Capítulo 5

Aprendizaje de etiquetas basado en transformaciones

Tradicionalmente, en lo que se refiere a la etiquetación de textos en lenguaje natural, y tal y como hemos visto en el capítulo anterior, se han preferido las aproximaciones puramente estocásticas frente a las aproximaciones basadas en reglas, debido a su buen rendimiento y sobre todo a sus facilidades de entrenamiento automático. Sin embargo, hemos visto también que algunas de las hipótesis de funcionamiento de los modelos de Markov no se adaptan del todo bien a las propiedades sintácticas de los lenguajes naturales. Debido a esto, inmediatamente surge la idea de utilizar modelos más sofisticados. Podríamos pensar por ejemplo en establecer condiciones que relacionen las nuevas etiquetas, no sólo con las etiquetas precedentes, sino también con las palabras precedentes. Podríamos pensar también en utilizar un contexto mayor que el de los etiquetadores basados en trigramas. Pero la mayoría de estas aproximaciones no tienen cabida dentro de los modelos de Markov, debido a la carga computacional que implican y a la gran cantidad de nuevos parámetros que necesitaríamos estimar. Incluso con los etiquetadores basados en trigramas hemos visto que es necesario aplicar técnicas de suavización e interpolación, ya que la estimación de máxima verosimilitud por sí sola no es lo suficientemente robusta.

Eric Brill presentó un sistema de etiquetación basado en reglas, el cual, a partir de un corpus de entrenamiento, infiere automáticamente las reglas de transformación [Brill 1993b], salvando así la principal limitación de este tipo de técnica que es precisamente el problema de cómo obtener dichas reglas. El etiquetador de Brill alcanza un rendimiento comparable al de los etiquetadores estocásticos y, a diferencia de éstos, la información lingüística no se captura de manera indirecta a través de grandes tablas de probabilidades, sino que se codifica directamente bajo la forma de un pequeño conjunto de reglas no estocásticas muy simples, pero capaces de representar interdependencias muy complejas entre palabras y etiquetas. Este capítulo describe las características principales del etiquetador de Brill y de su principio de funcionamiento: un paradigma de aprendizaje basado en transformaciones y dirigido por el error[1]. Veremos que este método es capaz de explorar un abanico mayor de propiedades, tanto léxicas como sintácticas, de los lenguajes naturales. En particular, se pueden relacionar etiquetas con palabras concretas, se puede ampliar el contexto precedente, e incluso se puede utilizar el contexto posterior.

5.1 Arquitectura interna del etiquetador de Brill

El etiquetador de Brill consta de tres partes, que se infieren automáticamente a partir de un corpus de entrenamiento: un etiquetador léxico, un etiquetador de palabras desconocidas, y un

[1] *Transformation-based error-driven learning.*

etiquetador contextual. A continuación se describe detalladamente cada una de ellas.

5.1.1 El etiquetador léxico

El etiquetador léxico etiqueta inicialmente cada palabra con su etiqueta más probable, sin tener en cuenta el contexto en el que dicha palabra aparece. Dicha etiqueta más probable se estima previamente mediante el estudio del corpus de entrenamiento. A las palabras desconocidas se les asigna en un primer momento la etiqueta correspondiente a sustantivo propio si la primera letra es mayúscula, o la correspondiente a sustantivo común en otro caso. Posteriormente, el etiquetador de palabras desconocidas aplica en orden una serie de reglas de transformación léxicas.

Si se dispone de un diccionario previamente construido, es posible utilizarlo junto con el que el etiquetador de Brill genera automáticamente. Más adelante veremos cómo se realiza esta integración y en qué circunstancias concretas este proceso ayuda a mejorar el rendimiento del etiquetador.

5.1.2 El etiquetador de palabras desconocidas

El etiquetador de palabras desconocidas opera justo después de que el etiquetador léxico haya etiquetado todas las palabras presentes en el diccionario, y justo antes de que se apliquen las reglas contextuales. Este módulo intenta *adivinar* una etiqueta para una palabra desconocida en función de su sufijo[2], de su prefijo, y de otras propiedades relevantes similares.

Básicamente, cada transformación consta de dos partes: una descripción del contexto de aplicación, y una regla de reescritura que reemplaza una etiqueta por otra. La plantilla genérica de transformaciones léxicas es la siguiente:

- `x haspref l A`: si los primeros `l` caracteres de la palabra son `x`, se asigna a la palabra desconocida la etiqueta `A`
- `A x fhaspref l B`: si la etiqueta actual de la palabra es `A` y sus primeros `l` caracteres son `x`, se cambia dicha etiqueta por `B`
- `x deletepref l A`: si borrando el prefijo `x` de longitud `l` obtenemos una palabra conocida, se asigna a la palabra desconocida la etiqueta `A`
- `A x fdeletepref l B`: si la etiqueta actual de la palabra es `A` y borrando el prefijo `x` de longitud `l` obtenemos una palabra conocida, se cambia dicha etiqueta por `B`
- `x addpref l A`: si añadiendo el prefijo `x` de longitud `l` obtenemos una palabra conocida, se asigna a la palabra desconocida la etiqueta `A`
- `A x faddpref l B`: si la etiqueta actual de la palabra es `A` y añadiendo el prefijo `x` de longitud `l` obtenemos una palabra conocida, se cambia dicha etiqueta por `B`
- `x hassuf l A`: si los últimos `l` caracteres de la palabra son `x`, se asigna a la palabra desconocida la etiqueta `A`
- `A x fhassuf l B`: si la etiqueta actual de la palabra es `A` y sus últimos `l` caracteres son `x`, se cambia dicha etiqueta por `B`
- `x deletesuf l A`: si borrando el sufijo `x` de longitud `l` obtenemos una palabra conocida, se asigna a la palabra desconocida la etiqueta `A`
- `A x fdeletesuf l B`: si la etiqueta actual de la palabra es `A` y borrando el sufijo `x` de longitud `l` obtenemos una palabra conocida, se cambia dicha etiqueta por `B`

[2] Por ejemplo, en inglés, una palabra terminada en `ing` podría etiquetarse como un verbo en gerundio.

`x addsuf l A`: si añadiendo el sufijo `x` de longitud `l` obtenemos una palabra conocida, se asigna a la palabra desconocida la etiqueta `A`

`A x faddsuf l B`: si la etiqueta actual de la palabra es `A` y añadiendo el sufijo `x` de longitud `l` obtenemos una palabra conocida, se cambia dicha etiqueta por `B`

`w goodright A`: si la palabra aparece inmediatamente a la derecha de la palabra `w`, se asigna a la palabra desconocida la etiqueta `A`

`A w fgoodright B`: si la etiqueta actual de la palabra es `A` y aparece inmediatamente a la derecha de la palabra `w`, se cambia dicha etiqueta por `B`

`w goodleft A`: si la palabra aparece inmediatamente a la izquierda de la palabra `w`, se asigna a la palabra desconocida la etiqueta `A`

`A w fgoodleft B`: si la etiqueta actual de la palabra es `A` y aparece inmediatamente a la izquierda de la palabra `w`, se cambia dicha etiqueta por `B`

`z char A`: si el caracter `z` aparece en la palabra, se asigna a la palabra desconocida la etiqueta `A`

`A z fchar B`: si la etiqueta actual de la palabra es `A` y el caracter `z` aparece en la palabra, se cambia dicha etiqueta por `B`

donde `A` y `B` son variables sobre el conjunto de todas las etiquetas, `x` es cualquier cadena de caracteres de longitud 1, 2, 3 o 4, `l` es la longitud de dicha cadena, `w` es cualquier palabra, y `z` es cualquier caracter.

Ejemplo 5.1 A continuación se muestran algunas de las reglas de transformación léxicas más comunes que el etiquetador de Brill encontró para el español:

`rse hassuf 3 V000f0PE1`: si los últimos 3 caracteres de la palabra son `rse`, se asigna a la palabra desconocida la etiqueta `V000f0PE1`, es decir, verbo infinitivo con un pronombre enclítico

`r hassuf 1 V000f0`: si el último caracter de la palabra es `r`, se asigna a la palabra desconocida la etiqueta `V000f0`, es decir, verbo infinitivo

`V000f0 or fhassuf 2 Scms`: si la etiqueta actual de la palabra es `V000f0`, es decir, verbo infinitivo, y sus últimos 2 caracteres son `or`, se cambia dicha etiqueta por la etiqueta `Scms`, es decir, sustantivo común, masculino, singular

`ría deletesuf 3 Vysci0`: si borrando el sufijo `ría` de longitud 3 obtenemos una palabra conocida, se asigna a la palabra desconocida la etiqueta `Vysci0`, es decir, verbo, primera y tercera personas del singular, postpretérito de indicativo

`Scfs r faddsuf 1 V3spi0`: si la etiqueta actual de la palabra es `Scfs`, es decir, sustantivo, común, femenino, singular, y añadiendo el sufijo `r` de longitud 1 obtenemos una palabra conocida, se cambia dicha etiqueta por la etiqueta `V3spi0`, es decir, verbo, tercera persona del singular, presente de indicativo

`el goodright Scms`: si la palabra aparece inmediatamente a la derecha de la palabra `el`, se asigna a la palabra desconocida la etiqueta `Scms`, es decir, sustantivo común, masculino, singular

`Scmp las fgoodright Scfp`: si la etiqueta actual de la palabra es `Scmp`, es decir, sustantivo común, masculino, plural, y aparece inmediatamente a la derecha de la palabra `las`, se cambia dicha etiqueta por la etiqueta `Scfp`, es decir, sustantivo común femenino, plural

`% goodleft Ncyyp`: si la palabra aparece inmediatamente a la izquierda de la palabra `%`, se asigna a la palabra desconocida la etiqueta `Ncyyp`, es decir, numeral cardinal, determinante y no determinante, masculino y femenino, plural

`w char Ze00`: si el caracter `w` aparece en la palabra, se asigna a la palabra desconocida la etiqueta `Ze00`, es decir, palabra extranjera

Esas reglas fueron generadas por el etiquetador de Brill después de ser entrenado con una porción de texto en español procedente del corpus ITU[3]. □

Por debajo del formato general de las reglas léxicas propuestas por Brill subyace un estudio lingüístico muy importante. Es por ello que esta plantilla genérica de transformaciones léxicas representa una manera muy elegante de integrar el manejo de palabras desconocidas dentro de una herramienta general de etiquetación, y la hace independiente del idioma a tratar. Pero considerando el caso concreto del español, se echan de menos fenómenos muy comunes tales como el sufijo `mente` para formar adverbios a partir de los adjetivos. En el corpus ITU existe un buen número de palabras que presentan dicha característica, pero ésta nunca podrá aparecer reflejada en una regla debido a la limitación de 4 caracteres de longitud en los prefijos y sufijos de las reglas extraídas automáticamente. No obstante, el etiquetador de Brill proporciona al usuario la posibilidad de añadir manualmente nuevas reglas después del entrenamiento.

Otras características relevantes para el tratamiento de palabras desconocidas en modelos de etiquetación basados en reglas fueron estudiadas por Mikheev, quien no sólo amplía el formalismo de reglas léxicas del etiquetador de Brill, sino que también propone su propio algoritmo para la inducción automática de dichas reglas [Mikheev 1997].

5.1.3 El etiquetador contextual

El etiquetador contextual actúa justo después del etiquetador de palabras desconocidas, aplicando en orden una secuencia de reglas contextuales que, al igual que las léxicas, también han sido previamente inferidas de manera automática a partir del corpus de entrenamiento. La plantilla genérica de transformaciones contextuales es la siguiente:

`A B prevtag C`: cambiar la etiqueta `A` por `B` si la palabra anterior aparece etiquetada con la etiqueta `C`

`A B prev1or2tag C`: cambiar la etiqueta `A` por `B` si una de las dos palabras anteriores aparece etiquetada con la etiqueta `C`

`A B prev1or2or3tag C`: cambiar la etiqueta `A` por `B` si una de las tres palabras anteriores aparece etiquetada con la etiqueta `C`

`A B prev2tag C`: cambiar la etiqueta `A` por `B` si la segunda palabra anterior aparece etiquetada con la etiqueta `C`

`A B nexttag C`: cambiar la etiqueta `A` por `B` si la palabra siguiente aparece etiquetada con la etiqueta `C`

`A B next1or2tag C`: cambiar la etiqueta `A` por `B` si una de las dos palabras siguientes aparece etiquetada con la etiqueta `C`

`A B next1or2or3tag C`: cambiar la etiqueta `A` por `B` si una de las tres palabras siguientes aparece etiquetada con la etiqueta `C`

`A B next2tag C`: cambiar la etiqueta `A` por `B` si la segunda palabra siguiente aparece etiquetada con la etiqueta `C`

[3] *International Telecommunications Union CCITT Handbook* (véase la sección 2.1).

- A B **prevbigram** C D: cambiar la etiqueta A por B si la palabra anterior aparece etiquetada con la etiqueta C y la segunda palabra anterior con D
- A B **nextbigram** C D: cambiar la etiqueta A por B si la palabra siguiente aparece etiquetada con la etiqueta C y la segunda palabra siguiente con D
- A B **surroundtag** C D: cambiar la etiqueta A por B si la palabra anterior aparece etiquetada con la etiqueta C y la siguiente con D
- A B **curwd** w: cambiar la etiqueta A por B si la palabra actual es w
- A B **prevwd** w: cambiar la etiqueta A por B si la palabra anterior es w
- A B **prev1or2wd** w: cambiar la etiqueta A por B si una de las dos palabra anteriores es w
- A B **prev2wd** w: cambiar la etiqueta A por B si la segunda palabra anterior es w
- A B **nextwd** w: cambiar la etiqueta A por B si la palabra siguiente es w
- A B **next1or2wd** w: cambiar la etiqueta A por la etiqueta B si una de las dos palabras siguientes es w
- A B **next2wd** w: cambiar la etiqueta A por B si la segunda palabra siguiente es w
- A B **lbigram** w x: cambiar la etiqueta A por B si las dos palabras anteriores son w y x
- A B **rbigram** w x: cambiar la etiqueta A por B si las dos palabras siguientes son w y x
- A B **wdand2bfr** x w: cambiar la etiqueta A por B si la palabra actual es w y la segunda palabra anterior es x
- A B **wdand2aft** w x: cambiar la etiqueta A por B si la palabra actual es w y la segunda palabra siguiente es x
- A B **wdprevtag** C w: cambiar la etiqueta A por B si la palabra actual es w y la anterior aparece etiquetada con la etiqueta C
- A B **wdnexttag** w C: cambiar la etiqueta A por B si la palabra actual es w y la siguiente aparece etiquetada con la etiqueta C
- A B **wdand2tagbfr** C w: cambiar la etiqueta A por B si la palabra actual es w y la segunda palabra anterior aparece etiquetada con la etiqueta C
- A B **wdand2tagaft** w C: cambiar la etiqueta A por B si la palabra actual es w y la segunda palabra siguiente aparece etiquetada con la etiqueta C

donde A, B, C y D son variables sobre el conjunto de todas las etiquetas, y w y x son cualquier palabra. La figura 5.1 resume gráficamente los posibles contextos que se pueden tener en cuenta a la hora de aplicar una transformación. La palabra a etiquetar es siempre la de la posición i, que aparece marcada con un asterisco. Los recuadros indican cuáles son las posiciones relevantes de cada esquema contextual. Si en una posición dada aparece un recuadro normal, el esquema considera la etiqueta de la palabra que está en esa posición. Si aparece un recuadro sombreado, el esquema considera la palabra concreta. Por ejemplo, la regla A B **prevbigram** C D responde al esquema 3, mientras que la regla A B **lbigram** w x utiliza el esquema 15.

Ejemplo 5.2 A continuación se muestran algunas de las reglas de transformación contextuales más comunes que el etiquetador de Brill encontró para el español, también a partir de una porción de texto del corpus ITU:

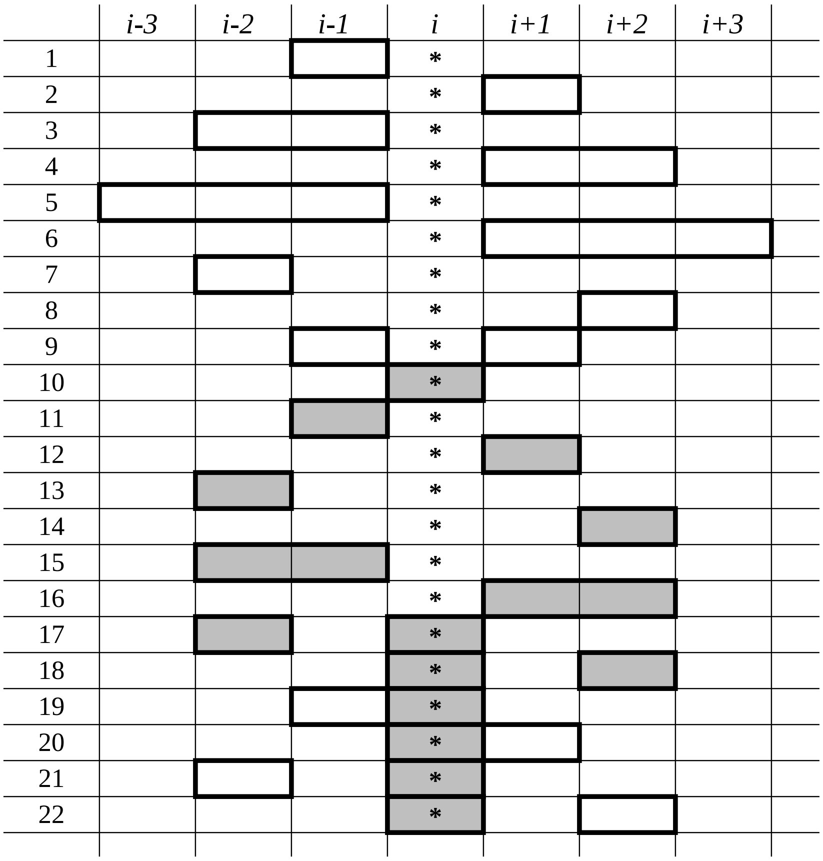

Figura 5.1: Esquemas contextuales de las reglas del etiquetador de Brill

`Afp0 Amp0 prevtag Scmp`: cambiar la etiqueta `Afp0`, es decir, adjetivo, femenino, plural, sin grado, por `Amp0`, es decir, adjetivo, masculino, plural, sin grado, si la palabra anterior aparece etiquetada con la etiqueta `Scmp`, es decir, sustantivo común, masculino, plural

`Scms Ams0 wdprevtag Scms receptor`: cambiar la etiqueta `Scms`, es decir, sustantivo común, masculino, singular, por `Ams0`, es decir, adjetivo, masculino, singular, sin grado, si la palabra actual es `receptor` y la anterior aparece etiquetada con la etiqueta `Scms`

`Scms Ams0 wdand2bfr el transmisor`: cambiar la etiqueta `Scms`, es decir, sustantivo común, masculino, singular, por `Ams0`, es decir, adjetivo, masculino, singular, sin grado, si la palabra actual es `transmisor` y la segunda palabra anterior es `el`

`P Scms nexttag P`: cambiar la etiqueta `P`, es decir, preposición, por `Scms`, es decir, sustantivo común, masculino, singular, si la palabra siguiente aparece etiquetada con la etiqueta `P`

A través de estos ejemplos, se puede apreciar que el formalismo de reglas contextuales del etiquetador de Brill constituye un sencillo pero potente mecanismo, que es capaz de resolver las concordancias de género y número dentro de una misma categoría, las ambigüedades entre distintas categorías, como por ejemplo, adjetivo/sustantivo, y hasta puede evitar que aparezcan dos preposiciones seguidas. □

5.2 Aprendizaje basado en transformaciones y dirigido por el error

El proceso de generación de las reglas, tanto las léxicas en el caso del etiquetador de palabras desconocidas, como las contextuales en el caso del etiquetador contextual, selecciona el mejor conjunto de transformaciones y determina su orden de aplicación. El algoritmo consta de los pasos que se describen a continuación. En primer lugar, se toma una porción de texto no etiquetado, se pasa a través de la fase o fases de etiquetación anteriores, se compara la salida con el texto correctamente etiquetado, y se genera una lista de errores de etiquetación con sus correspondientes contadores. Entonces, para cada error, se determina qué instancia concreta de la plantilla genérica de reglas produce la mayor reducción de errores. Se aplica la regla, se calcula el nuevo conjunto de errores producidos, y se repite el proceso hasta que la reducción de errores cae por debajo de un umbral dado.

La figura 5.2 ilustra gráficamente este procedimiento, que es el que da nombre a la técnica de entrenamiento desarrollada por Brill: aprendizaje basado en transformaciones y dirigido por el error. El usuario puede especificar los umbrales de error antes del entrenamiento, y puede también añadir manualmente nuevas reglas de transformación después del mismo.

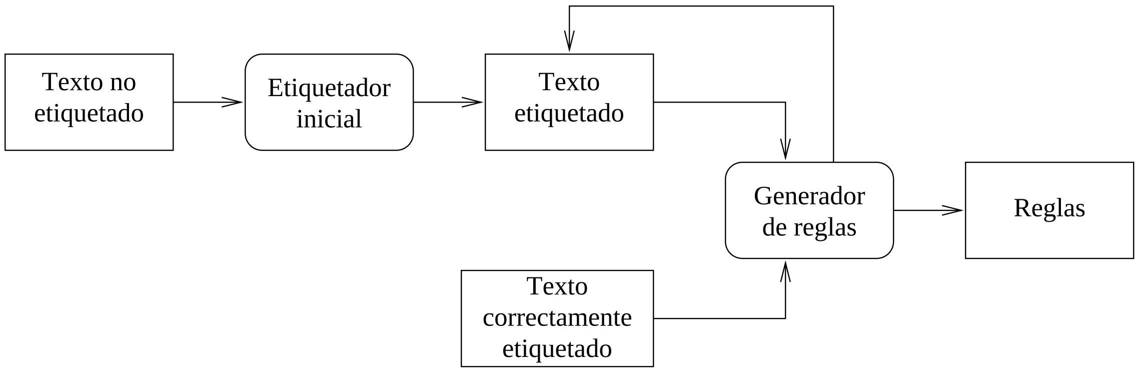

Figura 5.2: Aprendizaje basado en transformaciones y dirigido por el error

A las características anteriormente descritas podríamos añadir también una funcionalidad introducida posteriormente por el propio Brill, la cual permite obtener las k etiquetas más probables de una palabra[4] [Brill 1994], para ciertas aplicaciones en las que es posible relajar la restricción de una sola etiqueta por palabra. Brill implementa esta nueva funcionalidad mediante un sencillo cambio en el formato de las reglas:

> la acción *cambiar la etiqueta* `A` *por la etiqueta* `B` se transforma en *añadir la etiqueta* `A` *a la etiqueta* `B` o *añadir la etiqueta* `A` *a la palabra* `w`.

De esta manera, en lugar de reemplazar etiquetas, las reglas de transformación permiten ahora añadir etiquetas alternativas a una palabra. Sin embargo, el problema es que el etiquetador no nos proporciona información sobre la probabilidad de cada etiqueta. Es decir, si consideramos por ejemplo las dos mejores etiquetas para una palabra dada, la única conclusión que podemos extraer es que la que aparece en primer lugar es más probable que la que aparece en segundo lugar, pero bien podría ocurrir tanto que la primera fuera 100 veces más probable que la segunda, como que ambas fueran igualmente probables. La cuestión es que este tipo de información podría ser crucial para algunas aplicaciones, por ejemplo para la construcción de un análisis sintáctico. Los etiquetadores puramente estocásticos sí son capaces de proporcionar estas cifras y además lo hacen sin ningún esfuerzo computacional extra.

[4] *k-Best tags.*

5.3 Complejidad del etiquetador de Brill

La implementación original de Brill resulta considerablemente más lenta que las basadas en modelos probabilísticos. No sólo el proceso de entrenamiento consume muchísimo tiempo, tal y como veremos en el capítulo 7, sino que el proceso de etiquetación es también inherentemente lento. La principal razón de esta ineficiencia computacional es la potencial interacción entre las reglas, de manera que el algoritmo puede producir cálculos innecesarios.

Ejemplo 5.3 Si suponemos que `VBN` y `VBD` son las etiquetas más probables para las palabras `killed` y `shot`, respectivamente, el etiquetador léxico podría asignar las siguientes etiquetas[5]:

```
(1) Chapman/NP killed/VBN John/NP Lennon/NP
(2) John/NP Lennon/NP was/BEDZ shot/VBD by/BY Chapman/NP
(3) He/PPS witnessed/VBD Lennon/NP killed/VBN by/BY Chapman/NP
```

Dado que el etiquetador léxico no utiliza ninguna información contextual, muchas palabras pueden aparecer etiquetadas incorrectamente. Por ejemplo, en (1) la palabra `killed` aparece etiquetada incorrectamente como verbo en participio pasado, y en (2) `shot` aparece incorrectamente etiquetada como verbo en tiempo pasado.

Una vez obtenida la etiquetación inicial, el etiquetador contextual aplica en orden una secuencia de reglas e intenta remediar los errores cometidos. En un etiquetador contextual podríamos encontrar reglas como las siguientes:

```
VBN VBD prevtag NP
VBD VBN nexttag BY
```

La primera regla dice: *cambiar la etiqueta* `VBN` *por* `VBD` *si la etiqueta previa es* `NP`. La segunda regla dice: *cambiar* `VBD` *por* `VBN` *si la siguiente etiqueta es* `BY`. Una vez que aplicamos la primera regla, la palabra `killed` que aparece en las frases (1) y (3) cambia su etiqueta `VBN` por `VBD`, y obtenemos las siguientes etiquetaciones:

```
(4) Chapman/NP killed/VBD John/NP Lennon/NP
(5) John/NP Lennon/NP was/BEDZ shot/VBD by/BY Chapman/NP
(6) He/PPS witnessed/VBD Lennon/NP killed/VBD by/BY Chapman/NP
```

Una vez que aplicamos la segunda regla, la palabra `shot` de la frase (5) cambia su etiqueta `VBD` por `VBN`, generando la etiquetación (8), y la palabra `killed` de la frase (6) vuelve a cambiar su etiqueta `VBD` otra vez por `VBN`, y obtenemos la etiquetación (9):

```
(7) Chapman/NP killed/VBD John/NP Lennon/NP
(8) John/NP Lennon/NP was/BEDZ shot/VBN by/BY Chapman/NP
(9) He/PPS witnessed/VBD Lennon/NP killed/VBN by/BY Chapman/NP
```

Hemos visto que una regla `nextag` necesita mirar un *token* hacia adelante en la frase antes de poder ser aplicada, y hemos visto también que la aplicación de dos o más reglas puede producir una serie de operaciones que no se traducen en ningún cambio neto. Estos dos fenómenos constituyen la causa del no determinismo local del etiquetador de Brill. □

Este problema fue abordado por Roche y Schabes, quienes propusieron un sistema que codifica las reglas de transformación del etiquetador bajo la forma de un traductor de estado finito determinista [Roche y Schabes 1995]. El algoritmo de construcción de dicho traductor consta de cuatro pasos:

[5] La notación de las etiquetas es una adaptación del juego de etiquetas utilizado en el corpus BROWN [Francis y Kučera 1982]: `VBN` significa verbo en participio pasado, `VBD` es verbo en tiempo pasado, `NP` es sustantivo propio, `BEDZ` es la palabra `was`, `BY` es la palabra `by` y `PPS` es pronombre nominativo singular en tercera persona.

1. En primer lugar, cada transformación se convierte en un traductor de estado finito.

2. El segundo paso consiste en convertir cada traductor a su extensión local. La extensión local t_2 de un traductor t_1 se construye de tal manera que procesar una cadena de entrada a través de t_2 en un solo paso produce el mismo efecto que procesar cada posición de la cadena de entrada a través de t_1. Este paso se ocupa de casos como el siguiente. Supongamos un traductor que implementa la transformación *cambiar* `A` *por* `B` *si una de las dos etiquetas precedentes es* `C`. Este traductor contendrá un arco que transforma el símbolo de entrada `A` en el símbolo de salida `B`, de tal manera que dada la secuencia de entrada `CAA` tenemos que aplicarlo dos veces, en la segunda y en la tercera posiciones, para transformarla correctamente en `CBB`. La extensión local es capaz de realizar dicha conversión en un solo paso.

3. En el tercer paso, se genera un único traductor cuya aplicación tiene el mismo efecto que la aplicación de todos los traductores individuales en secuencia. En general, este traductor único es no determinista. Cuando necesita recordar un evento tal como `C` *apareció en la posición* i, lo hace lanzando dos caminos distintos: uno en el cual se supone que aparecerá una etiqueta que estará afectada por esa `C` precedente, y otro en el que se supone que tal etiqueta no aparecerá.

4. Este tipo de indeterminismo no es eficiente, de ahí que el cuarto paso se ocupe de transformar el traductor no determinista en uno determinista. Esto en general no es posible, ya que los traductores no deterministas pueden recordar eventos de longitud arbitraria y los deterministas no. Sin embargo, Roche y Schabes demuestran que las reglas que aparecen en los etiquetadores basados en transformaciones no generan traductores con esta propiedad. Por tanto, en la práctica siempre es posible transformar un etiquetador basado en transformaciones en un traductor de estado finito determinista.

Por tanto, el algoritmo de Brill podría necesitar RKn pasos elementales para etiquetar una cadena de entrada de n palabras, con R reglas aplicables en un contexto de hasta K *tokens*. Con el traductor de estado finito propuesto por Roche y Schabes, para etiquetar una frase de longitud n palabras, se necesitan sólo n pasos, independientemente del número de reglas y de la longitud del contexto que éstas utilizan. Esto significa que el proceso de etiquetación añade a la lectura del texto de entrada una carga computacional que es despreciable en comparación con tratamientos posteriores tales como los análisis sintáctico y semántico. Con los etiquetadores basados en traductores se pueden llegar a obtener velocidades de etiquetación de varias decenas de miles de palabras por segundo, mientras que con los etiquetadores basados en modelos de Markov esa velocidad puede ser de un orden de magnitud menos. Existen trabajos que estudian la transformación de modelos de Markov ocultos en traductores de estado finito [Kempe 1997], pero en este caso no se puede alcanzar una equivalencia completa ya que los autómatas no pueden simular de una manera exacta los cálculos de punto flotante involucrados en el algoritmo de Viterbi.

5.4 Relación con otros modelos de etiquetación

Se han esbozado ya algunas de las diferencias conceptuales más importantes que existen entre el etiquetador de Brill y los etiquetadores puramente estocásticos. Esta sección completa un poco más el estudio comparativo de los principios de funcionamiento de éstas aproximaciones y de otras relacionadas[6]. Finalmente, se citan otros posibles campos de aplicación en los que el

[6] El estudio analítico completo de los rendimientos de cada paradigma en el proceso de etiquetación será abordado en el capítulo 7.

aprendizaje basado en transformaciones ha funcionado también con éxito.

5.4.1 Árboles de decisión

El aprendizaje basado en transformaciones presenta algunas similitudes con los árboles de decisión[7]. Un árbol de decisión [Schmid 1994, Brown *et al.* 1991] se puede ver como un mecanismo que etiqueta todas las hojas dominadas por un nodo con la etiqueta de la clase mayoritaria de ese nodo. Posteriormente, a medida que descendemos por el árbol, reetiquetamos las hojas de los nodos hijos, si es que difieren de la etiqueta del nodo padre, en función de las respuestas a las cuestiones o decisiones que aparecen en cada nodo. Esta manera de ver los árboles de decisión es la que muestra el parecido con el aprendizaje basado en transformaciones, ya que ambos paradigmas realizan series de reetiquetados trabajando con subconjuntos de datos cada vez más pequeños.

En principio, el aprendizaje basado en transformaciones es más potente que los árboles de decisión [Brill 1995a]. Es decir, existen tareas de clasificación que se pueden resolver con el aprendizaje basado en transformaciones, pero no con los árboles de decisión. Sin embargo, no está muy claro si este tipo de potencia extra se utiliza o no en aplicaciones de procesamiento de lenguaje natural.

La principal diferencia entre estos dos modelos es que los datos de entrenamiento se dividen en cada nodo de un árbol de decisión, y que se aplica una secuencia de *transformaciones* distintas para cada nodo: la secuencia correspondiente a las decisiones del camino que va desde la raíz hasta ese nodo. Con el aprendizaje basado en transformaciones, cada transformación de la lista de transformaciones *aprendidas* se aplica a todo el texto, generando una reescritura cuando el contexto de los datos encaja con el de la regla. Como resultado, si minimizamos en función de los errores de etiquetación cometidos, en lugar de considerar otro tipo de medidas indirectas más comunes en el caso de los árboles de decisión, tales como la entropía, entonces sería relativamente sencillo alcanzar el 100% de precisión en cada nodo hoja. Sin embargo, el rendimiento sobre textos nuevos sería muy pobre debido a que cada nodo hoja estaría formado por un conjunto de propiedades totalmente arbitrarias, que aunque han sido extraídas de los datos de entrenamiento no son completamente generales.

Sorpredentemente, el aprendizaje basado en transformaciones parece ser inmune a este fenómeno [Ramshaw y Marcus 1994]. Esto se puede explicar parcialmente por el hecho de que el entrenamiento siempre se realiza sobre todo el conjunto de datos. Pero el precio que hay que pagar para obtener este tipo de robustez es que el espacio de secuencias de transformaciones debe ser grande. Una implementación *naive* del aprendizaje basado en transformaciones sería por tanto ineficiente. No obstante, existen maneras inteligentes de realizar las búsquedas en ese espacio [Brill 1995b].

5.4.2 Modelos probabilísticos en general

La gran ventaja de la etiquetación basada en transformaciones es que se pueden establecer decisiones sobre un conjunto de propiedades más rico que en el caso de los modelos puramente estocásticos. Por ejemplo, se puede utilizar simultáneamente información de los contextos izquierdo y derecho, y las palabras concretas, no sólo sus etiquetas, pueden influir en la etiquetación de las palabras vecinas.

Otro punto clave es que las reglas de transformación son más fáciles de entender y de modificar que las probabilidades de transición y de generación de palabras en los etiquetadores probabilísticos. Sin embargo, está claro también que es más difícil prever el efecto que puede

[7] También denominados *decision trees*.

llegar a tener la modificación de una regla dentro de una secuencia de aplicación, ya que el comportamiento de cada regla depende de la ejecución de las reglas previas y pueden surgir numerosas y complejas interacciones entre ellas.

Es importante también señalar que la etiquetación basada en transformaciones es claramente un método estadístico. Es decir, aunque no se hace un uso explícito de la teoría probabilística y aunque existe un componente basado en reglas, estas reglas se generan de una manera cuantitativa a partir de contadores calculados directamente sobre los textos. Por tanto, se trata de un método no supervisado[8] y de aplicación completamente automatizada. Desde este punto de vista, la única diferencia es que los cálculos con números se realizan sólo durante el proceso de aprendizaje, el cual además no presenta problemas de sobreentrenamiento, y una vez que el aprendizaje está hecho, el proceso de etiquetación resulta ser puramente simbólico y es por ello que se puede implementar en una estructura computacional muy eficiente.

Por último, dado que el rendimiento de los etiquetadores basados en transformaciones va a resultar muy similar al de los puramente estocásticos, la decisión final entre utilizar unos u otros dependerá casi exclusivamente de en qué tipo de sistema va a estar integrado el etiquetador y para qué tipo de aplicaciones se va a utilizar.

Además de al proceso de etiquetación, el aprendizaje basado en transformaciones se ha aplicado también al análisis sintáctico [Brill 1993a, Brill 1993c], al problema de la ligadura de la frase preposicional [Brill y Resnik 1994] y a la eliminación de ambigüedades semánticas [Dini *et al.* 1998].

[8] En el sentido de que lo que hay presente en los textos de entrenamiento son las etiquetas, pero lo que realmente se extrae de ellos son las reglas.