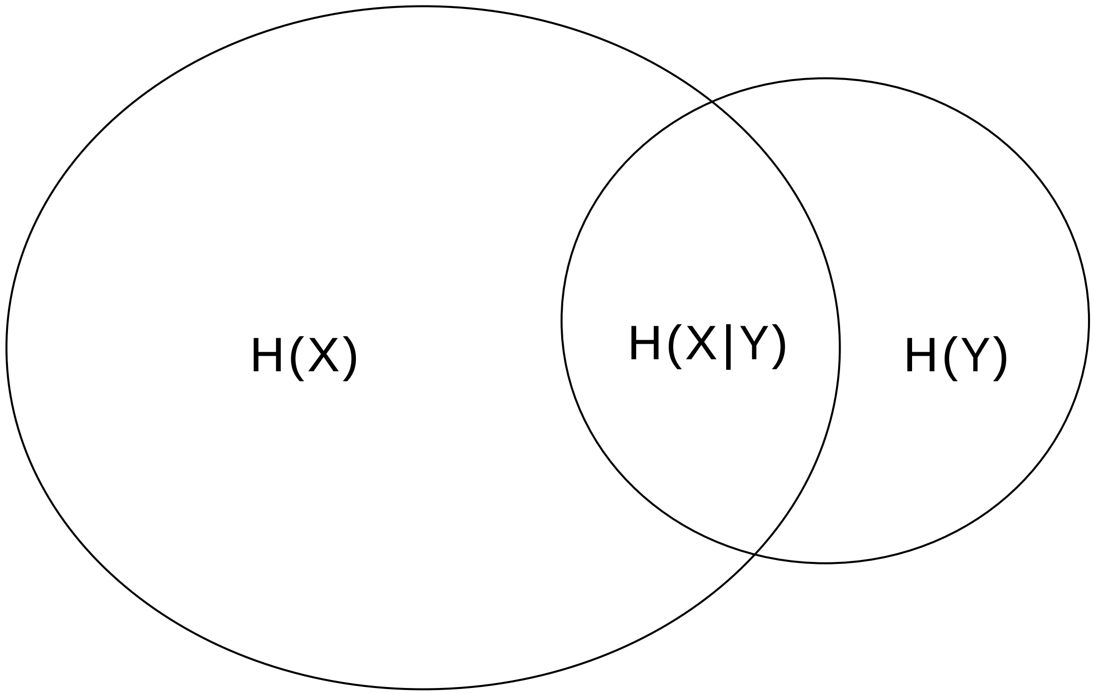
Lecture 11:
Context Coding

Thinh Nguyen
Oregon State University

Outline

- PPM
- JBIG
- Burrows-Wheeler Transform (bzip2)
- Move-To-Front

Context Coding

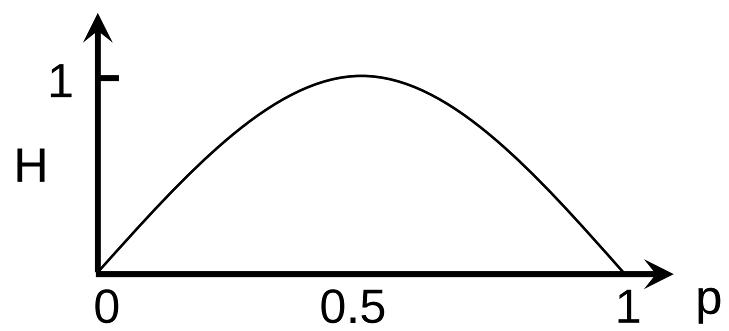


H(X|Y) <= H(X) → H(X|Y) takes fewer bits to code than H(X)

Context Coding

- The distribution of the next symbol based on the current context (past symbols) is skewed.

  - Next symbol is likely to be a certain alphabet the than others.
  - Less information
  - Easier to code.



Context Coding

- From information theory – The lower the information, the fewer bits are needed to code the symbol.

$$\inf(a) = \log_2\left(\frac{1}{P(a)}\right)$$

- Examples:
  - P(a) = 1023/1024, inf(a) = .000977
  - P(a) = 1/2, inf(a) = 1
  - P(a) = 1/1024, inf(a) = 10

Review of Entropy

- Entropy is the expected number of bit to code a symbol in the model with a_i having probability $P(a_i)$.

$$H = \sum_{i=1}^{m} P(a_i)\log_2\left(\frac{1}{P(a_i)}\right)$$

- Good coders should be close to this bound.
  - Arithmetic
  - Huffman
  - Golomb
  - Tunstall

Problem with Context Coding

- Context explosion!
  - Suppose we want to use 5-letter context to predict the next letter in an English paragraph.
  - Number of contexts = 24^{5}.
  - No storage for contexts.
  - Speed

Which Context to Use?

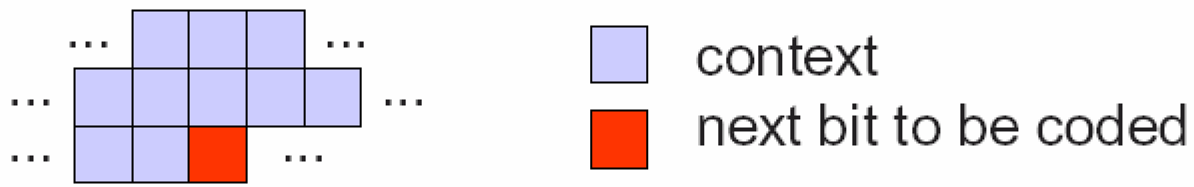
- Using previous table, which context for italicized letter?
  - “We pulled a he***a***vy wagon.”
  - “The the***a***tre was fun.”
  - “’Twas the***e***re haus!”

PPM- Prediction with Partial Matching

- Cleary and Witten (1984)
- Uses only current contexts (not all possible contexts)
- Uses arithmetic coding to code the context

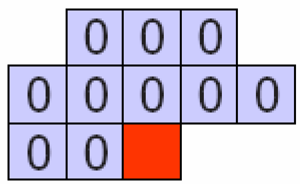
JBIG

- Coder for binary images
  - documents
  - graphics
- Codes in scan line order using context from the same and previous scan lines.



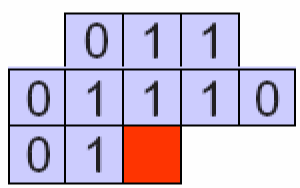
- Uses adaptive arithmetic coding with context.

JBIG Example



next bit	0	1
frequency	100	10

$$H = \frac{10}{110}\log\left(\frac{110}{10}\right) + \frac{100}{110}\log\left(\frac{110}{100}\right) = .44$$



next bit	0	1
frequency	15	50

$$H = \frac{15}{65}\log\left(\frac{65}{15}\right) + \frac{50}{65}\log\left(\frac{65}{50}\right) = .78$$

Issues with Context

- Context dilution
  - If there are too many contexts then too few symbols are coded in each context, making them in effective because of the zero-frequency problem.
- Context saturation
  - If there are too few contexts then the contexts might not be as good as having more contexts.
- Wrong context
  - poor predictors.

Burrows-Wheeler Transform

- Burrows-Wheeler (1994)
- BW Transform creates a representation of the data which has a small working set.
- The transformed data is compressed with move to front compression.
- The decoder is quite different from the encoder.
- The algorithm requires processing the entire string at once (it is not on-line).
- It is a remarkably good compression method.

Encoding Example

- abracadabra

1. Create all cyclic shifts of the string.

0	abracadabra
1	bracadabraa
2	racadabraab
3	acadabraabr
4	cadabraabra
5	adabraabrac
6	dabraabraca
7	abraabracad
8	braabracada
9	raabracadab
10	aabracadabr

Encoding Example

2. Sort the strings alphabetically in to array A

			A	
0	abracadabra		0	aabracadabr
1	bracadabraa		1	abraabracad
2	racadabraab		2	abracadabra
3	acadabraabr		3	acadabraabr
4	cadabraabra	→	4	adabraabrac
5	adabraabrac		5	braabracada
6	dabraabraca		6	bracadabraa
7	abraabracad		7	cadabraabra
8	braabracada		8	dabraabraca
9	raabracadab		9	raabracadab
10	aabracadabr		10	racadabraab

Encoding Example

3. L = the last column

A	
0	aabracadabr
1	abraabracad
2	abracadabra
3	acadabraabr
4	adabraabrac
5	braabracada
6	bracadabraa
7	cadabraabra
8	dabraabraca
9	raabracadab
10	racadabraab

L = rdarcaaaabb

Encoding Example

4. Transmit X the index of the input in A and L (using move to front coding).

A

0	aabracadabr
1	abraabracad
2	abracadabra
3	acadabraabr
4	adabraabrac
5	braabracada
6	bracadabraa
7	cadabraabra
8	dabraabraca
9	raabracadab
10	racadabraab

L = rdarcaaaabb

X = 2

Why Does BW Work?

- Ignore decoding for the moment.

- The prefix of each shifted string is a context for the last symbol.
  - The last symbol appears just before the prefix in the original.

- By sorting, similar contexts are adjacent.
  - This means that the predicted last symbols are similar.

Decoding Example

- We first decode assuming some information. We then show how compute the information.
- Let A^s be A shifted by 1

A		A^s	
0	aabracadabr	0	raabracadab
1	abraabracad	1	dabraabraca
2	abracadabra	2	aabracadabr
3	acadabraabr	3	racadabraab
4	adabraabrac	4	cadabraabra
5	braabracada	5	abraabracad
6	bracadabraa	6	abracadabra
7	cadabraabra	7	acadabraabr
8	dabraabraca	8	adabraabrac
9	raabracadab	9	braabracada
10	racadabraab	10	bracadabraa

Decoding Example

- Assume we know the mapping T[i] is the index in A^s of the string i in A.
- T = [2 5 6 7 8 9 10 4 1 0 3]

A		A^s	
0	aabracadabr	0	raabracadab
1	abraabracad	1	dabraabraca
2	abracadabra	2	aabracadabr
3	acadabraabr	3	racadabraab
4	adabraabrac	4	cadabraabra
5	braabracada	5	abraabracad
6	bracadabraa	6	abracadabra
7	cadabraabra	7	acadabraabr
8	dabraabraca	8	adabraabrac
9	raabracadab	9	braabracada
10	racadabraab	10	bracadabraa

Decoding Example

- Let F be the first column of A, it is just L, sorted.

F =

0	1	2	3	4	5	6	7	8	9	10
a	a	a	a	a	b	b	c	d	r	r

T =

0	1	2	3	4	5	6	7	8	9	10
2	5	6	7	8	9	10	4	1	0	3

- Follow the pointers in T in F to recover the input starting with X.

Decoding Example

X = 2

F =

0	1	<u>2</u>	3	4	5	6	7	8	9	10
a	a	a	a	a	b	b	c	d	r	r

T =

0	1	<u>2</u>	3	4	5	6	7	8	9	10
2	5	6	7	8	9	10	4	1	0	3

a

Decoding Example

F =

0	1	2	3	4	5	6	7	8	9	10
a	a	a	a	a	b	b	c	d	r	r

T =

0	1	2	3	4	5	6	7	8	9	10
2	5	6	7	8	9	10	4	1	0	3

ab

Decoding Example

F =

0	1	<u>2</u>	3	4	5	<u>6</u>	7	8	9	<u>10</u>
a	a	a	a	a	b	b	c	d	r	r

T =

0	1	<u>2</u>	3	4	5	<u>6</u>	7	8	9	<u>10</u>
2	5	6	7	8	9	10	4	1	0	3

abr

Decoding Example

- Why does this work?
- The first symbol of A[T[i]] is the second symbol of A[i] because A^s[T[i]] = A[i].

A		A^s	
0	aabracadabr	0	raabracadab
1	abraabracad	1	dabraabraca
2	abracadabra	2	aabracadabr
3	acadabraabr	3	racadabraab
4	adabraabrac	4	cadabraabra
5	braabracada	5	abraabracad
6	bracadabraa	6	abracadabra
7	cadabraabra	7	acadabraabr
8	dabraabraca	8	adabraabrac
9	raabracadab	9	braabracada
10	racadabraab	10	bracadabraa

Decoding Example

- How do we compute T from L and X?

	0	1	2	3	4	5	6	7	8	9	10
F =	a	a	a	a	a	b	b	c	d	r	r
L =	r	d	a	r	c	a	a	a	a	b	b

Note that L is the first column of A^s and A^s is in the same order as A.

If *i* is the *k*-th *x* in F then T[i] is the *k*-th *x* in L.

Decoding Example

	0	1	2	3	4	5	6	7	8	9	10
F =	a	a	a	a	a	b	b	c	d	r	r
L =	r	d	a	r	c	a	a	a	a	b	b

	0	1	2	3	4	5	6	7	8	9	10
T=	2	5	6	7	8						

Decoding Example

	0	1	2	3	4	5	6	7	8	9	10
F =	a	a	a	a	a	b	b	c	d	r	r
L =	r	d	a	r	c	a	a	a	a	b	b

T= 0 1 2 3 4 5 6 7 8 9 10

2 5 6 7 8 9 10

Decoding Example

	0	1	2	3	4	5	6	7	8	9	10
F =	a	a	a	a	a	b	b	c	d	r	r
L =	r	d	a	r	c	a	a	a	a	b	b

	0	1	2	3	4	5	6	7	8	9	10
T=	2	5	6	7	8	9	10	4			

Decoding Example

	0	1	2	3	4	5	6	7	8	9	10
F =	a	a	a	a	a	b	b	c	d	r	r
L =	r	d	a	r	c	a	a	a	a	b	b

	0	1	2	3	4	5	6	7	8	9	10
T=	2	5	6	7	8	9	10	4	1		

Decoding Example

	0	1	2	3	4	5	6	7	8	9	10
F =	a	a	a	a	a	b	b	c	d	r	r
L =	r	d	a	r	c	a	a	a	a	b	b

T =	0	1	2	3	4	5	6	7	8	9	10
	2	5	6	7	8	9	10	4	1	0	3

Notes on BW

- Alphabetic sorting does not need the entire cyclic shifted inputs. You just have to look at long enough prefixes.
  - A bucket sort will work here.
- Requires *entire* input. In practice, that's impossible.
  - Break input into blocks.
- There are high quality practical implementations:
  - Bzip
  - Bzip2 (public domain)

Move To Front Algorithm

- MTF is part of Burrows-Wheeler, basis for bzip2!
- Non-numerical data.
- The data have a relatively small working set that changes over the sequence.
  - Example: a b a b a a b c c b b c c c c b d b c c
- Move to Front algorithm:
  - Symbols are kept in a list indexed 0 to m-1.
  - To code a symbol output its index and move the symbol to the front of the list.

Example

- Example: a b a b a a b c c b b c c c c b d b c c
  0

0	1	2	3
a	b	c	d

Example

- Example: a b a b a a b c c b b c c c c b d b c c
  0 1

0	1	2	3
a	b	c	d

↓

0	1	2	3
b	a	c	d

Example

- Example: a b a b a a b c c b b c c c c b d b c c
  0 1 1

0	1	2	3
b	a	c	d

↓

0	1	2	3
a	b	c	d

Example

- Example: a b a b a a b c c b b c c c c b d b c c
  0 1 1 1

0	1	2	3
a	b	c	d

↓

0	1	2	3
b	a	c	d

Example

- Example: a b a b a a b c c b b c c c c b d b c c
  0 1 1 1 1

0	1	2	3
b	a	c	d

↓

0	1	2	3
a	b	c	d

Example

- Example: a b a b a a b c c b b c c c c b d b c c
  0 1 1 1 1 0

0	1	2	3
a	b	c	d

Example

- Example: a b a b a a b c c b b c c c c b d b c c
  0 1 1 1 1 0 1

0	1	2	3
a	b	c	d

↓

0	1	2	3
b	a	c	d

Example

- Example: a b a b a a b c c b b c c c c b d b c c
  0 1 1 1 1 0 1 2

0	1	2	3
b	a	c	d

↓

0	1	2	3
c	b	a	d

Example

- Example: a b a b a a b c c b b c c c c b d b c c
  0 1 1 1 1 0 1 2 0 1 0 1 0 00 1 3 1 2 0

0	1	2	3
c	b	d	a

Example

- Example: a b a b a a b c c b b c c c c b d b c c
  0 1 1 1 1 0 1 2 0 1 0 1 0 00 1 3 1 2 0

Frequencies of {a, b, c, d}

a	b	c	d
4	7	8	1

Frequencies of {0, 1, 2, 3}

0	1	2	3
8	9	2	1

Extreme Example

Input:

aaaaaaaaaabbbbbbbbbbccccccccccdddddddddd

Output

0000000000100000000020000000003000000000

Frequencies of a b c d

a	b	c	d
10	10	10	10

Frequencies of 0 1 2 3

0	1	2	3
37	1	1	1