UNIVERSITAT POLITÈCNICA DE VALÈNCIA

DSIC
DEPARTAMENT DE SISTEMES INFORMÀTICS I COMPUTACIÓ

Departamento de sistemas informáticos y computación

Predicción Estructurada Estadística

Cuestionario 1-A

Máster Universitario en Inteligencia Artificial, Reconocimiento de Formas e Imagen Digital

Autor
Francisco Javier Gil-Terrón Rodríguez

2021 - 2022

Tabla de contenidos

1. Cuestiones teóricas

1.1 Cuestión 1

Los lenguajes generados por una gramática probabilística no son necesariamente probabilísticos debido a que, para que un lenguaje sea probabilístico, la suma de las probabilidades de todas las cadenas pertenecientes a dicho lenguaje debe ser uno, lo que únicamente se cumple si el lenguaje ha sido generado con una gramática consistente.

Asimismo, no todos los lenguajes probabilísticos pueden ser generados por gramáticas probabilísticas puesto que existen lenguajes probabilísticos, como el que se propone en el ejemplo las diapositivas, que siguen una expresión factorial que no es representable con polinomios de finitos términos.

1.2 Cuestión 2

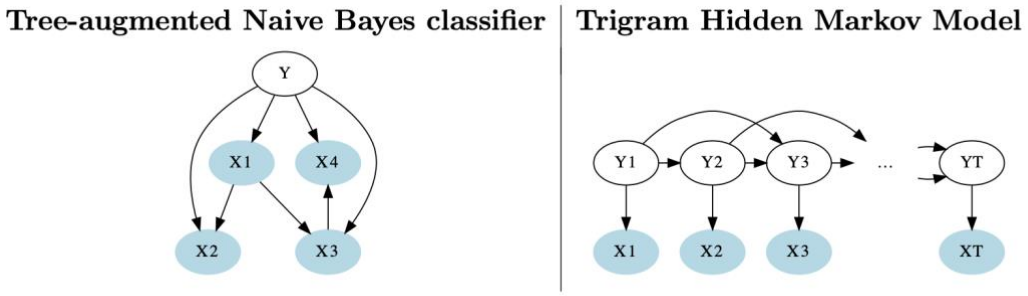

Figura 1 – Tree-augmented Naive Bayes classifier and Trigram Hidden Markov Model

Dadas las Redes Bayesianas que definen los modelos presentados en la figura anterior, proporcionar el resultado de la factorización para las probabilidades conjuntas $p(x^T_1, y)$ y $p(x^T_1, y^T_1)$ respectivamente.

$$p(x_1^T, y) = p(y) \cdot p(x_1|y) \cdot p(x_2|y, x_1) \cdot p(x_3|y, x_1) \cdot p(x_4|y, x_3)$$

$$p(x_1^T, y_1^T) = p(y_1) \cdot p(x_1|y_1) \cdot p(y_2|y_1) \cdot p(x_2|y_2) \cdot \prod_{t=3}^{T} p(y_t|y_{t-1}, y_{t-2}) \cdot p(x_t|y_t)$$

1.3 Cuestión 3

➢ **Definition**: Score for $x_{t+1} \dots x_T$ starting at $y_t = s \in \mathcal{Y}$

$$\beta_t(s) \quad \stackrel{\text{def}}{=} \quad \sum_{y_{t+1}^T;\, y_t = s} \quad \prod_{i=t+1}^{T} \Psi_i\,(y_{i-1},\ y_i,\ x_i)$$

➢ **Initialization**: $\forall\, s \in \mathcal{Y}$

$$\beta_T(s) \ = \ 1$$

➢ **Recursion**: $\forall\, t = T-1 \dots 1;$ and $\forall\, s \in \mathcal{Y}$

$$\beta_t(s) \ = \ \sum_{s' \in \mathcal{Y}} \Psi_{t+1}\,(y_t = s,\ y_{t+1} = s',\ x_{t+1}) \ \cdot \ \beta_{t+1}(s')$$

➢ **Final result**: The optimal score for x is:

$$\sum_{s} \ \Psi_1\,(y_0 = \text{null},\ y_1 = s,\ x_1) \ \cdot \ \beta_1(s)$$

Figura 2 – Algoritmo Backward

En primer lugar, la definición varía ligeramente para esta nueva versión del algoritmo:

$$\beta_t(s) = \sum_{y_{t+1}^T; y_{t=s}} \prod_{i=t+1}^{T} \Psi_i(y_{i-2}, y_i, x_i)$$

Por otro lado, en la inicialización habrá que añadir el otro estado inicial al que ya está en el algoritmo original:

$$\beta_{T-1}(s) = 1$$

A su vez, habrá que modificar la recursión teniendo en cuenta que el estado T-1 ya ha sido contemplado, y la forma de las nuevas transiciones:

$$\forall t \ = \ T-2 \dots 1 \text{ and } \forall\, s \in \mathcal{Y}$$

$$\beta_t(s) = \sum_{s' \in \mathcal{Y}} \Psi_{t+2}(y_t = s, y_{t+2} = s', x_{t+2}) \cdot \beta_{t+2}(s')$$

Por último, el resultado final tendrá la siguiente forma:

$$\beta_t(s) = \sum_{s} \Psi_1(y_{-1} = null, y_1 = s, x_1) \cdot \beta_1(s) \cdot \sum_{s'} \Psi_2(y_0 = null, y_2 = s', x_1) \cdot \beta_2(s')$$

1.4 Cuestión 4

➢ **Definition**: Given $x = x_1 \dots x_T \in \Sigma^*$ and $A \in N$

$$e(A, i, i+l) \quad \stackrel{\text{def}}{=} \quad P_\theta(A \stackrel{*}{\Rightarrow} x_{i+1} \dots x_{i+l})$$

➢ **Initialization**: $\forall\, A \in N; \qquad \forall\, i : 0 \dots T-1;$

$$e(A, i, i+1) \;=\; p(A \to b) \;\cdot\; \delta(b, x_{i+1})$$

➢ **Recursion**: $\forall\, A \in N; \qquad \forall\, l : 2 \dots T; \qquad \forall\, i : 0 \dots T-l;$

$$e(A, i, i+l) \;=\; \sum_{B,C \in N} \Big\{ p(A \to BC) \cdot \sum_{k=1,\dots,l-1} e(B, i, i+k) \;\cdot\; e(C, i+k, i+l) \Big\}$$

➢ **Final result**: The sentence probability is:

$$P_\theta(x) \;=\; e(S, 0, T)$$

Figura 3 – Algoritmo CKY-based Inside

En primer lugar, la definición del algoritmo no se ve modificada, que de igual manera sucede con la inicialización, ya que ramificar únicamente por la derecha modifica el algoritmo en la recursión.

Así pues, en la recursión, lo rama izquierda que antes de desarrollaba ahora acaba necesariamente en un nodo terminal, por lo que ya no será necesaria iterar sobre k y la nueva forma será:

$$\forall\, \mathrm{A} \in \mathrm{N}; \forall\, \mathrm{l}: 2 \dots \mathrm{T}; \;\; \forall\, i: 0 \dots T-l$$

$$e(A, i, i+l) = \sum_{B,C \in N} p(A \to BC) \cdot p(B \to b) \cdot e(C, i+1, i+l)$$

Finalmente, el resultado final también tendrá la misma expresión, mientras que el coste temporal del nuevo algoritmo se reducirá y será lineal *O*(n) en este caso, ya que en cada iteración se expande un nodo y el otro es terminal.

1.5 Cuestión 5

➢ **Definition**: Score for $x_1 \dots x_t$ ending at $y_t = s \in \mathcal{Y}$

$$\alpha_t(s) \quad \stackrel{\text{def}}{=} \quad \sum_{y_1^t;\, y_t = s} \; \prod_{i=1}^{t} \Psi_i\,(y_{i-1},\; y_i,\; x_i)$$

➢ **Initialization**: $\forall\, s \in \mathcal{Y}$

$$\alpha_1(s) \;=\; \Psi_1\,(y_0 = \text{null},\; y_1 = s,\; x_1)$$

➢ **Recursion**: $\forall\, t = 2 \dots T;$ and $\forall\, s \in \mathcal{Y}$

$$\alpha_t(s) \;=\; \sum_{s' \in \mathcal{Y}} \alpha_{t-1}(s') \;\cdot\; \Psi_t\,(y_{t-1} = s',\; y_t = s,\; x_t)$$

➢ **Final result**: The optimal score for x is:

$$\sum_{s} \alpha_T\;(s)$$

Figura 4 – Algoritmo Forward

Al igual que en la cuestión 3, la definición varía ligeramente teniendo en cuenta, en este caso, que se trabajará con trigramas:

$$\alpha_{\mathrm{t}}(s) = \sum_{y_1^t; y_{t=s}} \prod_{i=1}^{t} \Psi_i(y_{i-2}, y_{i-1}, y_i, x_i)$$

Y la inicialización deberá ser modificada de la siguiente forma $\forall\, s \in \mathcal{Y}$:

$$\alpha_1(\mathrm{s}) = \Psi_1(y_{-1} = \mathrm{null}, \mathrm{y}_0 = \mathrm{null}, \mathrm{y}_1 = s, \mathrm{x}_1)$$

A su vez, la recursión también cambia teniendo en cuenta que al ser trigrama se comenzará desde la tercera posición en lugar de la segunda, y las nuevas transiciones:

$$\forall t \;=\; 3 \dots \mathrm{T} \text{ and } \forall\, s \in \mathcal{Y}$$

$$\alpha_{\mathrm{t}}(s) = \sum_{s', s'' \in \mathcal{Y}} \alpha_{\mathrm{t}-1}(s') \;\cdot\; \Psi_t(y_{t-2} = s'', y_{t-1} = s', y_t = s, x_t)$$

Y, por último, hay que destacar que la expresión del resultado final no varía en esta versión del algoritmo.

2. Cuestión práctica

En este último apartado práctico de la tarea se propone, a través del toolkit SCFG, evaluar el rendimiento de una tarea de clasificación de triángulos en tres clases: isósceles, escaleno y equilátero. Para ello se contará con tres modelos a evaluar (G1, G2 y G3) para cada una de las clases y un conjunto de test o corpus para cada una de estas.

2.1 Evaluación estadística

La primera parte se centra en la evaluación estadística, para lo que se tomará como valores para la evaluación la log-verosimilitud, ya que, como cita el boletín, es la mejor opción. Esta parte se llevará a cabo con la ayuda de los scripts proporcionados para la práctica, en concreto, scfg.prob, y el desarrollado en Python que será adjuntado con el documento.

De esta manera se obtienen los siguientes valores para las distintas combinaciones de modelo-corpus, que serán divididos en función del modelo original (G1, G2 y G3):

Modelo	**Corpus**	**Log-verosimilitud**
G1-EQ	TS-EQ	-11.503,83
G1-EQ	TS-IS	-5.256,88
G1-EQ	TS-SC	-5.504,97
G1-IS	TS-EQ	-10.765,00
G1-IS	TS-IS	-10.187,98
G1-IS	TS-SC	-9.981,46
G1-SC	TS-EQ	-10.786,62
G1-SC	TS-IS	-10.222,26
G1-SC	TS-SC	-10.422,82

Tabla 1 – Valores de Log-Verosimilitud del modelo G1

Modelo	Corpus	Log-verosimilitud
G2-EQ	TS-EQ	-13.361,59
G2-EQ	TS-IS	-12.872,67
G2-EQ	TS-SC	-13.162,01
G2-IS	TS-EQ	-11.537,60
G2-IS	TS-IS	-10.863,17
G2-IS	TS-SC	-10.815,48
G2-SC	TS-EQ	-11.129,50
G2-SC	TS-IS	-10.662,49
G2-SC	TS-SC	-10.681,51

Tabla 2 – Valores de Log-Verosimilitud del modelo G2

Modelo	Corpus	Log-verosimilitud
G3-EQ	TS-EQ	-6.893,86
G3-EQ	TS-IS	-13.218,05
G3-EQ	TS-SC	-13.894,30
G3-IS	TS-EQ	-7.157,25
G3-IS	TS-IS	-7.134,80
G3-IS	TS-SC	-7.105,57
G3-SC	TS-EQ	-7.148,91
G3-SC	TS-IS	-7.137,55
G3-SC	TS-SC	-7.104,99

Tabla 3 – Valores de Log-Verosimilitud del modelo G3

Puesto que estamos trabajando con log-verosimilitud, los mejores resultados serán aquellos más altos, es decir, buscaremos maximizar este valor.

Inicialmente, podemos ver que el modelo G2 queda eclipsado por el G1, ya que este último tiene valores más altos de log-verosimilitud en todos los casos de prueba que se han llevado a cabo, lo cuál se repite en casi todos los casos del modelo G3 a excepción de un par, por lo que el modelo G2 será el que mayores dificultades presentará.

Si comparamos los otros dos modelos, G1 y G3, se puede observar que el primero presenta bajos valores de log-verosimilitud similares a excepción de los casos G1-EQ con los corpus TS-IS y TS-SC, donde la log-verosimilitud es mucho mayor. Por otro lado, en el segundo modelo, G3, ocurre lo contrario, donde se ven dos casos con una log-verosimilitud muy baja (estos dos que se acaban de mencionar) y el resto de los casos presentan una log-verosimilitud mayor.

A la vista de estos resultados, podríamos decir que el modelo que tendrá mejor comportamiento en este problema será el G3, pues, aunque sus máximos valores de log-verosimilitud no sean tan altos como los del modelo G1, presenta muchos más casos donde este valor es relativamente más alto.

2.2 Clasificación

Finalmente se evaluará el rendimiento de la propia clasificación de los modelos. Para ello, los resultados pueden estudiarse a partir de las matrices de confusión, obtenidas, una vez más, con el script que se adjuntará con este documento y el programa confus que se ha proporcionado.

Al igual que en el apartado anterior, se presentarán los resultados para cada uno de los tres modelos (G1, G2 y G3).

	EQ	**IS**	**SC**	**ERR**	**ERR%**
EQ	597	285	118	403	40.3
IS	88	471	441	529	52.9
SC	71	406	523	477	47.7
Total ERR				1409	46.97

Tabla 4 – Matriz de confusión del modelo G1

	EQ	IS	SC	ERR	ERR%
EQ	281	211	508	719	71.9
IS	71	215	714	785	78.5
SC	81	190	729	271	27.1
Total ERR				1775	59.17

Tabla 5 – Matriz de confusión del modelo G2

	EQ	IS	SC	ERR	ERR%
EQ	789	102	109	211	21.1
IS	178	512	310	488	48.8
SC	106	421	473	527	52.7
Total ERR				1226	40.87%

Tabla 6 – Matriz de confusión del modelo G3

Como se puede ver en las tasas de error de las matrices de confusión de los tres modelos y siguiendo con lo que se había comentado en el apartado anterior de evaluación estadística, el modelo G3 es el que mejores resultados ofrece siendo su tasa de error la más baja, mientras que el modelo G2 es el que mayor tasa de error tiene.

Más en detalle, a partir de la matriz de cada modelo, podemos analizar dónde se producen los errores en cada caso, por ejemplo, a partir de la matriz de confusión del modelo G1 se puede saber que este clasifica erróneamente una gran cantidad de triángulos isósceles confundiéndolos con escalenos y viceversa.

Por otro lado, en el caso del modelo G2, podemos observar que tiende a clasificar la mayoría de los triángulos como escalenos independientemente de que lo sean o no, por lo que obtiene una muy buena tasa de error para esta clase para muy mala para los otros dos, dando en conjunto los peores resultados de los tres modelos.

Por último, en la matriz de confusión del modelo G3, podemos ver que es similar al primer caso, ya que también confunde los triángulos isósceles con escalenos, pero con un error ligeramente inferior, mientras que el error es mayor al confundir escalenos con isósceles. No obstante, el rendimiento es muy bueno al clasificar los equiláteros, ofreciendo los mejores resultados globales de los tres modelos.