Curso
2010 | 2011

Cálculo de la Entropía de un Idioma

Práctica 1

David Morales Sáez
Alberto Manuel Mireles Suárez

Introducción

En esta práctica hemos implementado una aplicación que permite calcular la entropía del lenguaje castellano. La entropía es el valor medio de la cantidad de información que contiene un mensaje, partiendo del concepto de incertidumbre. Por ello, la cantidad de información transmitida es mayor cuanto más se disipe la incertidumbre, por lo que los caracteres con menor frecuencia de aparición aportan mayor información.

Como ejemplo, hemos empleando dos textos de diferente longitud para poder comparar los resultados obtenidos.

Desarrollo de la práctica

Para llevar a cabo el cálculo de la entropía, se ha diseñado una interfaz, desde la cual, puede escribirse o pegarse un texto para llevar a cabo el cálculo en base al texto introducido.

La aplicación muestra la probabilidad de aparición de cada carácter al igual que la probabilidad de aparición de ciertos caracteres dobles. En base a esos datos, se calcula la entropía para los caracteres simples y los caracteres dobles.

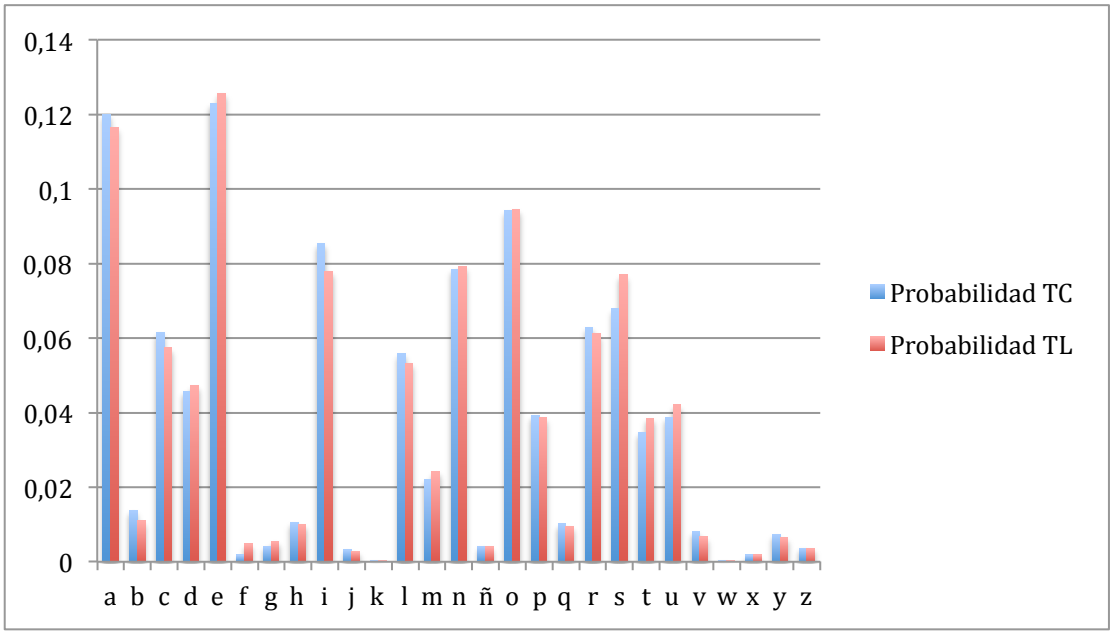
Para realizar los cálculos hemos extraído dos textos de la prensa digital actual de diferente tamaño (TC : Texto corto ; TL: Texto largo) y hemos obtenido los siguientes resultados.

Para los caracteres simples:

	Probabilidad TC	Probabilidad TL
a	0,120092	0,116607
b	0,013857	0,010992
c	0,061432	0,057348
d	0,045727	0,047312
e	0,122864	0,125687
f	0,001848	0,004779
g	0,004157	0,005496
h	0,010624	0,010036
i	0,08545	0,077897
j	0,003233	0,002867
k	0,000462	0,000239
l	0,055889	0,053286
m	0,022171	0,024134
n	0,078522	0,079092

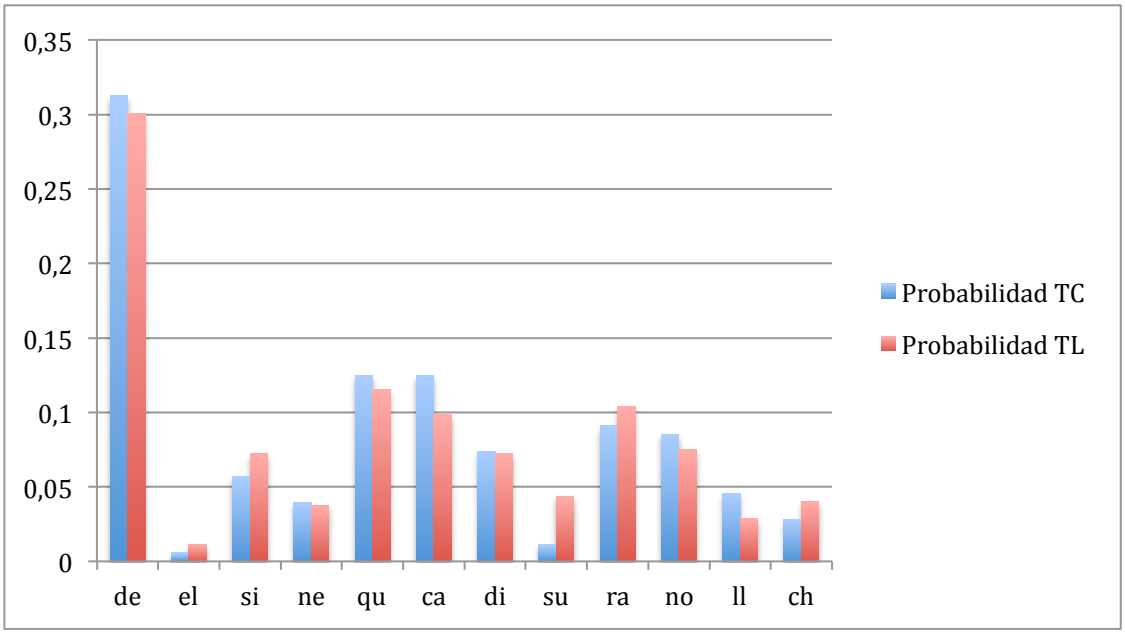
ñ	0,004157	0,004062
o	0,094226	0,094385
p	0,039261	0,03871
q	0,010162	0,009558
r	0,062818	0,061171
s	0,067898	0,076941
t	0,034642	0,038471
u	0,038799	0,042055
v	0,008214	0,006691
w	0,000462	0,000239
x	0,001848	0,001912
y	0,00739	0,006452
z	0,003695	0,003584
Entropía	4,002645	4,005368

La representación gráfica es la siguiente:

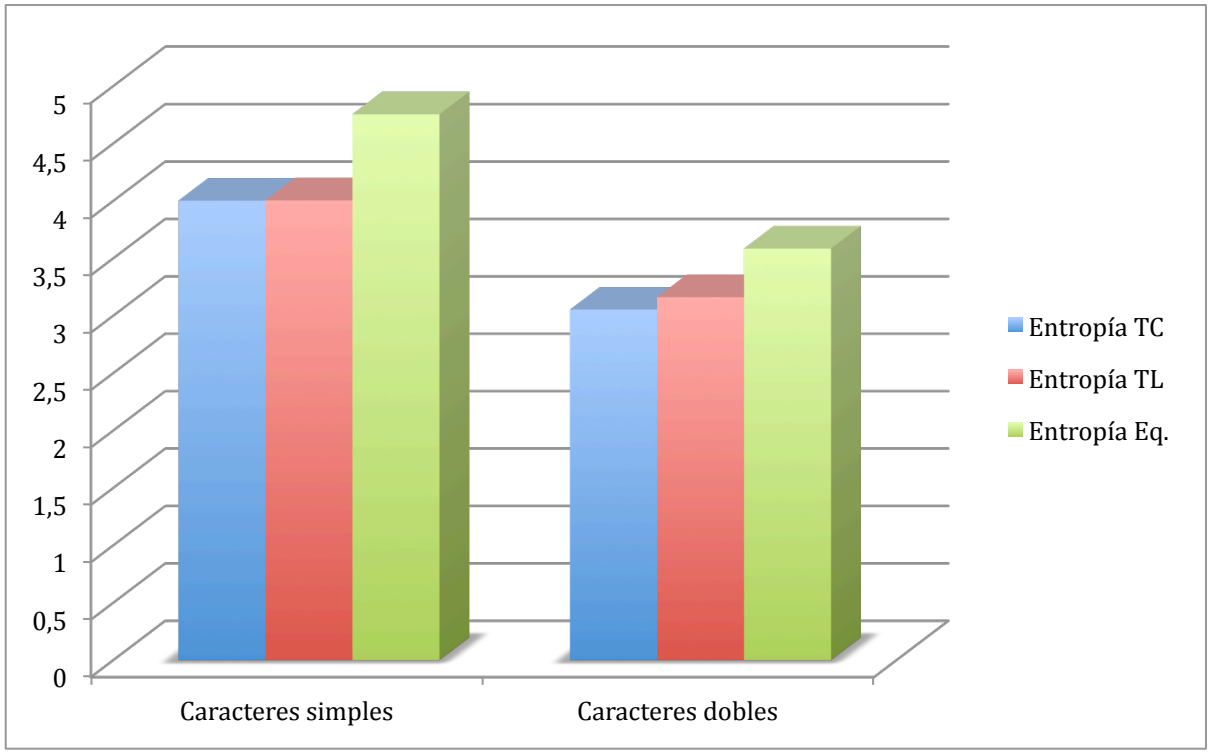


Para los caracteres dobles, los resultados son los siguientes:

	Probabilidad TC	Probabilidad TL
de	0,3125	0,300578
el	0,005682	0,011561
si	0,056818	0,072254
ne	0,039773	0,037572
qu	0,125	0,115607
ca	0,125	0,098266
di	0,073864	0,072254
su	0,011364	0,043353
ra	0,090909	0,104046
no	0,085227	0,075145
ll	0,045455	0,028902
ch	0,028409	0,040462
Entropía	3,053868	3,161653



A su vez, al representar gráficamente las entropías de los deferentes textos, obtenemos:



En la gráfica, se representa, junto a las entropías de los caracteres simples y dobles, la entropía cuando todos los caracteres son equiprobables, es decir, el valor máximo de la entropía, que para los caracteres simples es aproximadamente 4,7448 y para los caracteres dobles 3,5849.

Conclusión

Una vez estudiados los resultados obtenidos tenemos un cierto conocimiento sobre el idioma castellano. En base a los textos de muestra que hemos empleado, la entropía del idioma es de aproximadamente 4 ya que los resultados obtenidos en ambos textos son muy similares.

A su vez, examinando las gráficas, podemos observar que las letras más frecuentes son la E y la A mientras que las menos frecuentes son la K y la W, al menos en los textos usados. Como podemos ver, en el idioma hay letras que son muy empleadas y otras que apenas se usan, por lo que hay grandes diferencias en la gráfica, aunque los resultados entre el texto corto y el largo son similares.

Cuando observamos los datos de los caracteres dobles, con diferencia, el más frecuente es DE y el menos EL. Los resultados obtenidos pueden diferir dependiendo del texto empleado para calcular las probabilidades ya que pueden encontrarse textos en los que no aparezca alguna letra. Por ello, para poder obtener resultados más fiables se han de emplear textos más largos.

Anexo

Se incluye el código del cálculo de la entropía del lenguaje, eliminando las partes de la interfaz del programa:

```
 float probabilidades[28];
 float prob_par[12];

 for(int i = 0; i<28; i++)
    probabilidades[i]=0;
 for(int i=0; i<12; i++)
    prob_par[i]=0;

int veces[28], par[12];
int ncaracteres=0;
int npar=0;
for(int j = 0; j<28; j++)
    veces[j] = 0;
for(int j=0; j<12; j++)
    par[j] = 0;
 //En “frase” tenemos el texto introducido y “lon” es
 su tamaño
 int lon = strlen(frase);
 if(lon==0)
    return ;
 int i;
 for(i = 0; i<lon; i++)
 {
    switch(frase[i])
    {
        case 'a':
            veces[0]++;
            ncaracteres++;
            break;
        case 'b':
            veces[1]++;
            ncaracteres++;
            break;
        case 'c':
            veces[2]++;
            ncaracteres++;
            if((frase[i+1]=='a')||(frase[i+1]=='A'))
            {
                npar++;
                par[5]++;
            }
            if((frase[i+1]=='h')||(frase[i+1]=='H'))
            {
                npar++;
```

```
            par[11]++;
        }
        break;
    case 'd':
        veces[3]++;
        ncaracteres++;
        if((frase[i+1]=='e')||(frase[i+1]=='E'))
        {
            npar++;
            par[0]++;
        }
        if((frase[i+1]=='i')||(frase[i+1]=='I'))
        {
            npar++;
            par[6]++;
        }
        break;
    case 'e':
        veces[4]++;
        ncaracteres++;
        break;
    case 'f':
        veces[5]++;
        ncaracteres++;
        if((frase[i+1]=='l')||(frase[i+1]=='L'))
        {
            npar++;
            par[1]++;
        }
        break;
    case 'g':
        veces[6]++;
        ncaracteres++;
        break;
    case 'h':
        veces[7]++;
        ncaracteres++;
        break;
    case 'i':
        veces[8]++;
        ncaracteres++;
        break;
    case 'j':
        veces[9]++;
        ncaracteres++;
        break;
    case 'k':
        veces[10]++;
        ncaracteres++;
        break;
    case 'l':
        veces[11]++;
```

```
    ncaracteres++;
    if((frase[i+1]=='l')||(frase[i+1]=='L'))
    {
        npar++;
        par[10]++;
    }
    break;
case 'm':
    veces[12]++;
    ncaracteres++;
    break;
case 'n':
    veces[13]++;
    ncaracteres++;
    if((frase[i+1]=='e')||(frase[i+1]=='E'))
    {
        npar++;
        par[3]++;
    }
    if((frase[i+1]=='o')||(frase[i+1]=='O'))
    {
        npar++;
        par[9]++;
    }
    break;
case 'o':
    veces[15]++;
    ncaracteres++;
    break;
case 'p':
    veces[16]++;
    ncaracteres++;
    break;
case 'q':
    veces[17]++;
    ncaracteres++;
    if((frase[i+1]=='u')||(frase[i+1]=='U'))
    {
        npar++;
        par[4]++;
    }
    break;
case 'r':
    veces[18]++;
    ncaracteres++;
    if((frase[i+1]=='a')||(frase[i+1]=='A'))
    {
        npar++;
        par[8]++;
    }
    break;
case 's':
```

```
            veces[19]++;
            ncaracteres++;
            if((frase[i+1]=='i')||(frase[i+1]=='i'))
            {
                npar++;
                par[2]++;
            }
            if((frase[i+1]=='u')||(frase[i+1]=='U'))
            {
                npar++;
                par[7]++;
            }
            break;
        case 't':
            veces[20]++;
            ncaracteres++;
            break;
        case 'u':
            veces[21]++;
            ncaracteres++;
            break;
        case 'v':
            veces[22]++;
            ncaracteres++;
            break;
        case 'w':
            veces[23]++;
            ncaracteres++;
            break;
        case 'x':
            veces[24]++;
            ncaracteres++;
            break;
        case 'y':
            veces[25]++;
            ncaracteres++;
            break;
        case 'z':
            veces[26]++;
            ncaracteres++;
            break;
        case 'A':
            veces[0]++;
            ncaracteres++;
            break;
        case 'B':
            veces[1]++;
            ncaracteres++;
            break;
        case 'C':
            veces[2]++;
            ncaracteres++;
```

```
    if((frase[i+1]=='a')||(frase[i+1]=='A'))
    {
        npar++;
        par[0]++;
    }
    if((frase[i+1]=='h')||(frase[i+1]=='H'))
    {
        npar++;
        par[11]++;
    }
    break;
case 'D':
    veces[3]++;
    ncaracteres++;
    if((frase[i+1]=='e')||(frase[i+1]=='E'))
    {
        npar++;
        par[0]++;
    }
    if((frase[i+1]=='i')||(frase[i+1]=='I'))
    {
        npar++;
        par[6]++;
    }
    break;
case 'E':
    veces[4]++;
    ncaracteres++;
    if((frase[i+1]=='L')||(frase[i+1]=='l'))
    {
        npar++;
        par[1]++;
    }
    break;
case 'F':
    veces[5]++;
    ncaracteres++;
    break;
case 'G':
    veces[6]++;
    ncaracteres++;
    break;
case 'H':
    veces[7]++;
    ncaracteres++;
    break;
case 'I':
    veces[8]++;
    ncaracteres++;
    break;
case 'J':
    veces[9]++;
```

```
    ncaracteres++;
    break;
case 'K':
    veces[10]++;
    ncaracteres++;
    break;
case 'L':
    veces[11]++;
    ncaracteres++;
    break;
case 'M':
    veces[12]++;
    ncaracteres++;
    if((frase[i+1]=='L')||(frase[i+1]=='l'))
    {
        npar++;
        par[10]++;
    }
    break;
case 'N':
    veces[13]++;
    ncaracteres++;
    if((frase[i+1]=='e')||(frase[i+1]=='E'))
    {
        npar++;
        par[3]++;
    }
    if((frase[i+1]=='o')||(frase[i+1]=='O'))
    {
        npar++;
        par[9]++;
    }
    break;
case 'O':
    veces[15]++;
    ncaracteres++;
    break;
case 'P':
    veces[16]++;
    ncaracteres++;
    break;
case 'Q':
    veces[17]++;
    ncaracteres++;
    if((frase[i+1]=='u')||(frase[i+1]=='U'))
    {
        npar++;
        par[4]++;
    }
    break;
case 'R':
    veces[18]++;
```

```
            ncaracteres++;
            if((frase[i+1]=='a')||(frase[i+1]=='A'))
            {
                npar++;
                par[8]++;
            }
            break;
        case 'S':
            veces[19]++;
            ncaracteres++;
            if((frase[i+1]=='i')||(frase[i+1]=='I'))
            {
                npar++;
                par[2]++;
            }
            if((frase[i+1]=='u')||(frase[i+1]=='U'))
            {
                npar++;
                par[7]++;
            }
            break;
        case 'T':
            veces[20]++;
            ncaracteres++;
            break;
        case 'U':
            veces[21]++;
            ncaracteres++;
            break;
        case 'V':
            veces[22]++;
            ncaracteres++;
            break;
        case 'W':
            veces[23]++;
            ncaracteres++;
            break;
        case 'X':
            veces[24]++;
            ncaracteres++;
            break;
        case 'Y':
            veces[25]++;
            ncaracteres++;
            break;
        case 'Z':
            veces[26]++;
            ncaracteres++;
            break;
        case ' ':
            veces[27]++;
           // ncaracteres++;
```

```
    break;
default:
    int s = texto.at(i).toAscii();
    switch (s)
    {
    case -63: //Á
        veces[0]++;
        ncaracteres++;
        break;
    case -31: //á
        veces[0]++;
        ncaracteres++;
        break;
    case -55: //É
        veces[4]++;
        ncaracteres++;
        break;
    case -23: //é
        veces[4]++;
        ncaracteres++;
        break;
    case -51: //Í
        veces[8]++;
        ncaracteres++;
        break;
    case -19: //í
        veces[8]++;
        ncaracteres++;
        break;
    case -45: //Ó
        veces[15]++;
        ncaracteres++;
        break;
    case -13: //ó
        veces[15]++;
        ncaracteres++;
        break;
    case -8: //Ú
        veces[21]++;
        ncaracteres++;
        break;
    case -6: //ú
        veces[21]++;
        ncaracteres++;
        break;
    case -4: //ü
        veces[21]++;
        ncaracteres++;
        break;
    case -47: //Ñ
        veces[14]++;
        ncaracteres++;
```

```
                    break;
                case -15: //ñ
                    veces[14]++;
                    ncaracteres++;
                    break;
                }
                break;
        }
    }
    if(ncaracteres!=0)
        for(int k=0; k<28; k++)
            probabilidades[k] =
(veces[k])*((1.)/ncaracteres);
    if(npar!=0)
        for(int k=0; k<12; k++)
            prob_par[k] = (par[k])*((1.)/npar);

float entropia = 0.;
    for(int j=0; j<27; j++)
        if(probabilidades[j]!=0.)
            entropia+=
(probabilidades[j]*log2(probabilidades[j]));
    entropia *=-1.;
//Imprimimos la entropía de los caracteres simples

entropia = 0;
    for(int j=0; j<12; j++)
        if(prob_par[j]!=0)
            entropia+=(prob_par[j]*log2(prob_par[j]));
    entropia *=-1;
//Imprimimos la entropía de los caracteres dobles
```