CODAGE de SOURCE UNIVERSEL

rappel :

Th. Si $(X_n)_{n\geq 1}$ source sans mémoire (iid) de loi p d'alphabet fini $\mathcal{X}$,

alors $\exists$ un code préfixe $C : \mathcal{X} \to \{0,1\}^*$ et de longueur $L : \{0,1\}^* \to \mathbb{N}$ tq

$$H(X) \leq \mathbb{E}\left[L(C(X))\right] < H(X) + 1$$

Exercice Wrong code. Supposons que l'on connaisse pas la vraie loi p ; mais seult q.
Supposons que l'on choisisse un code C_q dont la longueur vérifie $L(C_q^*(x)) = \left\lceil \log \frac{1}{q(x)} \right\rceil$. Notons $H(p) = H(X)$ (calculée avec vrai loi de X)

Mq $$H(p) + \mathcal{D}(p\|q) \leq \mathbb{E}\left[L(C_q(X))\right] < H(p) + \mathcal{D}(p\|q) + 1$$

Preuve : $\mathbb{E}\left[L(C_q(X))\right] = \sum_{x\in\mathcal{X}} p(x) \left\lceil \log \frac{1}{q(x)} \right\rceil < \sum p(x) \left(\log \frac{p(x)}{q(x)} \cdot \frac{1}{p(x)} + 1 \right)$

$$= \mathcal{D}(p\|q) + H(p) + 1$$

□

. dessin pour la preuve de la borne inf

Donc croire que la vraie loi est $q(x)$ alors qu'en fait elle est $p(x)$ génère 1 accroissement de $\mathcal{D}(p\|q)$ de la longueur moyenne du code.

Question : quel accroissement de la longueur moyenne du code si l'on ne connait pas la loi ?

Objectif : si p inconnue, trouver un code qui rende la longueur de description du message la + petite possible ($\forall$ lois processus … dans 1 famille)

Notation

X processus iid d'alphabet $\mathcal{X} = \{a_1, a_j, a_{|\mathcal{X}|}\}$ (indice de l'alphabet)

p loi de X

x^n 1 réalisation de X de longueur n = message à coder
$= (x_1 \dots x_i, x_n)$ (indice du vecteur)

q_{x^n} ou q = type de x^n i.e loi empirique de x^n

↖ si pas d'ambiguité, on supprimera x^n pour ne pas surcharger les notations.

$\mathcal{Q}_n$ = ens des types de vecteurs de longueur n.

$\tilde{p}$ = une loi arbitraire

$T(\tilde{p})$ = classe de type $\tilde{p}$ = ens des vecteurs de longueur n et de type $\tilde{p}$

$= \{x^n : q_{x^n} = \tilde{p}\}$.

$$H(\tilde{p}) = -\sum_{a \in \mathcal{X}} q(a) \log_2 q(a)$$

Résumé :

- $|\mathcal{Q}_n| \leq (n+1)^{|\mathcal{X}|}$

Méthode des types

Stratégie 2 étapes

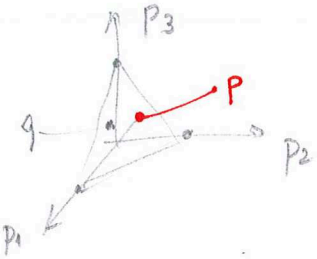
1) estimer la loi à partir de x^n ; et transmettre cette estimée
2) coder x^n selon cette estimée

Déf type du vecteur x^n est sa loi empirique

$$\forall j \in [\![1, |\mathcal{X}|]\!], \quad q_{x^n}(a_j) = \frac{|\{i : x_i = a_j\}|}{n} = \frac{n_j}{n} \equiv q(a_j)$$

Rappel le simplexe est $\left\{ (p_1, \dots p_j, \dots p_{|\mathcal{X}|}) : \forall j \in [\![1, |\mathcal{X}|]\!] \ p_j \geq 0 \text{ et } \sum_{j=1}^{|\mathcal{X}|} p_j = 1 \right\}$

Ex $|\mathcal{X}| = 3$



Soit $\mathcal{Q}_n$ = ens des types de vecteurs de taille n

(CT Th 11.1.1) Montrer que

a) $|\mathcal{Q}_n| \leq (n+1)^{|\mathcal{X}|}$

b) $|\mathcal{Q}_n| \leq (n+1)^{|\mathcal{X}|-1}$

c) $|\mathcal{Q}_n^*| \doteq \binom{n + |\mathcal{X}| - 1}{|\mathcal{X}| - 1} \approx n^{|\mathcal{X}|-1}$

indice : • $|\mathcal{X}|$ candidats, n votants ; $|\mathcal{Q}_n|$ = nbre de résultats de vote différents (c'est le pb.)

Pour le résoudre : considérer le pb.

$|\mathcal{X}|$ boules de couleurs $\neq$ ($\overset{R}{a_1} \dots \overset{V}{a_2} \dots \overset{B}{a_{|\mathcal{X}|}}$)

n tirages avec remise (car réserve de boules ∞)

résultats de tirages $\neq$.

Pour la suite a) est suffisant.

a) $n q_{x^n} = (n_1, \dots n_j, \dots n_{|x|})$ est 1 vecteur de taille $|x|$

avec $0 \le n_j \le n$, i.e chaque élément du vecteur peut prendre $n+1$ valeurs
(c'est seulement borne sup, car on n'a pas pris en cpte les contraintes) □

b) On sait que $\sum_{j=1}^{|x|} n_j = n$; Donc $n_{|x|}$ est déterminé par les autres valeurs $n_j, j \le |x|$

donc il y a au plus $(n+1)^{|x|-1}$ types possibles □

c) $n_1 =$ # fois où boule de couleur 1 est portée

n_2 : " " 2 "

$n_{|x|}$: " " $|x|$ "

$n = \sum n_j$ = nbre de tirages

Il y a correspondance entre :

- un choix de $n_1 n_2 \dots n_{|x|}$, $n_j \ge 0$ et $\sum n_j = n$
- une manière de placer $|x|-1$ boules séparatrices parmi $n+|x|-1$

nbre de choix possibles ≡ nbre de manières de placer les boules séparatrices □

Par formule de Stirling : $n! \sim \sqrt{2\pi n}\left(\frac{n}{e}\right)^n$ $n \to +\infty$

$$\log \binom{n+|x|-1}{|x|-1} \approx (|x|-1)\log(n+|x|-1) + \text{cste}$$
$$\approx (|x|-1)\log n \qquad n \to +\infty$$

11.1.2 (CT)

Th. Soit X^n processus i.i.d de loi $p(x)$. La probabilité de x^n dépend uniquement de son type et satisfait

$$p(x^n) : \boxed{\mathbb{P}(X^n = x^n) = 2^{n(-D(q_{x^n}\|p) - H(q_{x^n}))}}$$

Corollaire Si le type de x^n est p, alors

$$\boxed{\mathbb{P}(X^n = x^n) = 2^{-nH(p)}}$$

Preuve

$$\mathbb{P}(X^n = x^n) = \prod_{i=1}^{n} p(x_i) = \prod_{a \in \mathcal{X}} p(a)^{\overbrace{nq(a)}^{\#\text{occurences de } a \text{ dans } x^n}_{x^n}} = \prod_{a \in \mathcal{X}} 2^{nq(a)\log p(a)}$$

$$= \prod_{a \in \mathcal{X}} 2^{n(-q(a)\log q(a) + q(a)\log p(a) + q(a)\log q(a))}$$

$$= 2^{n \sum_{a \in \mathcal{X}} \left(-q(a)\log q(a) + q(a)\log p(a) + q(a)\log q(a)\right)}$$

$$= 2^{n(-D(q\|p) - H(q))}$$

□.

Th 11.1.3 (CT). Taille de la classe de type q. Quelle que soit le type q de la séquence x^n.

$$\boxed{|T(q)| \leq 2^{nH(q)}}$$

Preuve : $|T(q)|$ = nbre de manières d'arranger $nq(a_1)$ éléments a_1, $nq(a_2)$ éléments a_2, ... de 1 vecteur de taille n ($\mathcal{X} = \{a_1, \dots, a_{|\mathcal{X}|}\}$)

$$= \text{coefficient multinomial} = \binom{n}{nq(a_1), nq(a_2), \dots, nq(a_{|\mathcal{X}|})}$$

difficile à manipuler, d'où calcul de borne sup.

$$1 \geq \sum_{x^n \in T(q)} q(x^n) = \sum_{x^n} 2^{-nH(q)} = |T(q)|\, 2^{-nH(q)}$$

⚠ proba évaluée avec le type q. C'est différent de $\mathbb{P}(X^n = x^n)$ évalué avec vraie loi

Donc $|T(q)| \leq 2^{nH(q)}$

□.

Th 11.1.4 CT.
probabilité de la classe de type q, (selon la loi $\tilde{p}$) ~~Th. 11.1.4~~

* Soit un type $q \in \mathcal{Q}_n$. Soit 1 loi arbitraire $\tilde{p}$.
La probe de la classe de type q selon la loi $\tilde{p}$ satisfait

$$\tilde{p}(T(q)) \leq 2^{-n D(q\|\tilde{p})}$$

$$\tilde{p}(T(q)) = \sum_{x^n \in T(q)} \tilde{p}(x^n) = \sum_{x^n \in T(q)} 2^{-n(D(q\|\tilde{p}) + H(q))}$$

11.1.2

$$= |T(q)| \, 2^{-n(D(q\|\tilde{p}) + H(q))} \leq 2^{-n D(q\|\tilde{p})}$$

$|T(q)| \leq 2^{nH(q)}$ (11.1.3)

□

En particulier $p(T(q)) \leq 2^{-n D(q\|p)}$

Résumé

11.1.1 $|\mathcal{Q}_n| \leq (n+1)^{|\mathcal{X}|}$

11.1.2 $p(x^n) = 2^{-n(D(q\|p) + H(q))}$

11.1.3 $|T(q)| \leq 2^{nH(q)}$

11.1.4 $p(T(q)) \leq 2^{-n D(q\|p)}$

Définition ~~Soit~~ Un code à longueur fixe de ~~débit R~~ $(n, 2^{nR})$ pour le processus $X_1 \dots X_n$ de vraie distribution inconnue p, consiste en

fonction codage $f_n : \mathcal{X}^n \longrightarrow \{1, \dots, 2^{nR}\}$

— de décodage $g_n : \{1 \dots 2^{nR}\} \longrightarrow \mathcal{X}^n$

- R est appelé débit (il faut R bits pour coder 1 symbole X_i)
- la proba d'erreur pour ce code (calculée avec la vraie loi p) est

$$P_e^{(n)} = \underset{\text{évalué avec } p}{P_p}\left(x^n : g_n(f_n(x^n)) \neq x^n\right)$$

Définition Un code de débit R est dit universel si f_n et g_n ne dépendent pas de p et

si $P_e^{(n)} \xrightarrow[n \to +\infty]{} 0$

Théorème de codage universel Il existe une suite de codes universels de paramètres $(2^{nR}, n)$ tq pour toute source de loi $p(x)$ tq $\boxed{H(p) < R}$, alors $P_e^n \xrightarrow[n \to +\infty]{} 0$

interprétation : même taux de compression asymptotique qu'avec ou sans connaissance de la loi !!! ...

Preuve Soit R le débit de ce code. Soit

$$R_n = R - |\mathcal{X}| \frac{\log(n+1)}{n}$$

i.e. $f_n : \mathcal{X}^n \longrightarrow \{1 \dots 2^{nR}\}$

$x^n \longmapsto$ ⟷ identification index de type q

⟷ identification type q fixé

2^{nR}

taux $|\mathcal{X}| \log(n+1)$

longueur $= nR_n$

Soit $A = \{ x^n \in \mathcal{X}^n : H(q) \leq R_n \}$ ens des vecteurs dont type satisfait

$$|A| = \sum_{q \in \mathcal{Q}_n : H(q) \leq R_n} |T(q)| \leq \sum_{q \in \mathcal{Q}_n : H(q) \leq R_n} 2^{nH(q)} \leq \sum_{q \in \mathcal{Q}_n : H(q) \leq R_n} 2^{nR_n}$$

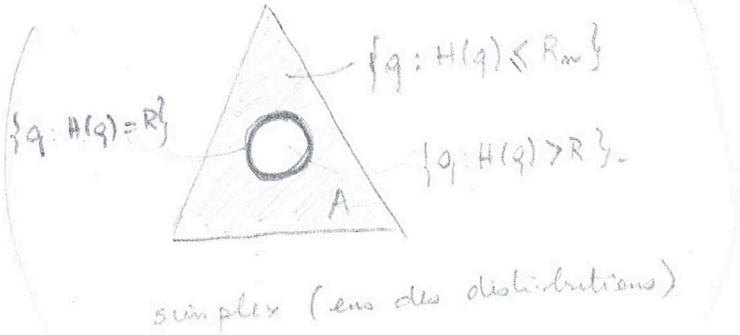
(11.1.3) ; ($H(q) \leq R_n$)

$$\leq |n+1|^{|\mathcal{X}|} \, 2^{nR_n} = 2^{n\left(R_n + |\mathcal{X}| \frac{\log(n+1)}{n}\right)} = 2^{nR}$$

(14.1.1) ; (def R)

codage $f_n(x^n) = \begin{cases} \text{indice de } x^n \text{ dans } A, & \text{si } x^n \in A \\ 0, & \text{si } x^n \notin A \end{cases}$

décodage tout indice est associé au vecteur x^n, si $x^n \in A$ ~~et est le~~
Tous les éléments de A sont reconstruits sans erreur
Tous les autres vecteurs ——— avec erreur



simplex (ens des distributions)

Rlq que ce codage est universel

$$P_e^{(n)} = 1 - \mathbb{P}_p(A) = \sum_{q \in \mathcal{Q}^n : H(q) > R_n} \mathbb{P}_p(T(q))$$

$$\leq (n+1)^{|\mathcal{X}|} \max_{q : H(q) > R_n} \mathbb{P}_p(T(q))$$

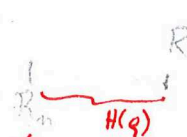
$$\leq (n+1)^{|\mathcal{X}|} \, 2^{-n \min_{q : H(q) > R_n} \mathbb{D}(q \| p)}$$

On . $R_n \xrightarrow[n \to +\infty]{} R$ et $\forall n \, R_n \leq R$ $(R_n \uparrow R)$

. par hypothèse $H(p) < R$

Donc $\exists n_0$ tq $\forall n \geq n_0$ $H(p) < R_n$

Donc $\min_{q : H(q) > R_n} \mathbb{D}(q \| p) > 0$ et $P_e^{(n)} \xrightarrow[n \to +\infty]{} 0$ décroissance exponentielle



□ 6.

CONCLUSION

- même taux de compression asymptotique avec/sans connaissance loi.

"normal" coder loi $\approx |\mathcal{X}| \log n$

coder vecteur $\approx H(p) \log n$

- à taille finie le coût supplémentaire est (admis)

$$\frac{\#\text{ degré de liberté loi}}{2} \log n$$

iid $\frac{|\mathcal{X}|-1}{2} \log n$

Markov ordre 1 $\frac{|\mathcal{X}|-1}{2} |\mathcal{X}| \log n$

- l'algorithme en 2 étapes est optimal.

- ce principe est utilisé en ML pour faire sélection modèle
  (model selection : MDL minimum description length)

Conclusion

ce principe du codage universel (CU) est utilisé en stats, statistique et s'appelle MDL =

(raffel) stat : à partir d'1 observation d'1 v.a. que dire sur la loi

[codage sans perte : message, trouver la description la + courte]

. 1 des principe MDL : choisir "la meilleure loi" est 1 loi qui compresse bien les données

. CU est aussi utilisé en ML :

Classification $x_1 \dots x_n$ n points à répartir en K classes (clusters) mais combien de classe ?

algo : init K centres (c_K random)

. iterate : - association à centre

$\forall k \quad \mathcal{C}_k = \{ x_i \text{ s.t. } |x_i - c_k| \leq |x_i - c_{k'}|, \forall k' \neq k \}$

- calcule centroïde

c_k = centre de $\mathcal{C}_k$

$c_k = \arg\min \sum_{x \in \mathcal{C}_k} \|x - c_k\|^2$

Choix K tq $\forall x_i$, $\tilde{x}_i$ codage (x_i) : $\lceil c_k \rceil \quad \lceil x_i - c_k \rceil$

$$\min_K \sum_{1 \leq i \leq n} L\left(\text{codage}_K(k)\right) + L\left(\text{codage}_K(x_i - c_k)\right)$$

$Q \to ?$
$m \to \infty$

7'

Cette technique est 1 parmi d'autres (pour les sources i.i.d)

LZ est 1 code à longueur variable pour les sources ergodiques, dont la caractérisation de la loi est complexe.

Algo Lempel Ziv 78-78 : repose sur analyse syntaxique (parsing) du message à coder.
CS 17

parsing d'1 chaîne de binaire (binary string) : est une segmentation de la chaîne en phrases séparées par des virgules

distinct parsing : parsing où toutes les phrases sont ≠

LZ parsing : distinct parsing où chaque phrase est la + petite phrase qui n'est pas déjà affiché

$x_1 x_2 \ldots x_n$: 0, 1, 01, 11, 00, 10, 000, 101, 001, 110, 1010,
étiquette 0 1 2 3 4 5 6 7 8 9 10 11

$c(x_1, \ldots, x_n)$ = # phrases dans le LZ parsing de la chaîne $x_1 \ldots x_n$

Algo pour coder $x_1 \ldots x_n$

1- faire LZ parsing
2- compter $c(x_1 \ldots x_n)$ et coder $c(x_1 \ldots x_n)$ avec u_n bits
3- étiqueter chaque phrase avec 1 mot de longueur u_n bits
4- La $k^{ième}$ phrase est de type $y_k = \begin{cases} y_{m_k}\,0 \\ y_{m_k}\,1 \end{cases}$ où m_k = étiquette de y_{m_k}

Coder cette $k^{ième}$ phrase (y_k) sous forme $\begin{cases} (m_k, 0) \\ (m_k, 1) \end{cases}$

nécessite $\log\big(c(x_1 \ldots x_n) + 1\big) + \begin{cases} 1 \\ 0 \text{ ou } 1 \end{cases}$
↑ explique gain ; ↑ si $y_{m_k} = \phi$

Ex $c(x_1 \ldots x_n) = 11$ → 4 bits
$c(\ldots) + 1 = 12$

$C_{LZ}(x_1 \ldots x_n) = \underbrace{0}_{0}\,\underbrace{000}_{0}\,1,\ \underbrace{000}_{1}\,1\,1,\ \underbrace{001}_{2}\,1,\ \underbrace{0001}_{1}\,0,\ \underbrace{0010}_{2}\,0,\ \underbrace{0101}_{5}\,0,$

En pratique :
- si mémoire finie, les phrases anciennes sont écrasées par les nouvelles
- variante célèbre de Storer Szymanski : on ne code que si codage est + court que la phrase d'origine en ajoutant 1 bit de flag
  0 si codage direct
  1 si codage avec position