Data Compression With and Without Deep Probabilistic Models

Lecture 2 (28 April 2022)

Recap from last lecture:

- source coding vs channel coding
- source-channel separation

- symbol codes:

"alphabet" (finite or countably infinite set)

↳ message $\underline{x} = (x_1, x_2, \ldots, x_k)$ with $x_i \in \mathfrak{X} \quad \forall i \in \{1, \ldots, k\}$

↳ code book $C: \mathfrak{X} \to \{0, 1, \ldots, B-1\}^*$ (usually: base $B=2$)

↳ code $C^*: \mathfrak{X}^* \to \{0, 1, \ldots, B-1\}^*$ with $C^*(\underline{x}) := C(x_1) \| C(x_2) \| \ldots \| C(x_k)$

concatenation

Recap from tutorial:

probabilistic model of the data source

length of code word $C(x)$ (in bits)

- Def. "expected code word length": $L_C := \mathbb{E}_p[\ell_C(x)] = \sum_{x \in \mathfrak{X}} p(x)\, \ell_C(x)$

- Def. "prefix free symbol code C" (or "prefix code" for short):
  no code word C(x) is the prefix of another code word C(x')

- Def. "uniquely decodable symbol code C*: C* is injective

- prefix free ⇒ unique decodability; but inverse is not necessarily true

- Huffman coding: algorithm that takes a probabilistic model p (on a finite alphabet) and generates a prefix code that is "tailored" for this probabilistic model.

Today: Source Coding Theorem

Two fundamental truths about lossless compression ("good news and bad news"):

- bad news: Consider a data source that produces symbols with probability distribution p. Then, there is a fundamental lower bound H[p], and no uniquely decodable compression code can reach an expected code word length L that is lower than H[p].

$$H_B[p] \leq L_C \;\left(= \mathbb{E}_p[\ell_C(x)]\right) \qquad \forall \text{ uniquely decodable codes } C$$

- good news: For every data source, there exists a prefix-free (and thus uniqueley decodable) code (the so-called "Shannon code") whose expected code word length approaches the fundamental lower bound H[p] with an overhead of less than 1 bit per symbol.

$$H_B[p] \leq L_{C_{\text{Shannon}}} < H_B[p] + 1$$

- bonus: For finite alphabets, the Huffman coding algorithm always produces an optimal symbol code (i.e., a symbol code with lowest possible expected code word length L)

$$H_B[p] \leq L_{C_{\text{Huff}}} \leq L_{C'} \quad \forall \text{ uniqu. dec. } C' \;\Rightarrow\; H_B[p] \leq L_{C_{\text{Huff}}} \leq L_{C_{\text{Shannon}}} < H[p] + 1$$

Kraft-McMillan Theorem

(a) $\forall$ B-ary uniquely decodable symbol codes:

$$\sum_{x\in\mathcal{X}} \frac{1}{B^{\ell_C(x)}} \leq 1 \quad (\text{where } \ell_C = |C(x)|) \qquad \text{"Kraft inequality"}$$

(Interpretation: we can't make code words arbitrarily short. If we shorten one code word by one bit, then we may have to make some other code word(s) longer or else our code can no longer be uniquely decodable)

(b) $\forall$ functions $\ell : \mathcal{X} \to \mathbb{N}_0$ that satisfy Kraft ineq.

$\exists$ B-ary prefix code C with $|C(x)| = \ell(x)$ $\forall x \in \mathcal{X}$

"length of $C(x)$"

Corollary: $\forall$ uniq. dec. sym. codes C:

$\exists$ prefix code C' with $|C'(x)| = |C(x)|$ $\forall x \in \mathcal{X}$

⇒ When searching for an optimal symbol code, it suffices to consider only prefix codes.
(Actually, we don't really have to search directly for an optimal symbol code. It suffices to search for an optimal assignment of code word lengths l(x) that satisfy the Kraft inequality. Once we have that, we can construct a prefix code with these code word lengths, see below.)

Proof of the Kraft-McMillan Theorem

Lemma: Let $s \in \mathbb{N}_0$, C uniq. dec. symbol code,

$Y_s := \{ \underline{x} \in \mathcal{X}^* \text{ with } |C^*(\underline{x})| = s \}$ "length of"

Then: $|Y_s| \leq B^s$

(Proof: • $\exists\, B^s$ distinct bit strings of length s
⇒ if $|Y_s| > B^s$ then $\exists\, \underline{x}, \underline{x}' \in \mathcal{X}^*$ with $\underline{x} \neq \underline{x}'$ but $C^*(\underline{x}) = C^*(\underline{x}')$
→ not possible because C is uniq. dec., i.e., $C^* : \mathcal{X}^* \to \{0, \ldots, B-1\}^*$ is injective)

Proof of part (a):

Claim:

$$\boxed{\forall \text{ B-ary uniquely decodable symbol codes: } \quad r := \sum_{x\in\mathcal{X}} \frac{1}{B^{\ell_C(x)}} \leq 1 \quad (\text{where } \ell_C = |C(x)|)}$$

• Let $k \in \mathbb{N}$

• $r^k = \Big(\sum_{x\in\mathcal{X}} B^{-\ell_C(x)}\Big)^k$

$$= \underbrace{\Big(\sum_{x_1\in\mathcal{X}} B^{-\ell_C(x_1)}\Big)\Big(\sum_{x_2\in\mathcal{X}} B^{-\ell_C(x_2)}\Big) \cdots \Big(\sum_{x_k\in\mathcal{X}} B^{-\ell_C(x_k)}\Big)}_{k \text{ factors}}$$

$$= \sum_{\substack{x_1\in\mathcal{X}, x_2\in\mathcal{X},\\ \ldots, x_k\in\mathcal{X}}} B^{-\ell_C(x_1)} B^{-\ell_C(x_2)} \cdots B^{-\ell_C(x_k)} = \sum_{\underline{x}\in\mathcal{X}^k} B^{-\sum_{i=1}^{k} \ell_C(x_i)}$$

(i) assume (for now) that $\mathcal{X}$ is finite. $\Rightarrow \exists\, l_{max} < \infty$ s.t. $l_c(x) \le l_{max}$ $\forall x \in \mathcal{X}$

$$r^k = \sum_{\underline{x}\in\mathcal{X}^k} B^{-\underbrace{\sum_{i=1}^{k} l_c(x_i)}_{=:s}} = \sum_{s=0}^{k\, l_{max}} \underbrace{\underbrace{|Y_s|}_{\le B^s \text{ (lemma)}} B^{-s}}_{\le 1} \le k\, l_{max} + 1$$

$$\Rightarrow r^k \le k\, l_{max} + 1 \quad \forall k \in \mathbb{N}$$

$$\Rightarrow \underbrace{\frac{r^k - 1}{k}}_{\substack{\to\infty \\ \text{if } r>1}} \le \underbrace{l_{max}}_{\substack{\text{const (indep. of } k) \\ <\infty}} \quad \forall k \in \mathbb{N} \qquad \Rightarrow \boxed{r \le 1}$$

(ii) if $\mathcal{X}$ is countably infinite: with restriction, assume that $\mathcal{X} = \mathbb{N}$

$$\text{Then } r = \sum_{x\in\mathcal{X}} \frac{1}{B^{l_c(x)}} \underset{\substack{\uparrow \\ \text{all terms are } \ge 0}}{=} \sum_{x=1}^{\infty} \frac{1}{B^{l_c(x)}} = \lim_{n\to\infty} \underbrace{\sum_{x=1}^{n} \frac{1}{B^{l_c(x)}}}_{\substack{\text{finite alphabet} \\ \{1,\dots,n\} \Rightarrow \text{(i) applies}}} \le 1$$

Proof of part (b) of the Kraft-McMillan Theorem:

Constructive proof, i.e., we show existence of such a prefix code by providing an explicit algorithm that constructs it for any l.

Claim:

$$\boxed{\begin{array}{l} \forall \text{ functions } l: \mathcal{X} \to \mathbb{N}_0 \text{ that satisfy Kraft ineq.} \\ \exists\ B\text{-ary prefix code } C \text{ with } \underbrace{|C(x)|}_{\text{"length of } C(x)\text{"}} = l(x) \quad \forall x \in \mathcal{X} \end{array}}$$

Algorithm (*):

- **Input:** function $l: \mathcal{X} \to \{0,\dots,B-1\}^*$ that satisfies Kraft ineq.: $\sum_{x\in\mathcal{X}} B^{-l(x)} \le 1$
- **Output:** prefix code $C: \mathcal{X} \to \{0,\dots,B-1\}^*$ with $|C(x)| = l(x)$ $\forall x \in \mathcal{X}$
- **Steps:**
  - sort symbols in $\mathcal{X} = \{x, x', x'', \dots\}$ s.t. $l(x) \ge l(x') \ge l(x'') \ge \dots$
  - initialize $\xi = 1$
  - for each $x \in \mathcal{X}$ in above order
    - ↳ update $\xi \leftarrow \xi - B^{-l(x)}$ ($\Rightarrow$ claim $\xi \in [0,1)$)
    - ↳ write $\xi = (0.\underbrace{???}_{l(x)}\dots)_B$
    - ↳ set $C(x)$ to the first $l(x)$ bits after "0." (pad with trailing zeros if necessary)

Claim: The resulting code book C is prefix free. (Proof: Problem Set 2)

Example: Simplified game of Monopoly $(B=2)$

x	$p(x)$	$-\log_2 p(x)$	$\ell(x)$ Shann.	$C(x)$ Shann.	$\ell'(x)$	$C'(x)$
2	$1/9$	3.17 →	(1) 4	1111	3	111
3	$2/9$	2.17 →	(3) 3	110	2	10
4	$3/9 = 1/3$	1.58 →	(5) 2	01	2	01
5	$2/9$	2.17 →	(4) 3	101	2	00
6	$1/9$	3.17 →	(2) 4	1110	3	110
	$H_2[p] \approx 2.20$ bits		$L_C = \frac{26}{9} \approx 2.89$		$L_{C'} = \frac{20}{9} \approx 2.22$	

$\xi = 1$

$(1.0000)_2$
$-(0.0001)_2$, $\ell = 4$

- $x=2$: $\xi \leftarrow 1 - 2^{-4} = (0.1111)_2$
- $x=6$: $\xi \leftarrow (0.1111)_2 - 2^{-4} = (0.1110)_2$
- $x=3$: $\xi \leftarrow (0.111)_2 - 2^{-3} = (0.110)_2$
- $x=5$: $\xi \leftarrow (0.110)_2 - 2^{-3} = (0.101)_2$
- $x=4$: $\xi \leftarrow (0.101)_2 - 2^{-2} = (0.011)_2$

Check that Kraft inequality holds for ℓ:

$$r = \sum_{x \in \mathfrak{X}} 2^{-\ell(x)} = \dots = \frac{5}{8} \leq 1 \quad ✓$$

Questions: (1) Can we efficiently find the optimal code word lengths l(x) that satisfy the Kraft McMillan inequalit and that lead to the lowest expected code word length L?

→ Huffman Coding

(2) Can we estimate the optimal expected code word length L without having to find the whole table of optimal code word lengths l(x)?

→ Optimization problem: minimize $L := \sum_{x \in \mathfrak{X}} p(x)\ell(x)$ for some fixed p over all $\ell : \mathfrak{X} \to \mathbb{N}$ that satisfy Kraft ineq.

→ Spoiler: we'll find $H_B[p] \leq L_{opt} < H_B[p] + 1$

entropy $H_B[p] = \mathbb{E}_p[-\log_B p(x)] = -\sum_{x \in \mathfrak{X}} p(x) \log_B p(x)$

To address question (2), we use the following strategy:
(i) We derive a lower bound on L.
(ii) We show that there exists a valid assignment of code word lengths that approaches the lower bound with less than one bit of overhead.

(i) • relaxed opt. problem: minimize $L = \sum_{x \in \mathfrak{X}} p(x)\ell(x)$ over all positive real valued fcts $\ell : \mathfrak{X} \to \mathbb{R}_{\geq 0}$ that satisfy

$$r := \sum_{x \in \mathfrak{X}} B^{-\ell(x)} \leq 1$$

⇒ treat $\ell(x)\ \forall x \in \mathfrak{X}$ as indep. variables & enforce constraint with Lagrange mult.

⇒ find stationary point

$$A := \sum_{x' \in \mathfrak{X}} p(x')\ell(x') + \lambda\left(\sum_{x' \in \mathfrak{X}} B^{-\ell(x')} - 1\right)$$

$$B^{-\ell(x')} = \exp\left(\ln\left(B^{-\ell(x')}\right)\right) = \exp\left(-\ell(x') \cdot \ln B\right)$$

$$\forall x: \quad 0 = \left.\frac{\partial A}{\partial(\ell(x))}\right|_{\ell^*} = p(x) - (\lambda \ln B)\, B^{-\ell^*(x')}$$

$$\Rightarrow \ell^*(x) = -\log_B p(x) + \alpha \qquad \text{with constant } \alpha = \log_B(\lambda \ln B)$$

$$0 = \left.\frac{\partial A}{\partial \lambda}\right|_{\ell^*} = \sum_{x\in\mathcal{X}} B^{-\ell^*(x)} - 1 = B^{-\alpha} \underbrace{\sum_{x\in\mathcal{X}} p(x)}_{=1} - 1 = \underbrace{B^{-\alpha} - 1}_{=0} \quad \Rightarrow \alpha = 0$$

$$\Rightarrow \ell^*(x) = -\log_B p(x)$$

"Information content of symbol x under the model p"

⇒ l* minimizes the expected code word length for under the relaxed constraint. Thus, for any other $\ell : \mathcal{X} \to \mathbb{R}_{\geq 0}$ that satisfies the Kraft inequality, we have:

$$\sum_{x\in\mathcal{X}} p(x)\,\ell(x) \geq \sum_{x\in\mathcal{X}} p(x)\,\ell^*(x)$$

in particular, it holds for integer valued ℓ that satisfy Kraft inequ

$\Rightarrow \forall$ uniquely dec. symbol codes C:

$$\underbrace{\sum_{x\in\mathcal{X}} p(x)\,\ell_C(x)}_{L_C} \geq \underbrace{-\sum_{x\in\mathcal{X}} p(x) \log_B p(x) = \mathbb{E}_p[-\log_B p(x)]}_{H_B[p]}$$

$$L_C \geq H_B[p]$$

entropy

(ii) How closely can we approach this lower bound (taking into account that l(x) must be integer)?
→ Answer: within an overhead of less than 1 bit per symbol.

Proof: choose $\ell(x) := \lceil \ell^*(x) \rceil = \lceil -\log_B p(x) \rceil \geq \ell^*(x) \quad \forall x \in \mathcal{X}$

$$\Rightarrow \sum_{x\in\mathcal{X}} B^{-\ell(x)} \leq \sum_{x\in\mathcal{X}} B^{-\ell^*(x)} = 1 \quad \rightarrow \text{satisfies Kraft ineq.}$$

$$\Rightarrow L_{\text{Shannon}} = \mathbb{E}_p\left[\lceil -\log_B p(x) \rceil\right] < \mathbb{E}_p\left[-\log_B p(x) + 1\right]$$

$$\boxed{\Rightarrow H_B[p] \leq L_{\text{Shannon}} < H_B[p] + 1}$$

"source coding theorem"

Note: If we use these code word lengths l(x) and apply Algorithm (*), then the resulting prefix code C is called the "Shannon Code for the probability distribution p".

→ see code C in the "Simplified Game of Monopoly" example above; more examples on Problem Set 2.