EC5.102: Information and Communication

(Lec-4)

Source coding-4

(10-March-2025)

Arti D. Yardi

Email address: arti.yardi@iiit.ac.in
Office: A2-204, SPCRC, Vindhya A2, 1st floor

Summary of the last class

Recap

- Introduction to source coding:
  - Definition
  - Expected length of code $L(C)$
- Types of source codes:
  - Singular/Non-singular codes
  - Uniquely decodable codes
  - Prefix or instantaneous

Recap

- Introduction to source coding:
  - Definition
  - Expected length of code $L(C)$
- Types of source codes:
  - Singular/Non-singular codes
  - Uniquely decodable codes
  - Prefix or instantaneous
- Algorithms for constructing source codes:
  - Huffman codes
  - Lempel-Ziv algorithm

Recap

- Introduction to source coding:
  - Definition
  - Expected length of code $L(C)$
- Types of source codes:
  - Singular/Non-singular codes
  - Uniquely decodable codes
  - Prefix or instantaneous
- Algorithms for constructing source codes:
  - Huffman codes
  - Lempel-Ziv algorithm
  - Encoding

Recap

- Introduction to source coding:
  - Definition
  - Expected length of code $L(C)$
- Types of source codes:
  - Singular/Non-singular codes
  - Uniquely decodable codes
  - Prefix or instantaneous
- Algorithms for constructing source codes:
  - Huffman codes
  - Lempel-Ziv algorithm
  - Encoding
  - Decoding

Lempel Ziv coding

Lempel-Ziv (L-Z) Coding: Example 1

- Consider the sequence: 1 0 1 0 1 1 0 1 1 0 1 0 1 0 1 0 1 0
- Parsing: Identify phrases of the smallest length that haven't appeared before.
- After parsing the phrases are: 1, 0, 1 0, 1 1, 0 1, 1 0 1, 0 1 0, 1 0 1 0
- Notice: New phrase is concatenation of a previous phrase and a new source message.
- Encoding: Lexicographic (dictionary) ordering of the previous phrase and the new source message are concatenated.

In which dictionary location the longest prefix of the content appeared before

The last message of the content

Dictionary location	Contents	Codeword
1	1	(0,1)
2	0	(0,0)
3	10	(1,0)
4	11	(1,1)
5	01	(2,1)
6	101	(3,1)
7	010	(5,0)
8	1010	(6,0)

Lempel-Ziv (L-Z) Coding

The steps of L-Z algorithm are as follows:

- Any sequence of the source output is uniquely parsed into phrases of varying length and these phrases are encoded using codewords of equal length. (This is a variable-to-fixed length coding scheme.)
- Parsing is done by identifying phrases of the smallest length that have not appeared before.
- The new phrase is the concatenation of a previous phrase and a new source message.
- Encoding: Lexicographic (dictionary) ordering of the previous phrase and the new source message are concatenated.

Today's agenda

- Kraft inequality
- Show that: For Prefix codes, $L(C) \geq H(X)$
- Statement of source coding theorem

Kraft inequality

Kraft inequality: Statement

- Suppose we wish to construct an **instantaneous code (prefix code)**

Kraft inequality: Statement

- Suppose we wish to construct an **instantaneous code (prefix code)**
- Let $\ell_1, \ell_2, \ldots, \ell_m$ be codeword lengths of an instantaneous code.

Kraft inequality: Statement

- Suppose we wish to construct an **instantaneous code (prefix code)**
- Let $\ell_1, \ell_2, \ldots, \ell_m$ be codeword lengths of an instantaneous code.
- Kraft inequality: Any binary instantaneous code with lengths $\ell_1, \ell_2, \ldots, \ell_m$ should satisfy:

$$\sum_{i=1}^{m} 2^{-\ell_i} \leq 1.$$

- Proof: In Class

Kraft inequality:Proof outline

Kraft inequality: Any binary instantaneous code with lengths $\ell_1, \ell_2, \ldots, \ell_m$ should satisfy: $\sum_{i=1}^{m} 2^{-\ell_i} \leq 1$.

Kraft inequality:Proof outline

Kraft inequality: Any binary instantaneous code with lengths $\ell_1, \ell_2, \ldots, \ell_m$ should satisfy: $\sum_{i=1}^{m} 2^{-\ell_i} \leq 1$.

- Without loss of generality, suppose $\ell_1 \leq \ell_2 \leq \ldots \leq \ell_m$.

Kraft inequality:Proof outline

Kraft inequality: Any binary instantaneous code with lengths $\ell_1, \ell_2, \ldots, \ell_m$ should satisfy: $\sum_{i=1}^{m} 2^{-\ell_i} \leq 1$.

- Without loss of generality, suppose $\ell_1 \leq \ell_2 \leq \ldots \leq \ell_m$.
- Binary tree: Branches represent codewords

Kraft inequality:Proof outline

Kraft inequality: Any binary instantaneous code with lengths $\ell_1, \ell_2, \ldots, \ell_m$ should satisfy: $\sum_{i=1}^{m} 2^{-\ell_i} \leq 1$.

- Without loss of generality, suppose $\ell_1 \leq \ell_2 \leq \ldots \leq \ell_m$.
- Binary tree: Branches represent codewords
- Prefix condition: Each codeword eliminates its descendants as possible codewords.

Kraft inequality:Proof outline

Kraft inequality: Any binary instantaneous code with lengths $\ell_1, \ell_2, \ldots, \ell_m$ should satisfy: $\sum_{i=1}^{m} 2^{-\ell_i} \leq 1$.

- Without loss of generality, suppose $\ell_1 \leq \ell_2 \leq \ldots \leq \ell_m$.
- Binary tree: Branches represent codewords
- Prefix condition: Each codeword eliminates its descendants as possible codewords.
- Focus on the branches at level ℓ_m:
  1. Some of them are codewords
  2. Some are descendants of codewords
  3. Some are neither

Kraft inequality:Proof outline

Kraft inequality: Any binary instantaneous code with lengths $\ell_1, \ell_2, \ldots, \ell_m$ should satisfy: $\sum_{i=1}^{m} 2^{-\ell_i} \leq 1$.

- Without loss of generality, suppose $\ell_1 \leq \ell_2 \leq \ldots \leq \ell_m$.
- Binary tree: Branches represent codewords
- Prefix condition: Each codeword eliminates its descendants as possible codewords.
- Focus on the branches at level ℓ_m:
  1. Some of them are codewords
  2. Some are descendants of codewords
  3. Some are neither
- A codeword at level ℓ_i has $2^{\ell_m - \ell_i}$ descendants at level ℓ_m.

Kraft inequality:Proof outline

Kraft inequality: Any binary instantaneous code with lengths $\ell_1, \ell_2, \ldots, \ell_m$ should satisfy: $\sum_{i=1}^{m} 2^{-\ell_i} \leq 1$.

- Without loss of generality, suppose $\ell_1 \leq \ell_2 \leq \ldots \leq \ell_m$.
- Binary tree: Branches represent codewords
- Prefix condition: Each codeword eliminates its descendants as possible codewords.
- Focus on the branches at level ℓ_m:
  1. Some of them are codewords
  2. Some are descendants of codewords
  3. Some are neither
- A codeword at level ℓ_i has $2^{\ell_m - \ell_i}$ descendants at level ℓ_m.

$$\sum_{i=1}^{m} 2^{\ell_m - \ell_i} \leq 2^{\ell_m}$$

For Prefix codes, $L(C) \geq H(X)$

Theorem

- Show that: The average length of any prefix-free source code is lower bounded by entropy of the source, i.e.,

$$L(C) \geq H(X).$$

- Proof: In Class

Source Coding Theorem (SCT)
(Formal statement)

Huffman codes with $L(C)$ approaching $H(X)$

Huffman code: Example

- A zero-memory source emits messages m_1 and m_2 with probabilities 0.8 and 0.2 respectively.
  - Find binary Huffman code.
  - Find Huffman code for its second and third order extensions.
  - Determine code efficiency in each case.
- Solution: In class

Huffman code: Solution of Example

- Huffman code for $n = 1$ will be 0 and 1.
  - $L(C) = 1$ and $H(X) = -(0.8 \log_2(0.8) + 0.2 \log_2(0.2)) = 0.72$ bits
  - $\eta = H(X)/L(C) = 0.72$

- Huffman code for $n = 2$:

$m_1\ m_1$	0.64	0
$m_1\ m_2$	0.16	11
$m_2\ m_1$	0.16	100
$m_2\ m_2$	0.04	101

  - $L_1(C) = 1.56$. But this word length for two messages of the original source.
  - Word length per message will be $L(C) = 1.56/2 = 0.78$
  - $\eta = 0.72/0.78 = 0.923$

Huffman code: Solution of Example

- Huffman code for $n = 3$:

$m_1\ m_1\ m_1$	0.512	0
$m_1\ m_1\ m_2$	0.128	100
$m_1\ m_2\ m_1$	0.128	101
$m_2\ m_1\ m_1$	0.128	110
$m_1\ m_2\ m_2$	0.032	11100
$m_2\ m_1\ m_2$	0.032	11101
$m_2\ m_2\ m_1$	0.032	11110
$m_2\ m_2\ m_2$	0.008	11111

- ◦ $L_3(C) = 2.184$. But this word length for three messages of the original source.
- ◦ Word length per message will be $L(C) = 2.184/3 = 0.728$
- ◦ $\eta = 0.72/0.728 = 0.989$

Observations from Example

- Summary:
  - For $n = 1$, $\eta = 0.72$
  - For $n = 2$, $\eta = 0.923$
  - For $n = 3$, $\eta = 0.989$
- Thus, as the block length increases, the coding efficiency improves and approaches to 1.
- As we use the Huffman coding algorithm over longer and longer blocks of symbols, the average number of bits required to encode each symbol approaches the entropy of the source.
- We will next see why is it so.

Source coding theorem

- Notation: X, X^n, $\mathcal{X}, \mathcal{X}^n$, $H(X)$

Source coding theorem

- Notation: X, X^n, $\mathcal{X}, \mathcal{X}^n$, $H(X)$
- Encoding: For $x^n \in \mathcal{X}^n$ codeword is $f^n(x^n)$

Source coding theorem

- Notation: X, X^n, $\mathcal{X}, \mathcal{X}^n$, $H(X)$
- Encoding: For $x^n \in \mathcal{X}^n$ codeword is $f^n(x^n)$
- Decoding: Codeword $f^n(x^n)$ is decoded as $g^n(f^n(x^n)) := \hat{x}^n$

Source coding theorem

- Notation: X, X^n, $\mathcal{X}, \mathcal{X}^n$, $H(X)$
- Encoding: For $x^n \in \mathcal{X}^n$ codeword is $f^n(x^n)$
- Decoding: Codeword $f^n(x^n)$ is decoded as $g^n(f^n(x^n)) := \hat{x}^n$
- Decoding error: $\hat{x}^n \neq x^n$

Source coding theorem

- Notation: X, X^n, $\mathcal{X}, \mathcal{X}^n$, $H(X)$
- Encoding: For $x^n \in \mathcal{X}^n$ codeword is $f^n(x^n)$
- Decoding: Codeword $f^n(x^n)$ is decoded as $g^n(f^n(x^n)) := \hat{x}^n$
- Decoding error: $\hat{x}^n \neq x^n$
- Statement of source coding theorem:

Source coding theorem

- Notation: X, X^n, $\mathcal{X}, \mathcal{X}^n$, $H(X)$
- Encoding: For $x^n \in \mathcal{X}^n$ codeword is $f^n(x^n)$
- Decoding: Codeword $f^n(x^n)$ is decoded as $g^n(f^n(x^n)) := \hat{x}^n$
- Decoding error: $\hat{x}^n \neq x^n$
- Statement of source coding theorem:
  - Achievability: If average length $L(C) > H(X)$, then there exists a sequence of codes $\{f^{(n)}, g^{(n)}\}$ of average length $L(C)$ such that

Source coding theorem

- Notation: X, X^n, $\mathcal{X}, \mathcal{X}^n$, $H(X)$
- Encoding: For $x^n \in \mathcal{X}^n$ codeword is $f^n(x^n)$
- Decoding: Codeword $f^n(x^n)$ is decoded as $g^n(f^n(x^n)) := \hat{x}^n$
- Decoding error: $\hat{x}^n \neq x^n$
- Statement of source coding theorem:
  - Achievability: If average length $L(C) > H(X)$, then there exists a sequence of codes $\{f^{(n)}, g^{(n)}\}$ of average length $L(C)$ such that

$$P_e^{(n)} := \mathbb{P}\left[\hat{X}^n \neq X^n\right] \to 0 \quad \text{as} \quad n \to \infty$$

Source coding theorem

- Notation: X, X^n, $\mathcal{X}, \mathcal{X}^n$, $H(X)$
- Encoding: For $x^n \in \mathcal{X}^n$ codeword is $f^n(x^n)$
- Decoding: Codeword $f^n(x^n)$ is decoded as $g^n(f^n(x^n)) := \hat{x}^n$
- Decoding error: $\hat{x}^n \neq x^n$
- Statement of source coding theorem:
  - Achievability: If average length $L(C) > H(X)$, then there exists a sequence of codes $\{f^{(n)}, g^{(n)}\}$ of average length $L(C)$ such that

$$P_e^{(n)} := \mathbb{P}\left[\hat{X}^n \neq X^n\right] \to 0 \quad \text{as} \quad n \to \infty$$

  - Converse:

    If $L(C) < H(X)$, then $P_e^{(n)} > 0$ for any n.

Understanding source coding theorem

- Source coding theorem establishes a fundamental limit on the rate at which the output of an information source can be compressed without causing large error probability at the receiver.

Understanding source coding theorem

- Source coding theorem establishes a fundamental limit on the rate at which the output of an information source can be compressed without causing large error probability at the receiver.
- This is one of the fundamental theorems of information theory.

Understanding source coding theorem

- Source coding theorem establishes a fundamental limit on the rate at which the output of an information source can be compressed without causing large error probability at the receiver.
- This is one of the fundamental theorems of information theory.
- It states that, a source with entropy rate H can be encoded with arbitrarily small error probability at any rate R (bits/source output) provided, $R > H$.

Understanding source coding theorem

- Source coding theorem establishes a fundamental limit on the rate at which the output of an information source can be compressed without causing large error probability at the receiver.
- This is one of the fundamental theorems of information theory.
- It states that, a source with entropy rate H can be encoded with arbitrarily small error probability at any rate R (bits/source output) provided, $R > H$.
- If $R < H$, the error probability will be bounded away from zero, independent of the complexity of the encoder and decoder employed.