Lingüística Computacional: Trabajo POS-Tagging

Jaime Ferrando Huertas, Javier Martínez Bernia

2020, MIARFID @ UPV

Contents

1 Introducción

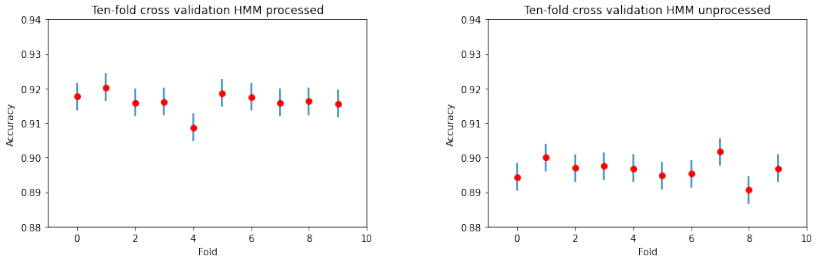
En lingüística computacional, el etiquetado gramatical (part-of-speech tagging, POS tagging) consiste en asignar a cada una de las palabras de tu texto con su categoría gramatical. Este proceso se puede realizar basándonos en la detección de la palabra o el contexto al que pertenece, las relaciones con palabras próximas en la misma frase o párrafo. Para esto se suelen usar algoritmos que realizan el etiquetado mediante etiquetas descriptivas previamente definidas.

En esta memoria se describe el trabajo realizado en las prácticas de la asignatura Lingüística Computacional sobre POS tagging, donde encontramos un análisis de distintas técnicas existentes para afrontar el problema.

2 Tarea 1

En esta tarea se ha realizado una evaluación del etiquetador 'hmm' sobre el corpus 'cess-esp' procesado y sin procesar. El procesado ha consistido en la reducción de las etiquetas del mismo mediante las reglas descritas en el laboratorio. El corpus original tiene 289 etiquetas y el procesado 66. La evaluación se ha hecho mediante validación cruzada en 10 bloques y se ha barajado el corpus antes de hacer las divisiones. En las gráficas de la figura 1 podemos observar los resultados obtenidos por cada bloque en los dos experimentos.

Como se puede ver en las gráficas, se alcanzan mejores resultados haciendo el procesado del corpus. Mientras que utilizando el corpus sin procesar alcanzamos una media de *accuracy* de 0.8965, procesando el corpus llegamos a una media de *accuracy* de 0.916279 entre los 10 bloques, por lo que la reducción de 289 etiquetas a 66 ha sido beneficiosa. A partir de ahora se usara el corpus procesado para el resto de tareas.



(a) Corpus procesado (b) Corpus sin procesar

Figure 1: Resultados para cada corpus

3 Tarea 2

En la segunda tarea se ha evaluado el etiquetador 'hmm' variando el tamaño del corpus de aprendizaje. Se ha dividido el corpus en 10 particiones, y se ha fijado

la última partición como conjunto de test. A continuación, se ha entrenado el etiquetador incrementando sucesivamente el conjunto de entrenamiento en cada ejecución, utilizando las 9 primeras particiones incrementalmente.

Figure 2: Resultados incrementando los datos de aprendizaje

Como podemos observar en la gráfica 2, el rendimiento del etiquetador mejora sucesivamente si aumentamos el tamaño del conjunto de datos de entrenamiento. La *accuracy* es mejor cuantos más datos de entrenamiento utilizamos. Este resultado era esperable, ya que generalmente cuantos más datos tengamos para entrenar el modelo el rendimiento será mejor.

4 Tarea 3

En la tercera tarea hemos aplicado un método de suavizado basado en sufijos sobre el etiquetador TNT y hemos analizado la repercusión del mismo en base a su longitud. La evaluación se ha hecho mediante validación cruzada en 10 bloques y se ha barajado el corpus antes de hacer las divisiones. En primer lugar, podemos ver la precisión del etiquetador TnT sin aplicar suavizado en la grafica 3.

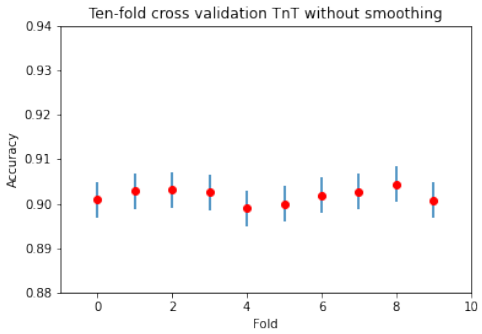


Figure 3: Resultados del etiquetador TnT sin aplicar suavizado

En segundo lugar, se ha estudiado cuál es la mejor longitud de sufijo para el etiquetador Affix Tagger, entrenándolo con distintas longitudes de sufijo. En la siguiente gráfica se puede ver la precisión del etiquetador por longitud de sufijo.

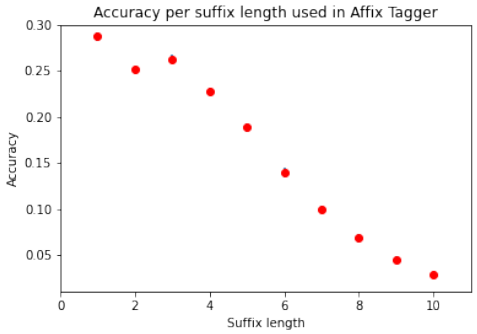


Figure 4: Precisión por longitud de sufijo en Affix Tagger

Como hemos podido observar, la mejor longitud de sufijo es 1, con la cual conseguimos mayor precisión al etiquetar el corpus. Por tanto, a la hora de utilizar TnT, aplicaremos un suavizado para las palabras desconocidas con el etiquetador Affix Tagger con longitud de sufijo igual a 1. En la siguiente gráfica podemos ver cómo mejora la precisión del etiquetador TnT cuando aplicamos este suavizado:

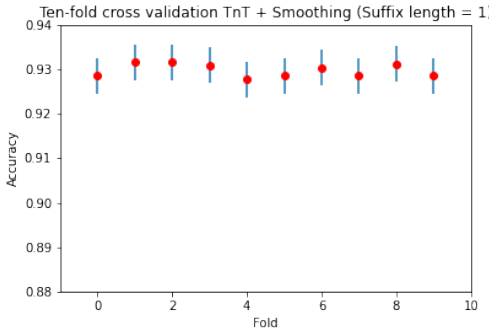


Figure 5: Resultados del etiquetador TnT aplicando suavizado con longitud de sufijo 1

Como hemos observado, la precisión media mejora aplicando esa longitud de sufijo, la cual era la mejor longitud utilizando solamente Affix Tagger. Sin embargo, si combinamos los dos etiquetadores, podemos ver que la mejor longitud de sufijo combinada con TnT es 3. En la siguiente gráfica podemos ver la precisión del etiquetador TnT aplicando suavizado con distinta longitud de sufijo:

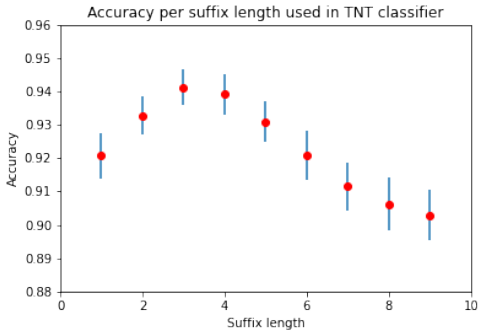


Figure 6: Resultados del etiquetador TnT aplicando suavizado con distintas longitudes de sufijo

Podemos observar como aumenta la precisión en los primeros incrementos de longitud pero tras sobrepasar longitud cuatro la calidad del modelo empeora. Los mejores resultados los encontramos con sufijo de longitud 3. En la siguiente gráfica podemos ver los resultados de la validación cruzada utilizando esta longitud de sufijo:

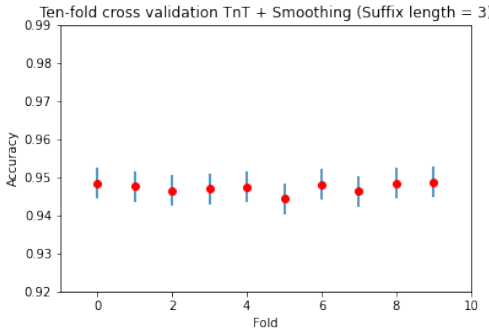


Figure 7: Resultados del etiquetador TnT aplicando suavizado con longitud de sufijo 3.

Finalmente, en la siguiente tabla tenemos un resumen de los resultados obtenidos en esta tarea:

Etiquetador	**Accuracy**
TnT sin suavizado	0.901795
TnT + suavizado sufijo longitud 1	0.929697
TnT + suavizado sufijo longitud 3	0.947277

Table 1: Resultados de los distintos etiquetadores.

Tras ver la tabla, concluimos esta tarea habiendo mejorado el rendimiento del etiquetador TnT en un 5.04% aplicando suavizado.

5 Tarea 4

Para la cuarta tarea hemos comparado el funcionamiento de los etiquetadores usados previamente contra el de tres etiquetadores clásicos presentes en el paquete NLTK, el etiquetador CRF, etiquetador perceptrón y el etiquetador de Brill. En la siguiente gráfica y posterior tabla mostramos su rendimiento evaluado con el método de Hold-out con una división 90/10 entrenamiento/test.

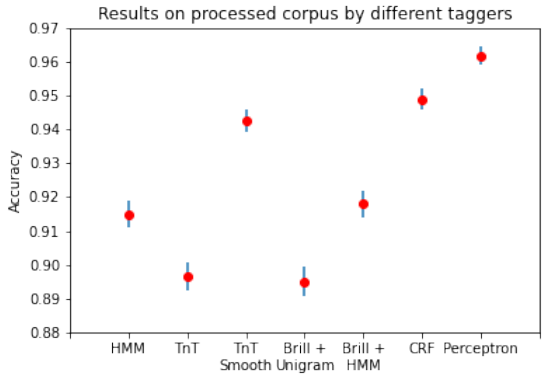


Figure 8: Resultados de los distintos etiquetadores.

Etiquetador	**Accuracy**	**I.C. 95%**
HMM Tagger	0.914798	[0.910837, 0.918759]
TnT Tagger	0.896563	[0.892242, 0.900883]
TnT Tagger + Smooth (suffix=3)	0.942308	[0.939000, 0.945616]
Brill + Unigram Tagger	0.894938	[0.890588, 0.899289]
Brill + HMM Tagger	0.917994	[0.914101, 0.921887]
CRF	0.948753	[0.945624, 0.951881]
Perceptrón	0.961591	[0.958864, 0.964318]

Table 2: Resultados de los distintos etiquetadores.

Podemos observar como el perceptrón obtiene mejor accuracy frente al resto de etiquetadores aunque CRF se sitúa bastante cerca a él, por lo que recomendamos una evaluación mas robusta como validación cruzada para determinar cual funcionaría mejor en nuestro corpus.

6 Tarea 5

Para la última tarea hemos evaluado la experiencia de uso del paquete Freeling en su instalación y para el etiquetado morfosintáctico del fichero *Alicia.txt*. Hemos adjuntado el fichero *outputfreeling.txt* en la entrega de la memoria donde incluimos la salida del etiquetado morfosintáctico del programa para nuestro fichero.

Se ha utilizado la API sobre Python para hacer llamadas al paquete de Freeling en C++ y se ha encontrado bastante intuitiva, al menos con los tutoriales que se enseñan en la página oficial. Lo que no ha resultado tan intuitivo o sencillo es la instalación del paquete. Por lo que hemos podido ver, este paquete estaba inicialmente escrito en C++ y arrastra ciertos inconvenientes de ello en su API de Python. Sigue siendo necesario descargar el source code y compilarlo manualmente. Este proceso no es muy directo tampoco, ya que se necesitan ciertos ajustes a algunas variables para lograr que compile. Una vez se ha compilado, debemos instalar los requisitos extras para el uso de librerías encima de C++ como la que hemos usado para Python.

En definitiva, el proceso de instalado no es muy agradable para el usuario pese a estar bien explicado en la documentación y dista mucho de la experiencia común con librerías de Python.

7 Conclusión

Tras ejecutar y analizar con detalle los experimentos propuestos en las tareas de este trabajo, hemos llegado a diversas conclusiones.

En primer lugar, se ha visto que si se trabaja con menos etiquetas se pueden aumentar ligeramente los resultados del etiquetado del corpus. Esto beneficia al rendimiento del etiquetador, pero tiene el coste de no tener unas etiquetas más específicas y detalladas como las originales.

En segundo lugar, se ha comprobado que cuanto mayor sea el conjunto de entrenamiento los resultados serán mejores, utilizando un conjunto fijo de test. Como hemos comentado, este resultado era esperable ya que en general, cuántos más datos se tengan para entrenar, mejor rendimiento conseguiremos con el clasificador, en este caso un etiquetador morfosintáctico.

En tercer lugar, se han evaluado las prestaciones del etiquetador TnT, utilizando un método de suavizado basado en sufijos para las palabras desconocidas. Se ha visto que utilizando una longitud de sufijo de 3 para el suavizado se consigue una mejora de los resultados del etiquetador sin aplicar este suavizado.

A continuación, se ha evaluado el funcionamiento de otros paradigmas de etiquetadores clásicos con el corpus procesado. Hemos visto que se consiguen resultados bastante altos y aceptables.

En la última tarea, se ha evaluado el paquete Freeling para el etiquetado morfosintáctico de un fichero así como su experiencia de uso. Se ha visto que el proceso de instalación no ha sido muy intuitivo y podría mejorarse, pero una vez instalado su uso ha sido instintivo.

En definitiva, hemos aprendido bastante sobre distintos modelos de POS-tagging y cómo se ven afectados por distintos parámetros. Ha sido un trabajo enriquecedor y agradecemos la presencia de trabajos de esta magnitud en la asignatura.

8 Preguntas sobre el trabajo

- ¿Es conveniente utilizar el etiquetador TnT solamente?

  No. Como hemos podido comprobar, si aplicamos un método de suavizado para etiquetar las palabras desconocidas el rendimiento del etiquetador será mejor, ya que TnT no incorpora un método de suavizado para las palabras desconocidas.

- ¿Viendo los resultados de la tabla en la tarea 4, qué etiquetador utilizarías para etiquetar el corpus?

  Si vemos la gráfica de la tarea 4, el etiquetador perceptrón es el que mejor resultado consigue, y como los intervalos de confianza no se solapan con otros etiquetadores, escogería el etiquetador perceptrón para esta tarea, ya que las diferencias en el rendimiento con los otros etiquetadores son significativas. Aunque habría que hacer una evaluación más robusta como validación cruzada para asegurarnos de cual es mejor.

Enlace al vídeo