Trabajo LC

Alejandro Granados Bañuls - Lorenzo del pianta Pérez

Introducción

En el presente trabajo se va a abordar diferentes modelos para el etiquetado POS (Part Of Speech) para el corpus *cess_esp*, disponible en el *package* de python *nltk*. En cada tarea se realizarán una serie de experimentos y se comentarán las conclusiones de estos experimentos.

Tarea 1

En este apartado se va a mostrar cómo afecta el tamaño de un corpus de etiquetas reducido frente a uno completo, es decir, se va a analizar cómo afecta un mayor número de etiquetas a la hora de clasificar. El modelo generado será un *hmm*.

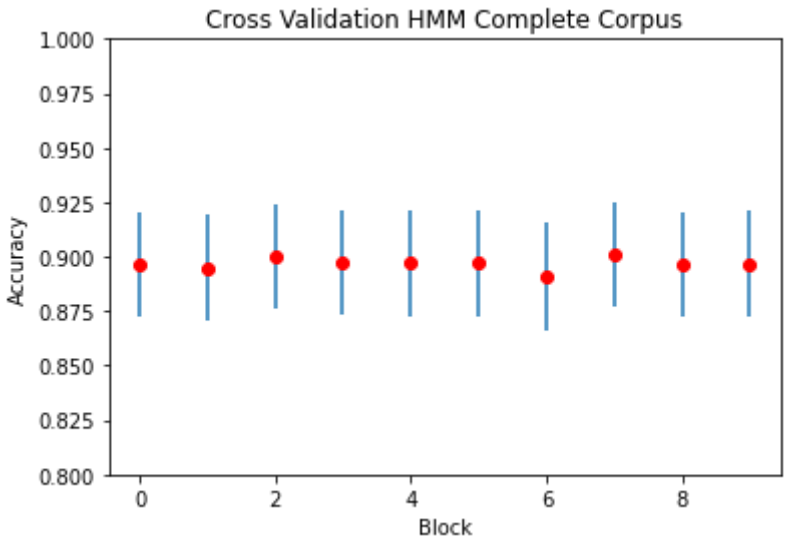

Figura 1: Precisión sobre el corpus completo

Bloque	Accuracy
0	89,26%
1	89,65%
2	89,97%
3	89,54%
4	89,62%
5	89,73%
6	89,78%
7	89,73%
8	90,36%
9	89,15%

Tabla 1: Precisión sobre el corpus completo

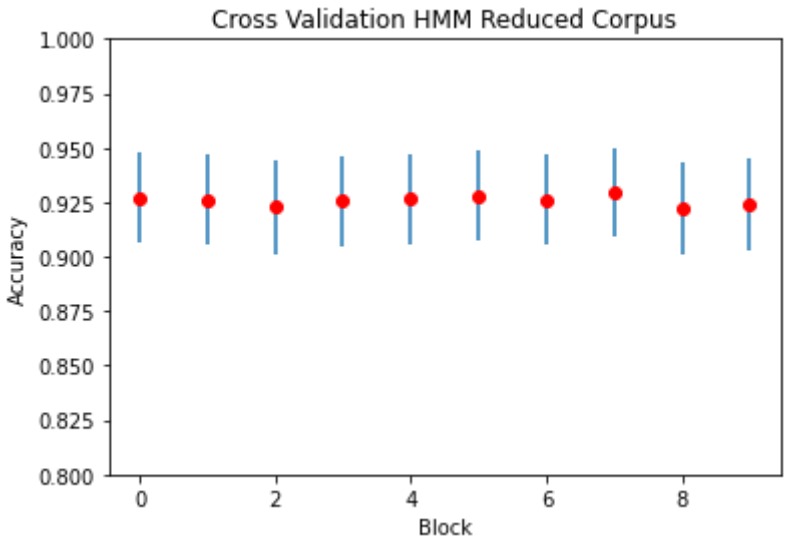

Figura 2: Precisión sobre el corpus reducido

Bloque	Accuracy
0	92,88%
1	92,40%
2	92,47%
3	92,75%
4	92,85%
5	92,64%
6	92,69%
7	92,07%
8	92,60%
9	92,56%

Tabla 2: Precisión sobre el corpus reducido

Se ha decidido añadir los intervalos de confianza en el final del cross validation y no en todas las precisiones de las tablas para no añadir ruido y que sea más fácil de visualizar.

A partir de las figuras 1 y 2 se puede ver como en los dos casos la tendencia de la precisión es similar, pero la precisión es mayor en el reducido. En el corpus completo se obtiene una precisión de un 89%-90%, en cambio con el reducido un 92%-93%.

Esto tiene mucho sentido, pues al haber menos etiquetas en el corpus reducido, es más sencillo aprender a clasificar y puede realizar una generalización más fácilmente.

Tarea 2

En esta tarea se va comprobar como afecta la cantidad de datos en el entrenamiento respecto a la precisión del modelo. Para ello, se dividirán los datos en 10 bloques, dejando 1 para *test*. El resto de

bloques se usrán para ir entrenando un *hmm*, aumentando el número de bloques usados en el entrenamiento, desde 1 a 9.

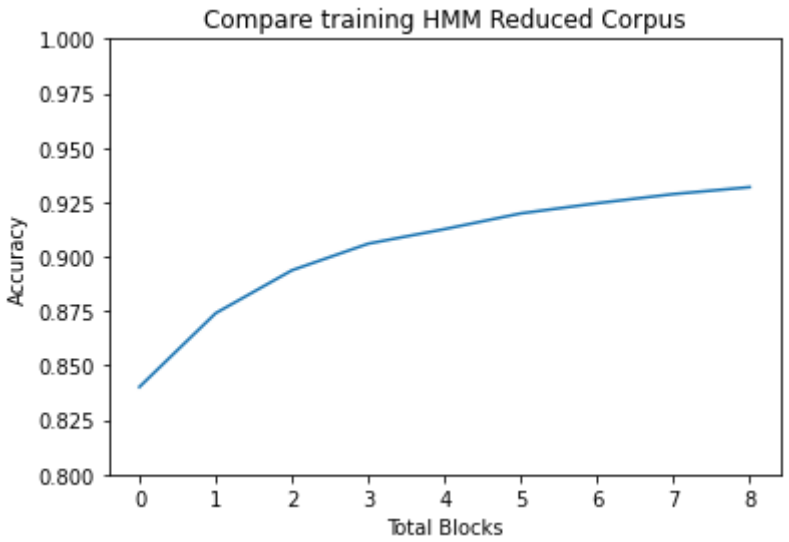

Figura 3: Precisión *hmm* aumentando entrenemaiento

Como se puede observar en la figura 3, cuantos más datos de entrenamiento se usan mayor es la precisión. Esto tienen mucho sentido, ya que cuanto más datos tienen, más casos se ven y más fácil es generalizar.

Tarea 3

En esta tarea se va experimentar el impacto del suavizado para palabras desconcidas de un modelo de etiquetado. El modelo que se usará para la experimentación es el *tnt* y el método de suavizado para las palabras desconocidas el *AffixTagger*, en el que se usarán los sufijos de las palabras para generar el modelo. Para poder elegir un buen parámetro para el sufijo, se deberá estimar el mejor tamaño de sufijo, para obtener la mejor precisión.

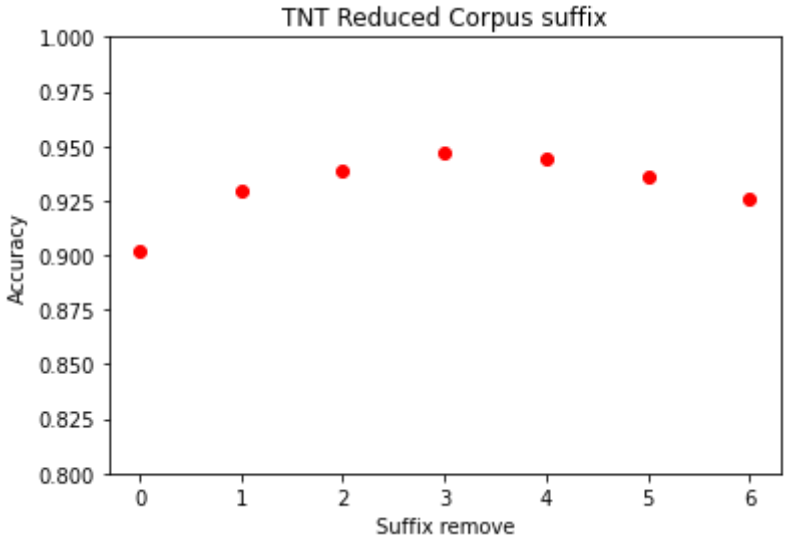

Figura 5: Precisión cross validation con diferentes tamaños de sufijos

En la figura 4 se puede observar como el mejor parámetro es para el *AffixTagger* es el 3, así que después de entrenar el modelo *tnt* con ese método de suavizado se obtiene una precisión de 93% - 94%, que es mayor que la obtenida en proyectos anteriores sin método de suavizado. Así que es mejor usar suavizado a no usarlo.

Tarea 4

En esta tarea se van a comprar varios etiquetadores y se va a obtener cuál es el mejor para la tarea dada. En esta tarea se usará el corpus reducido.

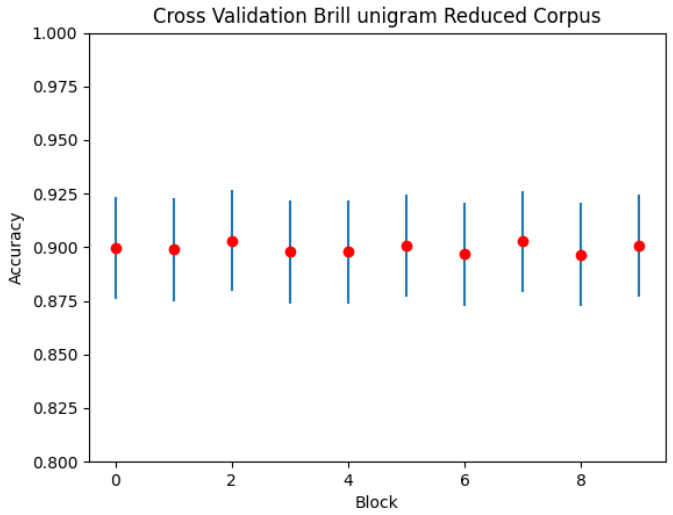

Figura 5: Etiquetador brill con unigramas

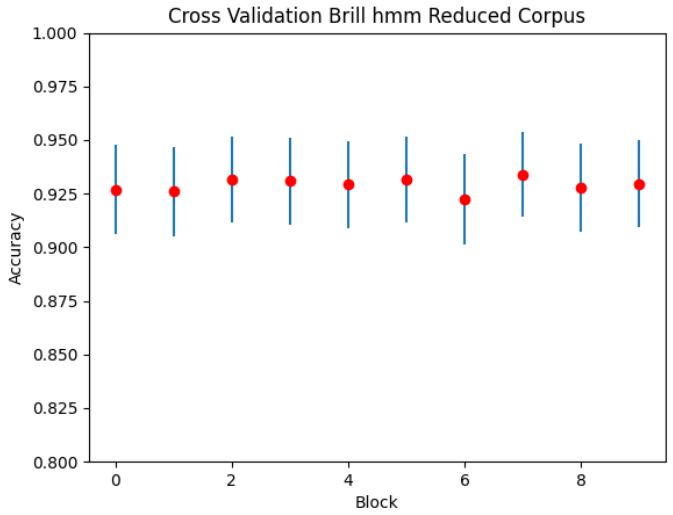

Figura 6: Etiquetador brill con biigramas

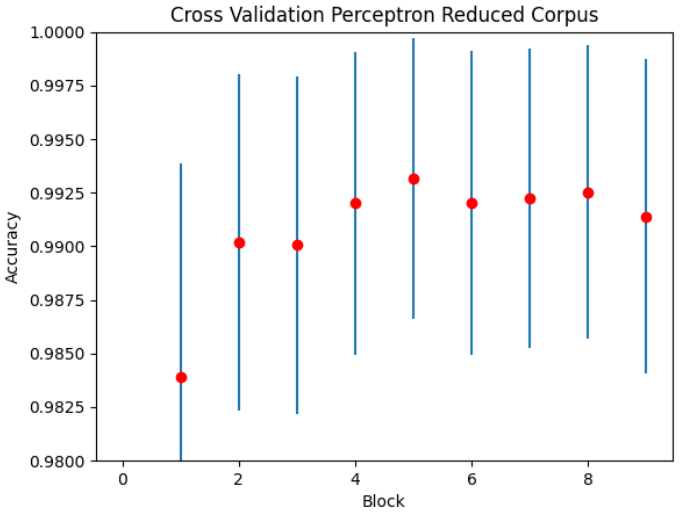

Figura 7: Etiquetador perceptron

Cómo se puede ver en la figuras, el modelo que mejores resultados proporciona es le modelo Perceptrón, así que es ese el que se debería usar para el problema del etiquetado POS para el corpus *cess_esp*, pues la precisión que se obtiene es de 98% - 99%.

Tarea 5

La herramienta *freeling* es una herramienta útil para realizar un etiquetado POS en textos, pues proporciona mucha información en el resultado finalcdel etieutado y contiene muchos idiomas en los que realizarlo. A parte, la herramienta proporciona funcionalidades para entrenar modelos POS, como un *hmm*.

Pero en el momento de la instalación y uso de de las APIs (de python3 en este caso) se dan varios problemas. En el actual proyecto se ha usado *Ubuntu 21.04* y la instalación ha dado varios inconvenientes, unos con permisos y otro con ambigüedad del directorio donde se instalaron las funcionlaidades.

A la hora de usar las APIs, la de python3 específicamente, hubieron problemas con el archivo *pyfreeling.py*, pues al instalar la aplicación se crearon dos directorios muy parecidos llamados freeling, y se producía una confusión sobre que directorio había que usar.

También daba problemas la variable de entorno *$FLINSTALL*, pues se asummía que se creaba en el proceso de instalación, y también se relaciona con la ambigüedad en los directorios creados por la instalación.

Hay que recalcar que la documentación, aunque muy completa, es difícil de seguir. Esto podría resolverse añadiendo ejemplos como pasa en otras documentaciones como *numpy* o *spacy*.

Aunque las funcionaldiades que incorpora la herramienta *freeling* son buenas y completas, hay opciones que también aportan esas funcionaldiades y son más sencillas de usar con una

documentación más completa como *spacy*, una librería de *python* dirigida al procesamiento del lenguaje natural que incorpora etiquetado POS.

Preguntas

¿Es necesario dividir en en tres el corpus (training, test, validación)?
Teniendo en cuenta que se está usando siempre cross validation con datos mezclados no sería necesario, pues sería díficil estar adaptando los modelos al conjunto de validación.

¿Se deberían haber probado más modelos?
Teniendo en cuenta que el perceptron proporciona una precisión muy buena se puede ddecir que no es necesario probar otros modelos, pero si que puede ser importante el comparar con más modelos y diferentes corpus, para saber si el modelo es adecuado a esta tarea o a más tareas, o si hay algún modelo que lo mejore.