Teoria dell'Informazione/RTI **Anno Accademico 2022–2023**

Lezione 4

Ugo Vaccaro

Nelle lezioni scorse abbiamo introdotto il seguente problema di codifica: data una sorgente discreta, stazionaria e senza memoria $X_1, X_2, \ldots, X_i, \ldots$, in cui $X_i = X$ per ogni i, con X avente distribuzione di probabilità $P = \{P(x) : x \in \mathsf{X}\}$, X= alfabeto sorgente, ed un intero k, vogliamo trovare una funzione di codifica $\mathsf{c} : \mathsf{X}^k \to \{0,1\}^n$, e decodifica $\mathsf{d} : \{0,1\}^n \to \mathsf{X}^k$, tale che la probabilità di errore $e(\mathsf{d},\mathsf{c}) = P\{\mathbf{x} \in \mathsf{X}^k : \mathsf{d}(\mathsf{c}(\mathbf{x})) \neq \mathbf{x}\}$ sia piccola (ovvero inferiore ad un ϵ precedentemente fissato), ed il rapporto n/k sia il più piccolo possibile. Ciò che abbiamo scoperto è che per il valore $n(k,\epsilon)$, da noi definito come più piccolo valore di n per cui esistono funzioni di codifica $\mathsf{c} : \mathsf{X}^k \to \{0,1\}^n$, e decodifica $\mathsf{d} : \{0,1\}^n \to \mathsf{X}^k$ con $e(\mathsf{d},\mathsf{c}) \leq \epsilon$, vale che

$$\lim_{k\to\infty} \frac{n(k,\epsilon)}{k} = H(P) = \sum_{x\in\mathsf{X}} P(x) \log \frac{1}{P(x)}.$$

Ci sono due problemi per il risultato provato:

1. Il metodo fornito nella scorsa lezione non è pratico dal punto di vista computazionale, in quanto richiede di determinare l'insieme $B(k,\delta)$. Il metodo proposto per il calcolo di $B(k,\delta)$ richiedeva di andare a vedere per ogni sequenza lunga k se essa soddisfaceva il test di appartenenza a $B(k,\delta)$. Ma ciò non è fattibile poichè il numero di sequenza di lunghezza k costruibili sull'alfabeto sorgente X è esponenziale nella lunghezza k.
2. La probabilità di errore è sì piccola a piacere ma comunque maggiore strettamente di 0.

In questa lezione vedremo come:

1. Codificare l'output della sorgente usando pochi bit con un metodo computazionalmente efficiente.
2. Ottenere probabilità di decodifica errata pari a 0.

Come prima cosa possiamo eliminare il vincolo che le codifiche $\mathsf{c} : \mathsf{X}^k \to \{0,1\}^n$ associno ad ogni sequenza sorgente una sequenza binaria composta da esattamente n bits. Ciò era motivato dal fatto che codifiche di lunghezza fissata rendono più semplice la decodifica (basta leggere n bits alla volta alla volta per decodificare).

Osserviamo inoltre che codifiche $\mathsf{c} : \mathsf{X}^k \to \{0,1\}^*$, dove $\{0,1\}^* = \cup_{n\geq 1}\{0,1\}^n$ presentano un problema, a causa della loro complessità di calcolo. Infatti, la cardinalità di $|\mathsf{X}^k|$ è pari a $|\mathsf{X}|^k$, che è esponenziale nella lunghezza k delle sequenze da codificare, per cui vi sarebbero grossi problemi pratici nella esplicitazione della funzione di codifica c. Pertanto, il desiderio di avere probabilità di decodifica errata pari a zero ci forza a dover fornire (e calcolare) un numero esponenziale di *differenti* sequenze binarie per *differenti* sequenze sorgenti in X^k (altrimenti si potrebbero verificare errori nella decodifica). Procediamo quindi nel seguente modo. Data una sorgente DSSM rappresentata dalla variabile casuale X, con alfabeto sorgente X e distribuzione di probabilità $P = \{P(x) : x \in \mathsf{X}\}$, definiamo innanzitutto una codifica (binaria) a lunghezza variabile per *singoli simboli* emettibili da X come una funzione $\mathsf{c} : \mathsf{X} \to \{0,1\}^*$, dove $\{0,1\}^* = \cup_{n\geq 1}\{0,1\}^n$. Per ogni $x \in \mathsf{X}$ la sequenza binaria $\mathsf{c}(x)$ associata a x sarà detta *parola codice* associata ad x, $\ell(x)$ denoterà la lunghezza di $\mathsf{c}(x)$ (ovvero il numero di 0 ed 1 che $\mathsf{c}(x)$ contiene), e l'insieme $C = \{\mathsf{c}(x) : x \in \mathsf{X}\}$ verrà detto il *codice* di X.

Esempio 1 *Supponiamo di avere una sorgente con alfabeto* $\mathsf{X} = \{a,b,c,d,e,f,g,h,i,\ell,m,n,o,p\}$. *Una possibile funzione di codifica* $\mathsf{c} : \mathsf{X} \to \{0,1\}^*$ *potrebbe essere la funzione che assegna le seguenti parole codice a lettere di* X*:* $\mathsf{c}(a) = 11, \mathsf{c}(b) = 011, \mathsf{c}(c) = 001, \mathsf{c}(d) = 01011, \mathsf{c}(e) = 101, \mathsf{c}(f) = 100, \mathsf{c}(g) = 0001, \mathsf{c}(h) = 01010, \mathsf{c}(i) = 01001, \mathsf{c}(\ell) = 0001, \mathsf{c}(m) = 000001, \mathsf{c}(n) = 0000001, \mathsf{c}(o) = 0000000, \mathsf{c}(p) = 01000$.

Per passare dalla codifica di simboli alla codifica di sequenze, definiamo la estensione di una codifica $\mathsf{c} : \mathsf{X} \to \{0,1\}^*$ di singoli simboli sorgente a sequenze sorgenti come una funzione $\mathsf{c}^* : \mathsf{X}^* \to \{0,1\}^*$, dove X^*=insieme di tutte le possibili sequenze (di qualsivoglia lunghezza) costruibili sull'alfabeto sorgente X, nel modo seguente

$$\forall n \quad \forall \mathbf{x} = x_1 x_2 \dots x_n \quad \mathsf{c}^*(x_1 \dots x_n) = \mathsf{c}(x_1)\mathsf{c}(x_2)\dots\mathsf{c}(x_n).$$

In altri termini, la codifica $\mathsf{c}^*(\mathbf{x})$ di $\mathbf{x} = x_1 x_2 \dots x_n$ è ottenuta concatenando una dopo l'altra le codifiche dei simboli individuali sorgente $x_1, x_2, \dots, x_n$, nell'ordine in cui essi appaiono nella sequenza $\mathbf{x} = x_1 x_2 \dots x_n$. Ovviamente, avremo che la lunghezza $\ell(\mathsf{c}^*(x_1 \dots x_n))$ della codifica $\mathsf{c}^*(x_1 \dots x_n)$ sarà pari a $\ell(\mathsf{c}(x_1)) + \ell(\mathsf{c}(x_2)) + \dots + \ell(\mathsf{c}(x_n))$.

Utilizzando questo approccio non dovremmo più specificare la codifica per ogni sequenza in X^k ma solo quella per i simboli in X.

Come misurare la "bontà" di una codifica a lunghezza variabile? Come al solito, vogliamo rappresentare ciò che la sorgente emette nella maniera più compatta possibile. Con l'approccio utilizzato nelle scorse lezioni, poichè utilizzavamo codifiche a lunghezza fissa potevamo misurare ciò semplicemente considerando il rapporto $\frac{n}{k}$, ma qui non possiamo procedere in tal modo, in quanto sequenze sorgenti di eguale lunghezza possono avere codifiche di lunghezza differente.

Ora, se il simbolo $x \in \mathsf{X}$ ha probabilità di essere emesso dalla sorgente pari a $P(x)$, sappiamo dalla legge dei grandi numeri che il numero di volte con cui x apparirà in una sequenza di k emissioni della sorgente (con k molto grande) è $\approx kP(x)$. Ogni volta che tale simbolo x sarà emesso useremo la parola codice $\mathsf{c}(x)$ ad esso associato e, quindi, utilizzeremo $\ell(x)$ bits. In totale, utilizzeremo $\approx k \times P(x)\ell(x)$ bits per codificare le $k \times P(x)$ occorrenze del simbolo x nella sequenza sorgente lunga k. Questo per il simbolo x. Sommando su tutti i simboli $x \in \mathsf{X}$ che la sorgente può emettere, otterremo che il numero totale di bits che utilizzeremo per rappresentare le k emissioni della sorgente sarà approssimativamente pari a $k \times \sum_{x \in \mathsf{X}} P(x)\ell(x)$, da cui segue che il numero di bits per simbolo sorgente che useremo sarà $\approx \sum_{x \in \mathsf{X}} P(x)\ell(x)$. La quantità $\sum_{x \in \mathsf{X}} P(x)\ell(x)$ è chiaramente pari alla lunghezza media della codifica c, e sarà quindi questo il parametro che cercheremo di minimizzare. Si chiama lunghezza media perchè corrisponde al valor medio della seguente variabile aleatoria che assume come valori le lunghezze delle parole con le probabilità con cui i simboli vengono emessi

$$\begin{pmatrix} \ell(x_1) & \ell(x_2) & \cdots & \ell(x_m) \\ P(x_1) & P(x_2) & \cdots & P(x_m) \end{pmatrix}.$$

Cerchiamo inoltre di capire quali proprietà deve soddisfare la codifica c per poter ammettere una funzione di decodifica d per cui $P\{\mathbf{x} \in X^k : \mathsf{d}(\mathsf{c}(\mathbf{x})) \neq \mathbf{x}\} = 0$. Ovviamente, richiediamo che

$$\forall x, x' \in \mathsf{X} \quad \text{deve valere che} \quad x \neq x' \Rightarrow \mathsf{c}(x) \neq \mathsf{c}(x'). \tag{1}$$

Purtroppo, ciò non basta. Immaginiamo una sorgente che abbia alfabeto $\mathsf{X} = \{a, b, c, d\}$, con codifica $\mathsf{c}(a) = 0, \mathsf{c}(b) = 010, \mathsf{c}(c) = 01, \mathsf{c}(d) = 10$. Se la sorgente emettesse ad, tale sequenza verrebbe codificata come $\mathsf{c}(a)\mathsf{c}(d) = 010$ che, guarda caso, è anche uguale a $\mathsf{c}(b)$. Di conseguenza, in fase di decodifica non sapremmo come decodificare la sequenza binaria 010, ovvero non sapremmo se essa corrisponde alla codifica della sequenza ad o alla codifica di b. Occorre quindi richiedere una proprietà più forte della (1). Per capire quale proprietà occorre richiedere, riguardiamo l'esempio di sopra per renderci conto che il problema sorge quando si codificano sequenze mediante la concatenazione (ovvero la scrittura di una appresso l'altra) di codifiche dei simboli individuali che compaiono nella sequenza.

Definizione 1 *Diremo che la codifica* $\mathsf{c} : \mathsf{X} \to \{0,1\}^*$ *è Unicamente Decifrabile (UD) se e solo se*

$$\forall n, m \quad \forall x_1 x_2 \dots x_n, y_1 y_2 \dots y_m \in \mathsf{X}^* \quad \textit{vale che} \quad x_1 x_2 \dots x_n \neq y_1 y_2 \dots y_m \Rightarrow \mathsf{c}^*(x_1 \dots x_n) \neq \mathsf{c}^*(y_1 \dots y_m).$$

É chiaro che la proprietà più forte che andavamo cercando è proprio quella espressa nella definizione appena data, in quanto permette, in linea di principio, di risalire dalle codifiche alle sequenze sorgenti originarie in quanto stiamo ora richiedendo che $\mathsf{c} : \mathsf{X} \to \{0,1\}^*$ sia iniettiva (e quindi invertibile).

Ritorniamo ora al nostro problema di partenza di trovare codifiche $\mathsf{c} : \mathsf{X} \to \{0,1\}^*$ che utilizzino "pochi" bit per simbolo sorgente. Abbiamo visto prima che se usiamo una codifica $\mathsf{c} : \mathsf{X} \to \{0,1\}^*$ per rappresentare le sequenze emesse da una sorgente con alfabeto X, utilizzeremo in media $\sum_{x\in\mathsf{X}} P(x)\ell(x)$ bits per ogni simbolo sorgente della sequenza. Vorremmo che tale quantità fosse la più piccola possibile. Abbiamo quindi lo stesso problema che abbiamo studiato nella lezione scorsa, solo che in tale lezione ogni sequenza sorgente di lunghezza k veniva codificata con sequenze binarie composte sempre da n bits, quindi il rapporto che ci interessava minimizzare era più semplicemente pari a n/k =numero di bits usati per simblo sorgente. Abbiamo, invece, in questa lezione il problema di stimare la quantità

$$\min \sum_{x\in\mathsf{X}} P(x)\ell(x) \tag{2}$$

dove il minimo è calcolato su *tutti* i possibili vettori $(\ell(x_1), \ldots, \ell(x_n))$ che corrispondono ad un vettore di lunghezze di una codifica UD per $\mathsf{X} = \{x_1, \ldots, x_n\}$. Come possiamo rappresentare matematicamente questo vincolo? Apriamo una parentesi per studiare questo fatto per enunciare (e dimostrare) la seguente importante diseguaglianza che và sotto il nome di Diseguaglianza di McMillan, dal nome dello scopritore (la cui foto è di fianco).

Teorema 1 (Diseguaglianza di McMillan). *Le lunghezze delle parole codice di una qualsiasi codifica unicamente decifrabile* $\mathsf{c} : \mathsf{X} \to \{0,1\}^*$ *soddisfano la diseguaglianza*

$$\sum_{x\in\mathsf{X}} 2^{-\ell(x)} \leq 1. \tag{3}$$

Dimostrazione. Per un generico intero k, consideriamo

$$\begin{aligned}
\left(\sum_{x\in\mathsf{X}} 2^{-\ell(x)}\right)^k &= \sum_{x_1\in\mathsf{X}} 2^{-\ell(x_1)} \times \sum_{x_2\in\mathsf{X}} 2^{-\ell(x_2)} \times \cdots \times \sum_{x_k\in\mathsf{X}} 2^{-\ell(x_k)} \\
&= \sum_{x_1\in\mathsf{X}} \sum_{x_2\in\mathsf{X}} \cdots \sum_{x_k\in\mathsf{X}} 2^{-\ell(x_1)} \times 2^{-\ell(x_2)} \times \cdots 2^{-\ell(x_k)} \\
&= \sum_{x_1\ldots x_k\in\mathsf{X}^k} 2^{-(\ell(x_1)+\ell(x_2)+\ldots+\ell(x_k))} \\
&= \sum_{\mathbf{x}\in\mathsf{X}^k} 2^{-\ell(\mathbf{x})} = \sum_{i=1}^{k\ell_{\max}} a(i)2^{-i}
\end{aligned}$$

dove $\ell_{\max} = \max\{\ell(x) : x \in \mathsf{X}\}$ ed $a(i)$ è pari al numero di sequenze sorgenti $\mathbf{x} \in \mathsf{X}^k$ che hanno una codifica $\mathsf{c}^*(\mathbf{x})$ binaria lunga i. Il numero *totale* di sequenze binarie lunghe i è pari a 2^i, da cui segue che il numero di sequenze sorgenti $\mathbf{x} \in \mathsf{X}^k$ che hanno una codifica $\mathsf{c}^*(\mathbf{x})$ binaria lunga i *non* può superare 2^i, altrimenti la codifica non sarebbe UD (detto in altri termini, se tale numero di sequenze sorgente fosse $> 2^i$ non avremmo sufficienti distinte sequenze binarie per poterle codificare ciascuna con una distinta sequenza binaria). Stiamo quindi dicendo che necessariamente vale $a(i) \leq 2^i, \forall i$. (L'ipotesi di UD l'abbiamo qui utilizzata). Otteniamo allora

$$\left(\sum_{x\in\mathsf{X}} 2^{-\ell(x)}\right)^k = \sum_{i=1}^{k\ell_{\max}} a(i)2^{-i} \leq \sum_{i=1}^{k\ell_{\max}} 2^i 2^{-i} = k\ell_{\max},$$

ovvero

$$\forall k \quad \sum_{x\in\mathsf{X}} 2^{-\ell(x)} \leq (k\ell_{\max})^{1/k} = 2^{\frac{1}{k}\log(k\ell_{\max})}.$$

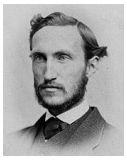
Poichè $\lim_{k\to\infty} \frac{1}{k}\log(k\ell_{\max}) = 0$, ne segue che $\sum_{x\in\mathsf{X}} 2^{-\ell(x)} \leq 1$, il che è quanto volevamo dimostrare. □

Da un punto di vista intuitivo abbiamo che per avere $\sum_{x\in\mathsf{X}} 2^{-\ell(x)} \leq 1$ deve capitare che i termini $2^{-\ell(x)}$ non possono essere troppo grandi, ovvero i termini $\ell(x)$ non possono essere troppo piccoli. Quindi, la condizione di UD ci dice che le lunghezze delle parole codice, che utilizziamo che codificare i simboli sorgenti, non possono essere troppo piccole. Ciò di fatto funge da vincolo nella minimizzazione.

Sulla base di questo risultato appena ottenuto, possiamo riformulare il nostro problema originale nel modo seguente

$$\begin{aligned} &\min \sum_{x\in\mathsf{X}} P(x)\ell(x) \\ \text{soggetto a } &\sum_{x\in\mathsf{X}} 2^{-\ell(x)} \leq 1 \\ &\ell(x) \in \mathbb{N} \quad \forall \mathbf{x} \in \mathsf{X}. \end{aligned} \tag{4}$$

Ricordiamo che in tale problema le variabili decisionali sono le $\ell(x)$, le $P(x)$ sono fissate e sono quelle della sorgente.



Questo è un problema di minimizzazione non facile da risolvere, sia a causa del vincolo di interezza sulle variabili $\ell(x)$, che della forma della regione su cui si minimizza. Per risolvere il problema (e molti altri in seguito ...), proviamo innanzitutto un risultato matematico molto utile. Il risultato và sotto il nome di *Diseguaglianza di Gibbs*, dal nome dello scopritore, la cui foto è di fianco. Introduciamo innanzitutto il seguente utile concetto.

Definizione 2 *Sia* $\boldsymbol{P} = (p_1, \ldots, p_n)$ *tale che* $p_i > 0$, $\forall i$, *e* $\sum_{i=1}^n p_i = 1$. *Sia inoltre* $\boldsymbol{Q} = (q_1, \ldots, q_n)$ *un vettore di reali tale che* $q_i > 0$, $\forall i$, *e* $\sum_{i=1}^n q_i \leq 1$. *Chiameremo Divergenza Informazionale tra* $\boldsymbol{P}$ *e* $\boldsymbol{Q}$ *la quantità*

$$D(\boldsymbol{P}||\boldsymbol{Q}) = \sum_{i=1}^n p_i \log \frac{p_i}{q_i} = \sum_{i=1}^n p_i \log \frac{1}{q_i} - H(P).$$

Sussiste il seguente risultato.

Lemma 1 (Diseguaglianza di Gibbs).

Sia $\boldsymbol{P} = (p_1, \ldots, p_n)$ *una distribuzione di probabilità, ovvero* $p_i > 0$, $\forall i$, *e* $\sum_{i=1}^n p_i = 1$. *Sia inoltre* $\boldsymbol{Q} = (q_1, \ldots, q_n)$ *un vettore di reali tale che* $q_i > 0$, $\forall i$, *e* $\sum_{i=1}^n q_i \leq 1$. *Vale che*

$$D(\boldsymbol{P}||\boldsymbol{Q}) = \sum_{i=1}^n p_i \log \frac{p_i}{q_i} \geq 0$$

con uguaglianza a zero se e solo se $p_i = q_i$, *per ogni* $i = 1, \ldots, n$.

Dimostrazione. Proveremo l'equivalente risultato che $-D(\mathbf{P}||\mathbf{Q}) \leq 0$. Si ha

$$\begin{aligned}
-D(\mathbf{P}||\mathbf{Q}) = -\sum_{i=1}^{n} p_i \log \frac{p_i}{q_i} &= \sum_{i=1}^{n} p_i \log \frac{q_i}{p_i} \quad (\text{poichè } -\log x = \log(1/x)) \\
&\leq \log \sum_{i=1}^{n} p_i \frac{q_i}{p_i} \quad (\text{applicando la diseguaglianza di Jensen alla funzione log}) \\
&= \log \sum_{i=1}^{n} q_i \leq \log 1 = 0.
\end{aligned}$$

Le condizioni sotto le quali si ha l'uguaglianza a zero si ottengono facilmente dalle condizioni per cui vale l'uguaglianza nella diseguaglianza di Jensen, (nel nostro caso ciò vorrebbe dire che $\frac{q_i}{p_i} = c, \forall i$, per qualche costante c) e dal fatto che $\log \sum_{i=1}^{n} q_i = 0$ se e solo se $\sum_{i=1}^{n} q_i = 1$. Queste due condizioni insiemi implicano che l'uguaglianza a 0 si ha se e solo se $p_i = q_i$, infatti:

$$q_i = p_i c \iff \sum_{i=1}^{n} q_i = \sum_{i=1}^{n} p_i c \iff 1 = c.$$

□

La diseguaglianza appena provata è uno dei motivi per cui la divergenza informazionale $D(\mathbf{P}||\mathbf{Q})$ viene usata, in molti campi (per es., Statistica, Machine Learning,...) come una misura di *dissimilarità* tra vettori di probabilità.

Ritorniamo al problema di stimare il minimo nella (4). Useremo il Lemma di Gibbs appena dimostrato. Consideriamo una *generica* funzione di codifica UD $\mathsf{c} : \mathsf{X} \to \{0,1\}^*$ per la sorgente X con alfabeto sorgente X, dove $\ell(x)$ denota la lunghezza di $\mathsf{c}(x)$. Applicando note proprietà dei logaritmi, otteniamo:

$$\begin{aligned}
\sum_{x \in \mathsf{X}} P(x)\ell(x) = \sum_{x \in \mathsf{X}} P(x) \log 2^{\ell(x)} &= \sum_{x \in \mathsf{X}} P(x) \log \frac{1}{2^{-\ell(x)}} \\
&= \sum_{x \in \mathsf{X}} P(x) \log \frac{P(x)}{P(x)2^{-\ell(x)}} \\
&= \sum_{x \in \mathsf{X}} P(x) \log \frac{P(x)}{2^{-\ell(x)}} + \sum_{x \in \mathsf{X}} P(x) \log \frac{1}{P(x)} \\
&= D(\mathbf{P}||\mathbf{Q}) + \sum_{x \in \mathsf{X}} P(x) \log \frac{1}{P(x)} \qquad \mathbf{Q} = (2^{-\ell(x_1)}, \ldots, 2^{-\ell(x_n)}) \\
&\geq \sum_{x \in \mathsf{X}} P(x) \log \frac{1}{P(x)} = H(X).
\end{aligned}$$

dove l'ultima diseguaglianza è conseguenza della Diseguaglianza di Gibbs. Notiamo che poichè la codifica c è UD, per la disuguaglianza di McMillan sappiamo che $\sum_{x \in \mathsf{X}} 2^{-\ell(x)} \leq 1$ e quindi per la disuguaglianza di Gibbs $D(\mathbf{P}||\mathbf{Q}) \geq 0$, con $\mathbf{Q} = (2^{-\ell(x_1)}, \ldots, 2^{-\ell(x_n)})$.

Visto che la diseguaglianza $\sum_{x \in \mathsf{X}} P(x)\ell(x) \geq H(X)$ vale *qualsivoglia* siano le lunghezze $\ell(x)$ di una codifica UD, varrà ovviamente anche che il $\min \sum_{x \in \mathsf{X}} P(x)\ell(x)$ calcolato in (4) è limitato inferiormente dall'entropia $H(X)$ della sorgente X. Quindi otteniamo che il minimo numero *medio* di bit per simbolo sorgente $\sum_{x \in \mathsf{X}} P(x)\ell(x)$ di una qualsivoglia codifica unicamente decifrabile dei simboli di una generica sorgente X non può essere inferiore all'entropia $H(X)$ della sorgente X.