Una vez muestreada una señal y transformada en valores pertenecientes a un conjunto discreto, esto es, *digitalizada*, un paso natural es *codificarla*, representarla de manera eficiente para su almacenamiento y transmisión. Por ejemplo, hay decenas de miles de palabras en español y podríamos codificar cada una con un número de cinco cifras. En cuanto lo pensemos un poco, veremos que en un texto normal es más eficiente codificar las palabras comunes con números más cortos, igual que cuando informalmente se abrevia "que" con "q".

Lo que se preguntó Shannon en un famosísimo artículo [4] fue cuán eficiente puede ser este proceso dependiendo de la frecuencia con que aparecen los datos que queramos codificar. Antes de abordar este problema, introdujo una función de las probabilidades muy importante en teoría de la información. Para motivarla, escribe literalmente en [4] que quiere definir *"una cantidad que mida, en algún sentido, cuánta información se* produce [...] *o, mejor, con qué frecuencia se produce"* y unas líneas más adelante *"una medida de cuánta* elección *está implicada en la selección de un evento o cuánta incertidumbre tenemos en el resultado"*.

Tomando prestado un nombre de la termodinámica, llamó a esta función *entropía* y la denotó con $H(p_1,\dots,p_n)$. Aquí $\sum_j p_j = 1$ y $p_j \in [0,1]$ porque son probabilidades. Parece normal limitarse a $p_j \in (0,1]$ para evitar situaciones que no ocurren nunca, y así lo haremos después, aunque ahora es irrelevante. El caso $p_1 = p_2 = \cdots = p_n = 1/n$ aparece varias veces y lo abreviaremos aquí escribiendo $\eta(n) = H(1/n, \overset{n\text{ veces}}{\dots\dots}, 1/n)$. Shannon pidió tres propiedades

i) H es continua.

ii) $\{\eta(n)\}_{n=1}^{\infty}$ es una sucesión creciente.

iii) $\eta(n) = H\big(\frac{a_1}{n},\dots,\frac{a_k}{n}\big) + \sum_{j=1}^{k} \frac{a_j}{n}\eta(a_j)$ para $a_j \in \mathbb{Z}_{\geq 0}$ con $\sum_{j=1}^{k} a_j = n$.

La primera propiedad es solo algo técnico, la segunda es natural porque más posibilidades dan más incertidumbre. La tercera es la que suena más rara. Para motivarla, imagina que tenemos que escoger una bola de una urna en la que hay n bolas distinguibles. La entropía es $\eta(n)$ porque cada una tiene probabilidad $1/n$. Digamos que alguien fija k colores y pinta a_j bolas del color j para cada $1 \leq j \leq k$ y después distribuye las bolas en k urnas de acuerdo con su color. A nosotros nos da igual el color que tengan para hacer una elección, pero ahora podríamos llevarla a cabo en dos pasos: primero escogemos una urna con la probabilidad que indique el número de bolas que contiene, lo cual tiene entropía $H\big(\frac{a_1}{n},\dots,\frac{a_k}{n}\big)$ y después dentro de esa urna escogemos una bola, lo que tiene entropía $\eta(b_j)$ para la urna j que escogeremos a_j veces de cada n. Sumar las incertidumbres es lo que da lugar a iii). Lo que indica esta propiedad es que la incertidumbre no varía si descomponemos una elección en varios pasos.

Shannon demostró que la única función que cumple las propiedades anteriores es

$$H(p_1,\dots,p_n) = -\sum_{j=1}^{n} p_j \log_2 p_j \tag{1}$$

o esta fórmula multiplicada por una constante positiva. Aquí $\log_2$ es el logaritmo en base dos

y se conviene que $p\log_2 p$ es cero si $p = 0$. Antes de probar esta afirmación, te propongo un ejercicio previo para que ordenes tus ideas.

1) Lee [5, §2] (quizá saltándote la entropía condicional), da un vistazo a la primera sección de [3], relee la explicación que te he dado aquí y con todo ello trata de escribir unas líneas motivando la noción de entropía. La extensión la dejo a tu arbitrio. Puede ser algo breve o unos cuantos párrafos.

Pasamos ahora a la prueba de (1) que se sigue de los cuatro ejercicios siguientes.

2) Comprueba que tomando $a_1 = n = k = 1$ en iii) se deduce $\eta(1) = 0$. Cambiando n por n^2 y con una elección adecuada de los a_j muestra que $\eta(n^2) = 2\eta(n)$. Elabora un argumento inductivo para concluir que, en general, $\eta(n^m) = m\,\eta(n)$ para $m \in \mathbb{Z}_{\geq 0}$.

3) Para cualquier $n \in \mathbb{Z}^+$ y $\ell \in \mathbb{Z}_{\geq 0}$ existe un $m \in \mathbb{Z}_{\geq 0}$ tal que $2^m \leq n^\ell \leq 2^{m+1}$. Utilizando las propiedades, prueba $m\,\eta(2) \leq \eta(n^\ell) \leq (m+1)\eta(2)$ y justifica

$$\frac{\eta(n)}{\log_2 n} = \lim_{\ell\to\infty} \frac{\eta(n^\ell)}{\log_2(n^\ell)} = \eta(2).$$

4) De lo anterior se sigue $\eta(n) = \eta(2)\log_2 n$. Despeja $H\big(\frac{a_1}{n}, \ldots, \frac{a_k}{n}\big)$ de iii) para obtener que (1) es cierto con $p_j = a_j/n$ salvo multiplicar por $\eta(2)$. Explica por qué este resultado se cumple aunque los p_j no sean fracciones.

5) Está claro que multiplicar por cualquier constante positiva no altera las propiedades, por tanto, es lícito escoger $\eta(2) = 1$. Comprueba que realmente (1) satisface las tres propiedades.

Parafraseando lo dicho al principio, en pocas palabras, el objetivo de Shannon era saber hasta qué punto se puede comprimir un fichero usando la codificación adecuada. Parece una pregunta demasiado ingenua, sin embargo tiene una respuesta matemática bastante precisa.

Para introducir la teoría de la codificación, suponemos que tenemos un conjunto finito $S = \{s_1, \ldots, s_n\}$ que es un espacio de probabilidad, es decir, cada s_j tiene asignada una probabilidad p_j y $\sum_j p_j = 1$. Extendiendo ligeramente la notación anterior, se define la entropía de S como $H(S) = H(p_1, \ldots, p_n)$. Uno puede tener en mente que S es un conjunto de caracteres o palabras que aparecen con cierta probabilidad.

Un *código binario prefijo* es una función $C : S \longrightarrow \mathcal{B}$ donde $\mathcal{B}$ es el conjunto de cadenas finitas de *bits* (listas de ceros y unos) tal que ningún $C(s_i)$ sea "prefijo" de un $C(s_j)$, es decir, que $C(s_j)$ no empiece por la cadena $C(s_i)$. La definición de C se extiende a cadenas de N elementos de S, esto es, a $S^N = S \times \overset{N \text{ veces}}{\cdots} \times S$, de la manera obvia: concatenando las imágenes. Por ejemplo, si $C(\mathtt{a}) = 1011$ y $C(\mathtt{b}) = 010$ entonces $C(\mathtt{a},\mathtt{b}) = C(\mathtt{ab}) = 1011010$. Aquí $(\mathtt{a},\mathtt{b})$ o $\mathtt{ab}$, como lo prefiramos escribir, está en S^2. En S^N se asigna a $(s_{j_1}, s_{j_2}, \ldots, s_{j_N})$ la probabilidad $p_{j_1}p_{j_2}\cdots p_{j_N}$ con ello se convierte también en un espacio de probabilidad. Si no ves claro que las probabilidades suman uno, escribe alguna línea en tu trabajo para justificarlo.

La condición de prefijo se introduce para que no haya ambigüedades y las codificaciones se puedan descifrar. Para ilustrar este problema, si en $S = \{\mathtt{a}, \mathtt{b}, \mathtt{c}\}$ definimos $C(\mathtt{a}) = 0$, $C(\mathtt{b}) = 1$ y $C(\mathtt{c}) = 01$, no se cumple la condición de prefijo porque $C(\mathtt{a})$ es prefijo de $C(\mathtt{c}) = 0$ y no sabríamos si 101 significa `bc` o `bab`.

6) Explica en pocas palabras porqué los códigos prefijo cumplen que $C : S^N \longrightarrow \mathcal{B}$ es inyectiva y que las imágenes para distintos valores de N son disjuntas, por tanto, siempre se pueden descifrar.

En realidad, hay códigos descifrables que no son prefijo, pero se puede demostrar que, en algún sentido, se pueden reducir a estos [2] y por ello los resultados matemáticos que probaremos (Teorema 1, Corolario 2 y Lema 3) son válidos para todos los códigos binarios descifrables. No nos meteremos en esto para no alargar más la hoja. A partir de ahora usaremos la palabra "código" como sinónimo de "código binario prefijo". También supondremos $p_j \neq 0$ porque no tiene sentido asignar codificaciones a símbolos que no aparecen nunca.

La *longitud* de una cadena de bits es el número de bits que contiene y la indicaremos con barras. Así escribiremos $|110| = 3$, $|01| = 2$, etc. En relación con esto, se define la *longitud media* de un código como

$$L(C) = \sum_{j=1}^{n} p_j \big|C(s_j)\big|$$

y, consecuentemente, si consideramos el código en S^N (sobre cadenas de N elementos)

$$L_N(C) = \sum_{j_1,\ldots,j_N=1}^{n} p_{j_1} \cdots p_{j_N} \big|C(s_{j_1}, \ldots, s_{j_N})\big|.$$

Estas definiciones son naturales: la codificación de cada elemento contribuye a la longitud de acuerdo con la frecuencia con que aparece.

Por ejemplo, en $S = \{\mathtt{a}, \mathtt{b}, \mathtt{c}\}$ con $p_1 = \frac{1}{2}$, $p_2 = p_3 = \frac{1}{4}$ consideremos los códigos C, C' y C'' que responden a estas tablas:

	a	b	c
C	00	01	10

	a	b	c
C'	10	0	11

	a	b	c
C''	0	10	11

El primero es el más obvio: números consecutivos en binario. Los otros son mejores porque reducen la longitud y el mejor de todos es el último porque da una codificación más corta al elemento más frecuente, con lo que conseguiremos una mayor *compresión*.

7) Comprueba que, realmente, con las definiciones anteriores $L(C) > L(C') > L(C'')$. Calcula también $L_2(C)$ y $L_2(C'')$.

En breve veremos que el siguiente ejercicio se deduce del Teorema 1, pero ahora, para practicar, quiero que lo hagas sin usarlo. Si quieres, no lo reflejes en el trabajo.

8) Demuestra que C'' es un código óptimo en S en cuanto a la compresión, en el sentido de que no existe otro código C''' con $L(C''') < L(C'')$.

Con estas definiciones, ya podemos dar una formulación matemática a la pregunta de cuánto se puede comprimir un fichero. Lo que deseamos es encontrar una codificación que minimice la longitud media. Shannon no dio una solución completa a este problema (lo hizo Huffman cuatro años después), pero estableció unos límites superior e inferior que son muy importantes en la teoría y están recogidos en el siguiente resultado. En inglés se le llama *source coding theorem*. En español alguna vez he leído *teorema de codificación de fuentes*.

Teorema 1. *Sea S como antes y $H(S)$ la entropía asociada a sus probabilidades, entonces $H(S) \leq \min_C L(C) < H(S) + 1$ donde el mínimo es sobre todos los códigos C definidos en S.*

En términos prácticos, la siguiente consecuencia (no obvia) es bastante interesante.

Corolario 2. *Para cualquier $N \in \mathbb{Z}^+$ se tiene $NH(S) \leq \min_C L_N(C) < NH(S) + 1$ donde el mínimo es sobre todos los códigos C definidos en S^N. En particular, cuando N tiende a infinito, $N^{-1} \min_C L_N(C) \to H(S)$.*

9) Comprueba que el código C'' del ejercicio anterior cumple $L(C'') = H(S)$ y, por tanto, según el Teorema 1, tiene la mínima longitud media posible.

Te cuento un ejemplo numérico para que veas el interés práctico y entiendas el significado de estos resultados. Supongamos un fichero con un texto muy largo compuesto solo por dos caracteres `A` y `B` que aparecen aleatoriamente con frecuencias del 80 % y del 20 %. Es decir, $S = \{\texttt{A}, \texttt{B}\}$, $p_1 = 0.8$, $p_2 = 0.2$. la entropía es aproximadamente 0.72. Si el texto tiene T letras, los ordenadores habitualmente codifican cada letra con 8 bits, con lo cual ocupará $8T$ bits en la memoria. Si codificamos cada letra por separado, lo único sensato es lo obvio de codificar $C(\texttt{A}) = 0$ y $C(\texttt{B}) = 1$ con lo que todo el texto ocupará T bits en memoria. Se tiene $L(C) = 1$ lo que cuadra con el rango del Teorema 1. Si agrupamos los caracteres de N en N, el texto quedará dividido en T/N bloques y según el Corolario 2, con un código adecuado, cada bloque pasará a ocupar una longitud de entre $0.72N$ y $0.72N + 1$ bits. En total el texto ocupará entre $0.72T$ y $0.72(T + 1/N)$ bits. Si N es muy grande, lo que requiere que T sea muchísimo mayor (para que tenga sentido considerar la media de las longitudes), en el límite la longitud será $0.72T$, es decir, la entropía está relacionada con el factor óptimo de compresión.

El software de compresión que tenemos habitualmente en nuestros ordenadores, como `zip`, se basa en un algoritmo que en el límite, para ficheros enormes, tiende a ser óptimo, aunque es justo aclarar que la convergencia es lenta. Te propongo en los tres ejercicios siguientes que compruebes estas cosas en la práctica[1]. En el trabajo incluye el programa y las explicaciones

[1]Si estás muy perdida, lo mismo te sirve de algo el programa python que tengo en `https://matematicas.uam.es/~fernando.chamizo/dark/d_anti_comp.html` y los comentarios que hago allí. Quizá tengas que incluir también `import random` en la cabecera.

teóricas que consideres convenientes.

10) Escribe un programa en el lenguaje que prefieras que genere un texto de 10^6 caracteres que responda al esquema anterior y que lo guarde en un fichero. Es decir que esté compuesto por caracteres `A` y `B` al azar teniendo el primero una frecuencia del 80 % y el segundo del 20 %.

11) Mira las propiedades del fichero y comprueba que ocupa 8 millones de bits. Seguramente te aparezca en otras unidades, si es en bytes, será un millón. Ahora comprime el fichero con el *software* que tengas a mano y comprueba que se cumple que el tamaño en bits es mayor que $10^6 H(S)$. En principio cambiando 10^6 por longitudes mucho mayores se debería ver que la tasa de compresión se acerca a $H(S)$, pero creo que es difícil que lo puedas apreciar por la lentitud de la convergencia.

12) Experimenta con otras asignaciones de las probabilidades o con S incluyendo más caracteres comprobando en cada caso que se respecta la cota inferior. Se obtienen mejores resultados con probabilidades uniformes. Por ejemplo, prueba con S conteniendo 10 símbolos y $p_1 = \cdots = p_{10} = \frac{1}{10}$.

Pasamos ahora a las demostraciones. En primer lugar, veamos porqué el Corolario 2 es realmente un corolario. Si lo piensas un instante, lo único que hay que hacer es resolver el siguiente ejercicio:

13) Demuestra que $H(S^N) = NH(S)$. Indicación: Justifica que el coeficiente de $\log_2 p_m$ en $\sum_{j_1,\ldots,j_N=1}^{n} p_{j_1}\cdots p_{j_N} \sum_{l=1}^{N} \log_2 p_{j_l}$ es $\sum_{l=1}^{N} \sum_{\substack{j_1,\ldots,j_N=1 \\ j_l=m}}^{n} p_{j_1}\cdots p_{j_N} = \sum_{l=1}^{N} p_m \big(\sum_{j=1}^{N} p_j\big)^{N-1}$ y esto es Np_m.

Para la prueba del Teorema 1 se utiliza el siguiente resultado auxiliar:

Lema 3 (Desigualdad de Kraft)**.** *Dado un código sobre $S = \{s_1, \ldots, s_n\}$, las longitudes de las codificaciones $l_j = \big|C(s_j)\big|$ satisfacen $\sum_{j=1}^{n} 2^{-l_j} \leq 1$. Recíprocamente, dados $l_j \in \mathbb{Z}^+$ cualesquiera que verifiquen esta desigualdad, siempre existe un código C tal que $l_j = \big|C(s_j)\big|$.*

14) Lee la demostración de la desigualdad de Kraft en [6] y refléjala con tus palabras en el trabajo. Solo tienes que leer la parte titulada *Proof for prefix codes*, que son los que consideramos aquí. Debes suponer $r = 2$ porque nuestros códigos son binarios.

Lo que queda es probar el Teorema 1. Los dos primeros ejercicios dan la cota inferior y el tercero la cota superior.

15) Considera la función $F(x_1, \ldots, x_n) = -\sum_{j=1}^{n} p_j \log_2 x_j$. Usando multiplicadores de Lagrange u otro método, prueba que su mínimo en la región $\{\vec{x} \in (0,1)^n : \sum_{j=1}^{n} x_j = 1\}$ se alcanza para $x_j = p_j$.

16) Utilizando el ejercicio anterior y el Lema 3, escribiendo $\sigma = \left(\sum_{j=1}^{n} 2^{-l_j}\right)^{-1}$ con la notación allí indicada, deduce la cota inferior en el Teorema 1 justificando

$$H(S) = F(p_1, \ldots, p_n) \leq F(2^{-l_1}\sigma, \ldots, 2^{-l_n}\sigma) \leq F(2^{-l_1}, \ldots, 2^{-l_n}) = L(C).$$

17) Para cada j sea l_j un entero que cumple $-\log_2 p_j \leq l_j < 1 - \log_2 p_j$. Explica por qué se puede aplicar la segunda parte del Lema 3 y deduce de ello que existe un código de longitud media menor que $\sum_{j=1}^{n} p_j(1 - \log_2 p_j)$ que es igual a $H(S) + 1$.

Por si quieres mirar bibliografía sobre los conceptos que parecen en esta hoja, la referencia [1] es buenísima y elemental. Es un clásico que supongo que se podrá encontrar por internet. Si quieres algo más matemático y de mayor nivel, [2] es muy legible.

Tarea a entregar. Escribe un documento que combine las soluciones de los ejercicios anteriores. Como no quiero que te agobies al final, me estoy planteando reducir las hojas a 4 a costa de extender un poco su longitud y por eso en está propongo una extensión de a lo más 8 páginas sin contar ni código ni bibliografía. La idea es que, si quieres, los ejercicios pueden complementarse con más explicaciones que reflejen las que doy aquí y leas en las referencias. El resultado será el tercer capítulo de tu TFG con el título *Codificación y compresión* o cualquier variante que te parezca adecuada. Quizá es más propio *Codificación* o *Teoría de la información*, escoge el que prefieras.

Referencias

[1] A. Rényi. *A diary on information theory.* Akadémiai Kiadó (Publishing House of the Hungarian Academy of Sciences), Budapest, 1984.

[2] S. Roman. *Coding and information theory*, volume 134 of *Graduate Texts in Mathematics.* Springer-Verlag, New York, 1992.

[3] T. D. Schneider. Information theory primer. `http://www.ece.iit.edu/~biitcomm/research/references/Thomas%20D.%20Schneider/Information%20Theory%20Primer.pdf`, 2004.

[4] C. E. Shannon. A mathematical theory of communication. *Bell System Tech. J.*, 27:379–423, 623–656, 1948.

[5] R. Tesoro. Cuando la taba se eleva, brota el azar. *Gac. R. Soc. Mat. Esp.*, 16(3):543–559, 2013. https://gaceta.rsme.es/english/abrir.php?id=1162.

[6] Wikipedia contributors. Kraft-McMillan inequality — Wikipedia, the free encyclopedia. https://en.wikipedia.org/w/index.php?title=Kraft%E2%80%93McMillan_inequality&oldid=1254083406, 2024. [Online; accessed 1-February-2025].