1	f_2	$f_{1,2}$	$-E(x_1, x_2)$
0	0	1	1	2	

3 Tasks by Hand on a Simple Example

- To illustrate the tasks, let's take a simple 2-variable example,

$$E(x_1, x_2) = -f_1(x_1) - f_2(x_2) - f_{1,2}(x_1, x_2),$$

where

$$f_1(x_1) = \begin{cases} 1 & x_1 = 0 \\ 2 & x_1 = 1 \end{cases}, \quad f_2(x_2) = \begin{cases} 1 & x_1 = 0 \\ 3 & x_2 = 1 \end{cases}, \quad f_{1,2}(x_1, x_2) = \begin{cases} 2 & x_1 = x_2 \\ 1 & x_1 \neq x_2 \end{cases}$$

- x_1 wants to be 1, x_2 really wants to be 1, both want to be same.
- We can think of the possible states/energies in a big table:

x_1	x_2	f_1	f_2	$f_{1,2}$	$-E(x_1, x_2)$
0	0	1	1	2	4

3 Tasks by Hand on a Simple Example

- To illustrate the tasks, let's take a simple 2-variable example,

$$E(x_1, x_2) = -f_1(x_1) - f_2(x_2) - f_{1,2}(x_1, x_2),$$

where

$$f_1(x_1) = \begin{cases} 1 & x_1 = 0 \\ 2 & x_1 = 1 \end{cases}, \quad f_2(x_2) = \begin{cases} 1 & x_1 = 0 \\ 3 & x_2 = 1 \end{cases}, \quad f_{1,2}(x_1, x_2) = \begin{cases} 2 & x_1 = x_2 \\ 1 & x_1 \neq x_2 \end{cases}$$

- x_1 wants to be 1, x_2 really wants to be 1, both want to be same.
- We can think of the possible states/energies in a big table:

x_1	x_2	f_1	f_2	$f_{1,2}$	$-E(x_1, x_2)$
0	0	1	1	2	4
0	1				

3 Tasks by Hand on a Simple Example

- To illustrate the tasks, let's take a simple 2-variable example,

$$E(x_1, x_2) = -f_1(x_1) - f_2(x_2) - f_{1,2}(x_1, x_2),$$

where

$$f_1(x_1) = \begin{cases} 1 & x_1 = 0 \\ 2 & x_1 = 1 \end{cases}, \quad f_2(x_2) = \begin{cases} 1 & x_1 = 0 \\ 3 & x_2 = 1 \end{cases}, \quad f_{1,2}(x_1, x_2) = \begin{cases} 2 & x_1 = x_2 \\ 1 & x_1 \neq x_2 \end{cases}$$

- x_1 wants to be 1, x_2 really wants to be 1, both want to be same.
- We can think of the possible states/energies in a big table:

x_1	x_2	f_1	f_2	$f_{1,2}$	$-E(x_1, x_2)$
0	0	1	1	2	4
0	1	1	3	1	5

3 Tasks by Hand on a Simple Example

- To illustrate the tasks, let's take a simple 2-variable example,

$$E(x_1, x_2) = -f_1(x_1) - f_2(x_2) - f_{1,2}(x_1, x_2),$$

  where

$$f_1(x_1) = \begin{cases} 1 & x_1 = 0 \\ 2 & x_1 = 1 \end{cases}, \quad f_2(x_2) = \begin{cases} 1 & x_1 = 0 \\ 3 & x_2 = 1 \end{cases}, \quad f_{1,2}(x_1, x_2) = \begin{cases} 2 & x_1 = x_2 \\ 1 & x_1 \neq x_2 \end{cases}$$

- x_1 wants to be 1, x_2 really wants to be 1, both want to be same.
- We can think of the possible states/energies in a big table:

x_1	x_2	f_1	f_2	$f_{1,2}$	$-E(x_1, x_2)$
0	0	1	1	2	4
0	1	1	3	1	5
1	0	2	1	1	4
1	1	2	3	2	7

Decoding on Simple Example

x_1	x_2	f_1	f_2	$f_{1,2}$	$-E(x_1, x_2)$
0	0	1	1	2	4
0	1	1	3	1	5
1	0	2	1	1	4
1	1	2	3	2	7

- Decoding is finding the minimizer of $E(x_1, x_2)$:

Decoding on Simple Example

x_1	x_2	f_1	f_2	$f_{1,2}$	$-E(x_1, x_2)$
0	0	1	1	2	4
0	1	1	3	1	5
1	0	2	1	1	4
1	1	2	3	2	7

- Decoding is finding the minimizer of $E(x_1, x_2)$:
  - In this case it is $x_1 = 1$ and $x_2 = 1$.

Inference on Simple Example

x_1	x_2	f_1	f_2	$f_{1,2}$	$-E(x_1, x_2)$
0	0	1	1	2	4
0	1	1	3	1	5
1	0	2	1	1	4
1	1	2	3	2	7

- One inference task is finding Z:

Inference on Simple Example

x_1	x_2	f_1	f_2	$f_{1,2}$	$-E(x_1, x_2)$
0	0	1	1	2	4
0	1	1	3	1	5
1	0	2	1	1	4
1	1	2	3	2	7

- One inference task is finding Z:
  - In this case $Z = \exp(4) + \exp(5) + \exp(4) + \exp(7) \approx 1354$.

Inference on Simple Example

x_1	x_2	f_1	f_2	$f_{1,2}$	$-E(x_1, x_2)$	$p(x_1, x_2)$
0	0	1	1	2	4	0.04
0	1	1	3	1	5	0.11
1	0	2	1	1	4	0.04
1	1	2	3	2	7	0.81

- One inference task is finding Z:
  - In this case $Z = \exp(4) + \exp(5) + \exp(4) + \exp(7) \approx 1354$.
- With Z you can find the probability of configurations:
  - E.g., $p(x_1 = 0, x_2 = 0) = \exp(4)/Z = 0.04$.

Inference on Simple Example

x_1	x_2	f_1	f_2	$f_{1,2}$	$-E(x_1, x_2)$	$p(x_1, x_2)$
0	0	1	1	2	4	0.04
0	1	1	3	1	5	0.11
1	0	2	1	1	4	0.04
1	1	2	3	2	7	0.81

- One inference task is finding Z:
  - In this case $Z = \exp(4) + \exp(5) + \exp(4) + \exp(7) \approx 1354$.
- With Z you can find the probability of configurations:
  - E.g., $p(x_1 = 0, x_2 = 0) = \exp(4)/Z = 0.04$.
- Inference also includes finding marginals like $p(x_1 = 1)$:
  - E.g, $p(x_1 = 1) = \sum_{x_2} p(x_1 = 1, x_2) = 0.04 + 0.81 = 0.85$.

Sampling on Simple Example

x_1	x_2	f_1	f_2	$f_{1,2}$	$-E(x_1, x_2)$	$p(x_1, x_2)$	cumsum
0	0	1	1	2	4	0.04	0.04
0	1	1	3	1	5	0.11	0.15
1	0	2	1	1	4	0.04	0.19
1	1	2	3	2	7	0.81	1.00

- Sampling is generating configurations according to $p(x_1, x_2)$:
  - E.g., 81% of the time we should return $x_1 = 1$ and $x_2 = 1$.

Sampling on Simple Example

x_1	x_2	f_1	f_2	$f_{1,2}$	$-E(x_1, x_2)$	$p(x_1, x_2)$	cumsum
0	0	1	1	2	4	0.04	0.04
0	1	1	3	1	5	0.11	0.15
1	0	2	1	1	4	0.04	0.19
1	1	2	3	2	7	0.81	1.00

- Sampling is generating configurations according to $p(x_1, x_2)$:
  - E.g., 81% of the time we should return $x_1 = 1$ and $x_2 = 1$.
- To implement this:
  1. Generate a random number $u \in [0, 1]$.
  2. Find the smallest cumsum of the probabilities greater than u.

  - If $u = 0.59$ return $x_1 = 1$ and $x_2 = 1$
  - If $u = 0.12$ return $x_1 = 0$ and $x_2 = 1$.

Homework: First two UGM demos

For tomorrow, download UGM and read/run the first two demos:

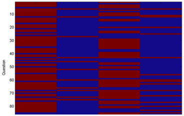
Small UGM Demo

In this demo, we use a very simple undirected graphical model (UGM) to represent a very simple probabilistic scenario, show how to input the model into UGM, and perform decoding/inference/sampling in the model.

Cheating Students Scenario

Consider a scenario where four students (Cathy, Heather, Mark, and Allison) have to write two multiple choice tests. For the first test, all four students are put in different rooms. Since Heather and Allison studied very hard, they get 90% of the questions right. On the other hand, Cathy and Mark didn't study, so they only pick the right choice 25% of the time.

We will assume that the chance of a student answering a question right is independent of the question (there are no 'hard' or 'easy' questions, and Allison/Heather don't make the same mistakes because they studied together). If we simulate 100 independent questions under this scenario, the results would look something like:



Chain UGM Demo

The naive methods for decoding/inference/sampling discussed in the previous demo require exponential time in both the number of nodes and the number of states, so they are completely impractical if the number of nodes or the number of states is non-trivial. In this demo, we consider the case where the graph structure is a chain. By taking advantage of the conditional independence properties induced by the chain structure, it is possible to derive polynomial-time methods for decoding/inference/sampling. These methods can be applied even when the chain is very long, and when each node in the chain can take many values.

Computer Science Graduate Careers

In this demo, we will build a simple Markov chain model of what people do after they graduate with a computer science degree. In this model, we will assume that every new computer science grad initially starts out in one of three states:

State	Probability	Description
Industry	0.60	They work for a company or own their own company.
Grad School	0.30	They are trying to get a Masters or PhD degree.
Video Games	0.10	They mostly play video games.

After each year, the grad can either stay in the same state, or transition to a different state. We will build a model for the first sixty years after graduation. Besides the three possible initial states above, we will also consider four states that you could transition into:

State	Description
Industry (with PhD)	They work for a company or own their own company, and have a PhD.
Academia	They work as a post-doctoral researcher or professor.
Video Games (with PhD)	They mostly play video games, but have a PhD.
Deceased	They no longer work in computer science.

Note that we haven't yet stated the probability of transitioning into each of these states, since we will make these probabilities

Reviews/expands on material from today, introduces Markov chains.

(should take less 15 minutes)

Schedule

M	T	W	R	F
Motivation/Exact/Small Mark	Chain/Tree Mark	Condition/Cutset/Supernodes Julie	Junction Tree Mehran	Semi-Markov/Graph Cuts Alireza
MRF/CRF/SSVM Mark	ICM/Block/Alpha Julieta	MCMC/Herding Jason	Hidden/RBM/Younes Ankur	Structure Learning Sharan
Variational/MF Mark	Bethe/Kikuchi Nasim	TRBP/Convex Reza	LP/SDP Issam	BCFW Reza