Octubre 2021

Lingüística Computacional: Evaluación de etiquetadores morfosintácticos para el español

Alumnos:
Jose Arias Moncho
Victoria Beltrán Domínguez

Índice

Introducción

En este trabajo, se plantea construir y evaluar las prestaciones de distintos modelos de lenguaje utilizando el paquete *nltk*. Para ello, utilizando el corpus español *cess-esp,* se experimentará con diferentes modelos y técnicas de suavizado para observar cómo varían.

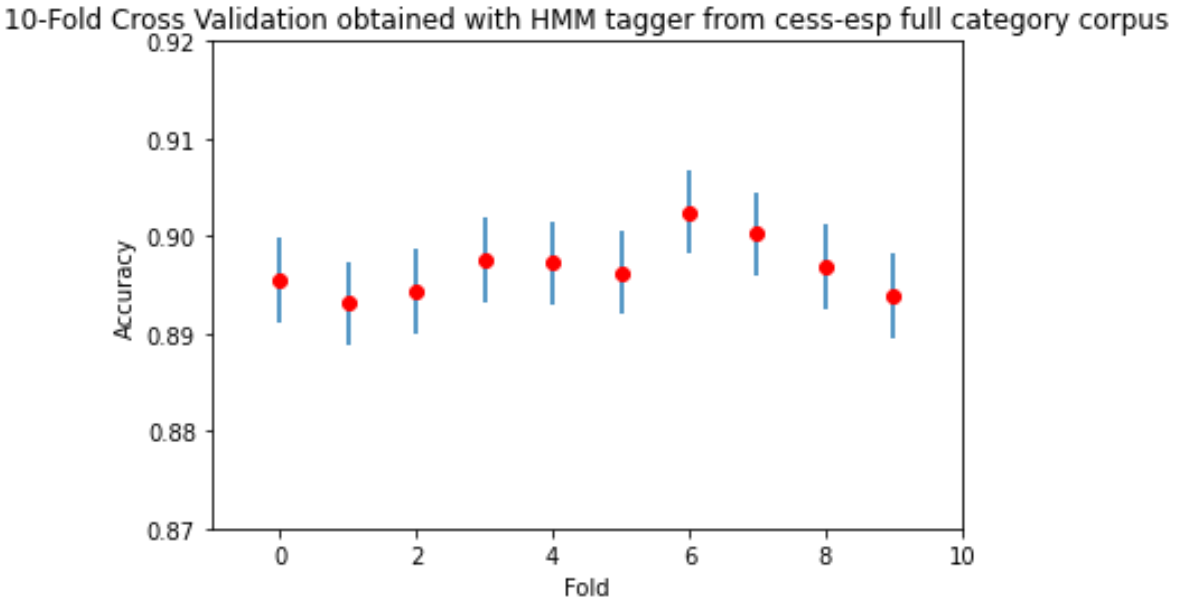
De esta manera se proponen una serie de tareas. En la primera tarea deberemos evaluar un modelo creado con el etiquetador hmm para el corpus cess-esp (con el conjunto de categorías completo y reducido). A continuación, se deberá estudiar cómo varían las prestaciones del etiquetador hmm cuando varía el tamaño del corpus de aprendizaje. Una vez descubierta la variación de las prestaciones al cambiar el corpus de aprendizaje, deberemos descubrir cuál es la longitud del sufijo óptima e incorporarlo como modelo de suavizado al etiquetador tnt, comprobando así si aumenta sus prestaciones y, finalmente deberemos probar diferentes paradigmas de etiquetado e indagar sobre la instalación y funcionamiento del etiquetador morfosintáctico *Freeling*.

Pasamos a detallar cómo se han solventado cada una de las tareas, así como qué resultados hemos obtenido.

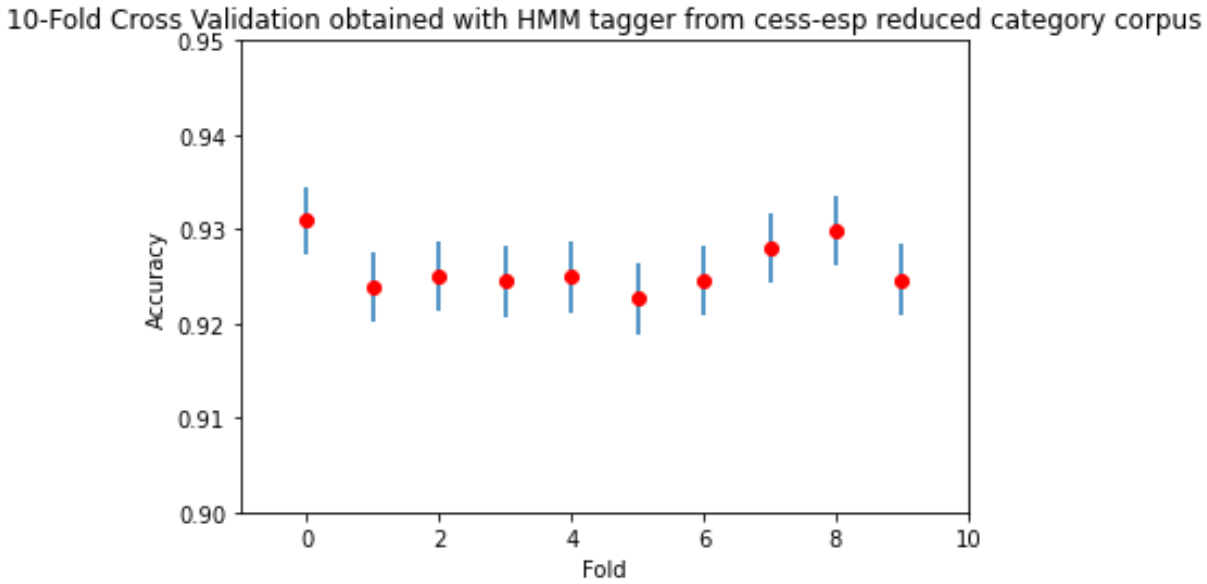
Tarea 1: Evaluación del etiquetador ‘hmm’ sobre el corpus ‘cess-esp’ utilizando el juego de categorías completo y reducido

En la primera tarea se debe trabajar con el etiquetador basado en modelos de Markov (hmm) y realizar validación cruzada con 10 particiones sobre el corpus correspondiente al juego de categorías completo y reducido.

Los resultados del etiquetador con el corpus completo pueden observarse en la siguiente gráfica:



Por otro lado, para el corpus con categorías reducidas se obtienen los siguientes resultados:



Los resultados en forma de tabla se presentan a continuación:

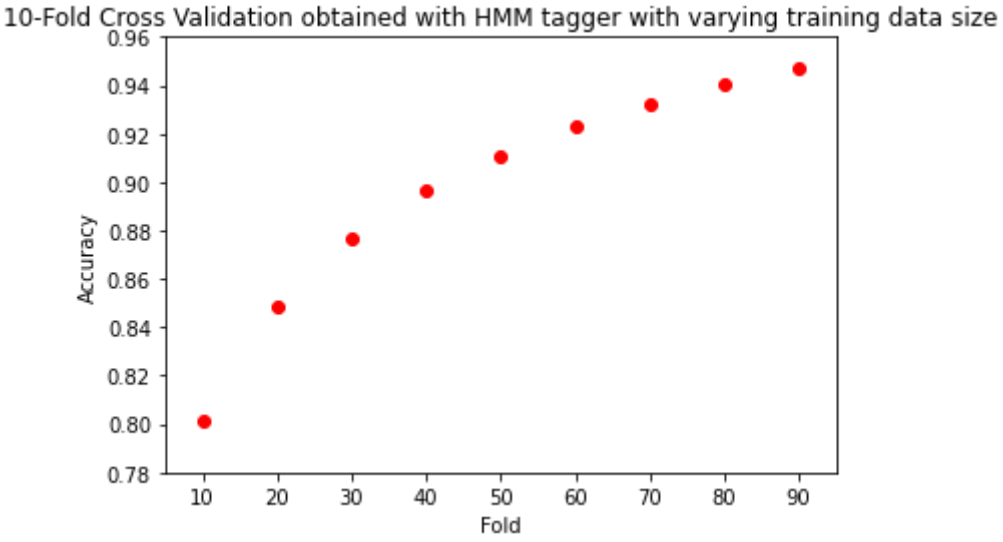
Tipo de etiquetado	Precisión (%)	Intervalo de confianza (%)
Completo	89.5	[89.1, 89.9]
	89.3	[88.9, 89.7]
	89.4	[89.0, 89.8]
	89.7	[89.3, 90.1]
	89.7	[89.3, 90.1]
	89.6	[89.2, 90.0]
	90.2	[89.8, 90.6]
	90	[89.6, 90.4]
	89.6	[89.2, 90.0]
	89.3	[88.9, 89.7]

Reducido	93	[92.7, 93.3]
	92.3	[92.0, 92.6]
	92.5	[92.2, 92.8]
	92.4	[92.1, 92.7]
	92.4	[92.1, 92.7]
	92.2	[91.9, 92.5]
	92.4	[92.1, 92.7]
	92.8	[92.5, 93.1]
	92.9	[92.6, 93.2]
	92.4	[92.1, 92.7]

Como se puede observar, los resultados obtenidos a partir del corpus con categorías reducidas son mejores, debido a que se simplifica la tarea de POS-tagging al reducir el número de clases en la clasificación de las palabras. En concreto, podemos observar que en el corpus con categorías reducidas se mejora la precisión del modelo obtenido en aproximadamente un 2%.

Tarea 2: Evaluación de las prestaciones del etiquetador respecto a la cantidad de datos de aprendizaje

En esta segunda tarea se pide trabajar con el mismo etiquetador que en la tarea anterior con el fin de comprobar de qué manera afecta en las prestaciones trabajar con más datos de entrenamiento. Realizando una serie de experimentos obtenemos la siguiente gráfica:



Los resultados en forma de tabla para esta gráfica serían:

Porcentaje de los datos de entrenamiento	**Precisión (%)**	**Intervalo de confianza (%)**
10%	80.1	[80.0, 80.2]
20%	84.8	[84.7, 84.9]
30%	87.6	[87.5, 87.7]
40%	89.6	[89.5, 89.7]
50%	91.0	[90.9, 91.1]
60%	92.2	[92.1, 92.3]
70%	93.2	[93.1, 93.3]
80%	94.0	[93.9, 94.1]
90%	94.6	[94.5, 94.7]

Tal y como podemos comprobar en los resultados obtenidos en la tabla superior, la precisión incrementa de manera proporcional con la cantidad de datos que se han utilizado para entrenar al etiquetador, indicando que de manera general, cuántos más datos se utilizan para entrenar, mejor será la precisión obtenida por el modelo.

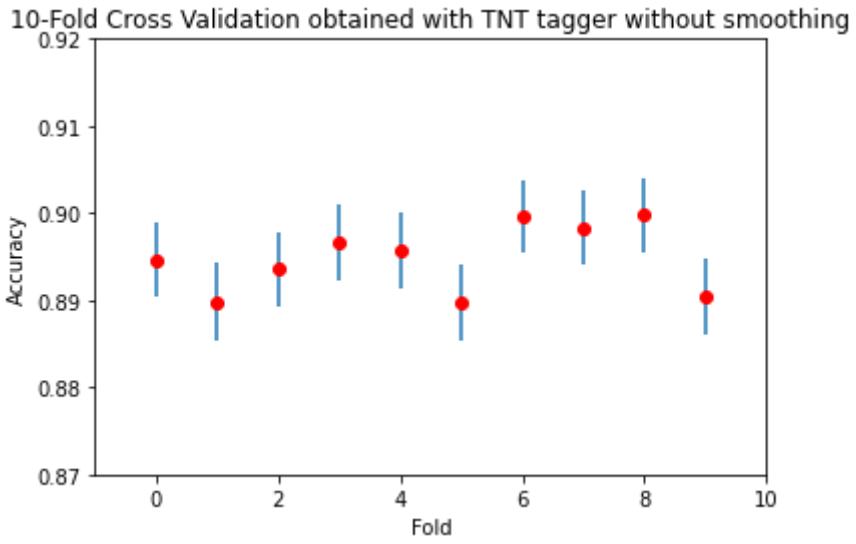
Tarea 3: Evaluación del método de suavizado para palabras desconocidas para el etiquetador ‘tnt’

En esta tercera tarea se pide crear un modelo utilizando tnt sin suavizado. A continuación, se realizará un experimento para descubrir cuál es la longitud de sufijo óptima y se incorporará como modelo de suavizado al etiquetador tnt, comprobando así si aumenta sus prestaciones.

De esta manera, en primer lugar obtenemos los resultados del etiquetador tnt sin realizar ningún tipo de suavizado. Los resultados se muestran en la siguiente tabla:

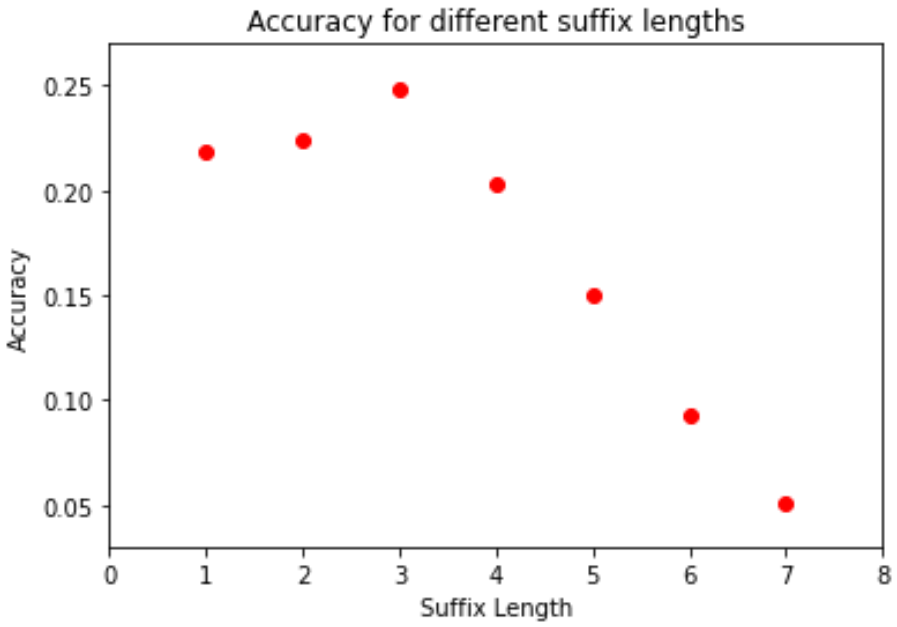
Etiquetador	Precisión (%)	Intervalo de confianza (%)
Tnt sin suavizado	89.4	[89.0, 89.8]
	88.9	[88.5, 89.3]
	89.3	[88.9, 89.7]
	89.6	[89.2, 90.0]
	89.5	[89.1, 89.9]
	88.9	[88.5, 89.3]
	89.9	[89.5, 90.3]
	89.8	[89.4, 90.2]
	89.9	[89.5, 90.3]
	89.0	[88.6, 89.4]

Y la visualización gráfica se muestra a continuación:



Una vez tenemos los resultados del etiquetador tnt sin suavizar, procedemos a comprobar qué longitud de sufijo obtiene los mejores resultados. Para ello, vamos a trabajar con un modelo Affix Tagger, experimentando con diferentes longitudes del tamaño de sufijo ($|suf| = \{1, 2, 3, 4, 5, 6, 7\}$). Se maneja para ello el corpus cess-esp (utilizando el primer 90% de los datos para entrenar al modelo y el 10% restante para evaluarlo). Los resultados pueden observarse a continuación:

Tamaño del sufijo	**Precisión (%)**
1	21.8
2	22.3
3	24.8
4	20.2
5	15.0
6	9.2
7	5.0

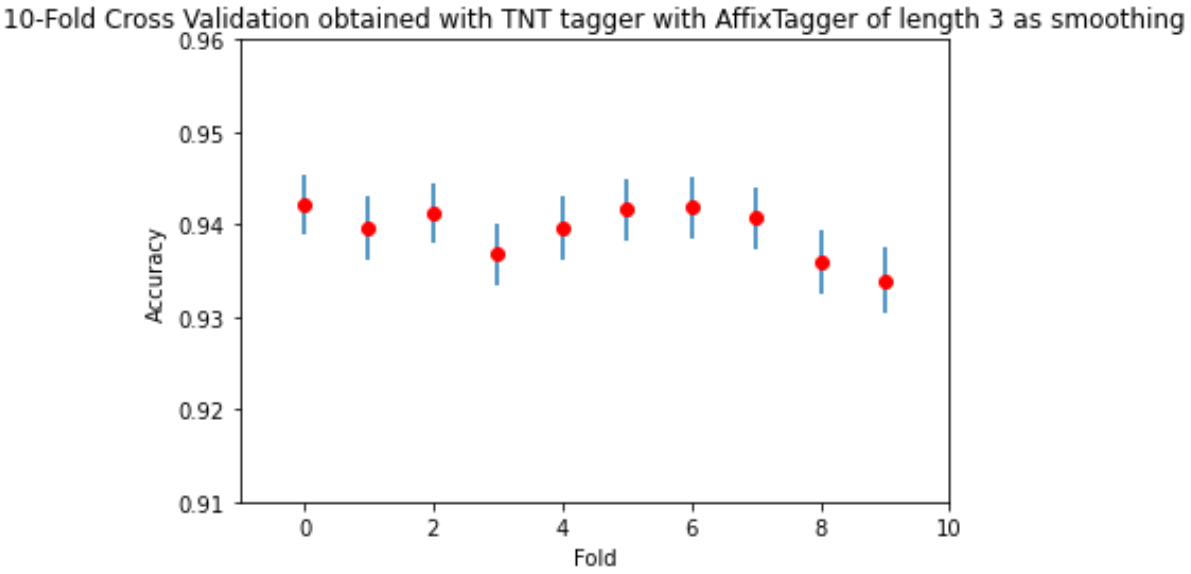


Como podemos ver en la tabla y gráfica anteriores, el modelo que entrena con $|suf| = 3$ obtiene los mejores resultados, por lo que este modelo será nuestro modelo suavizador del etiquetador tnt.

Así, entrenamos el modelo tnt esta vez utilizando como modelo de suavizado el mejor modelo AffixTagger($|suf| = 3$) entrenado con todo el conjunto de datos del corpus cess-esp. Realizando validación cruzada con el modelo entrenado, obtenemos estos resultados:

Etiquetador	**Precisión (%)**	**Intervalo de confianza (%)**
Tnt con un modelo de suavizado Affix Tagger con longitud de sufijo 3	94.2	[93.9, 94.5]
	93.9	[93.6, 94.2]
	94.1	[93.8, 94.4]
	93.6	[93.3, 93.9]
	93.9	[93.6, 94.2]
	94.1	[93.8, 94.4]
	94.1	[93.8, 94.4]
	94.0	[93.7, 94.3]

	93.5	[93.2, 93.8]
	93.3	[93.0, 93.6]



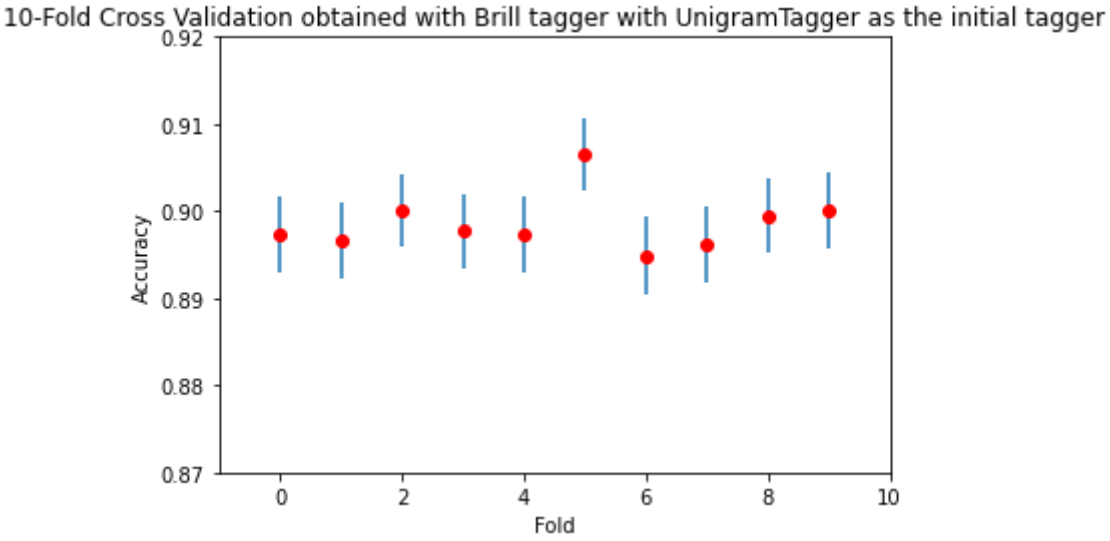
Con estos resultados podemos comprobar cómo se consigue una mejora en comparación con los obtenidos sin suavizado, de forma que este modelo Affix Tagger aumenta la precisión del etiquetador en aproximadamente un 5%.

Tarea 4: Evaluación del resto de etiquetadores

En esta tarea se nos indica que debemos trabajar con otros etiquetadores con el fin de compararlos con los resultados obtenidos utilizando hmm y el tnt.

Por ello, en primer lugar hemos trabajado con el etiquetador Brill. Este etiquetador se basa en la idea de utilizar un etiquetador inicial entrenado y aplicarle una serie de reglas morfológicas que tendrán en cuenta casos concretos en los que realizar cambios en el POS-tagging. De esta forma, en primer lugar hemos utilizado un modelo de unigramas que nos ha reportado los siguientes resultados:

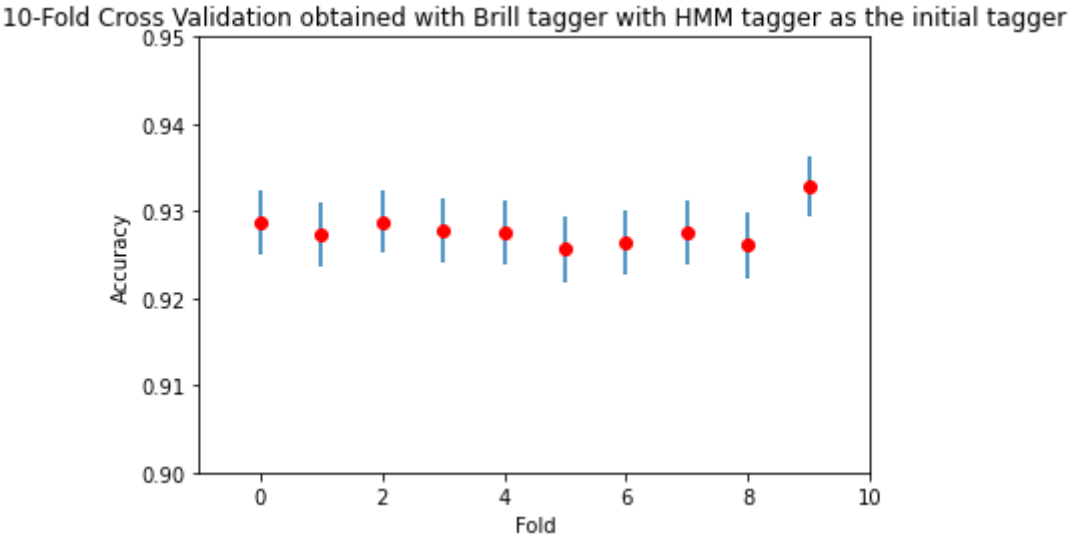
Etiquetador	Precisión (%)	Intervalo de confianza (%)
Brill Tagger con Unigramas	89.7	[89.3, 90.1]
	89.6	[89.2, 90.0]
	90.0	[89.6, 90.4]
	89.7	[89.3, 90.1]
	89.7	[89.3, 90.1]
	90.6	[90.2, 91.0]
	89.4	[89.0, 89.8]
	89.6	[89.2, 90.0]
	89.9	[89.5, 90.3]
	89.9	[89.5, 90.3]



Este etiquetador obtiene resultados inferiores a los que se obtiene con el modelo hmm con el corpus de categorías reducido y en comparación al modelo tnt con suavizado, siendo un 2% inferior en el mejor caso.

Tras esto, vamos a utilizar el mismo Brill tagger, pero como etiquetador inicial vamos a trabajar con un hmm, de tal forma que los resultados pueden observarse debajo:

Etiquetador	Precisión (%)	Intervalo de confianza (%)
Brill Tagger con HMM	92.8	[92.5, 93.1]
	92.7	[92.4, 93.0]
	92.8	[92.5, 93.1]
	92.7	[92.4, 93.0]
	92.7	[92.4, 93.0]
	92.5	[92.2, 92.8]
	92.6	[92.3, 93.9]
	92.7	[92.4, 93.0]
	92.6	[92.3, 93.9]
	93.2	[92.9, 93.5]

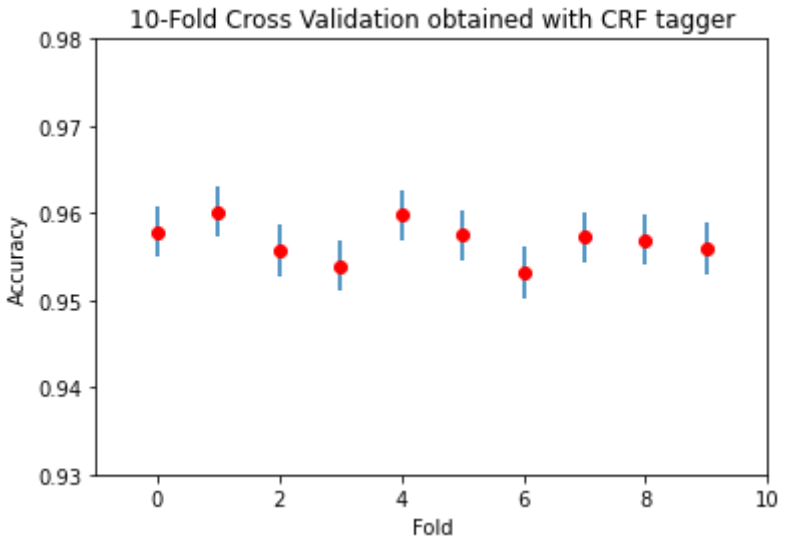


En esta gráfica podemos observar que al utilizar un etiquetador inicial más sofisticado se obtienen mejores resultados, pero estos siguen sin ser

impresionantes, siendo prácticamente idénticos a los obtenidos con el etiquetador hmm, sin utilizar el Brill.

Tras utilizar el etiquetador Brill, vamos a comprobar el desempeño del etiquetador CRF, que trabaja reconociendo partes del texto que considera relevantes y buscando patrones en los datos. Tras ejecutarlo, los resultados obtenidos son los siguientes:

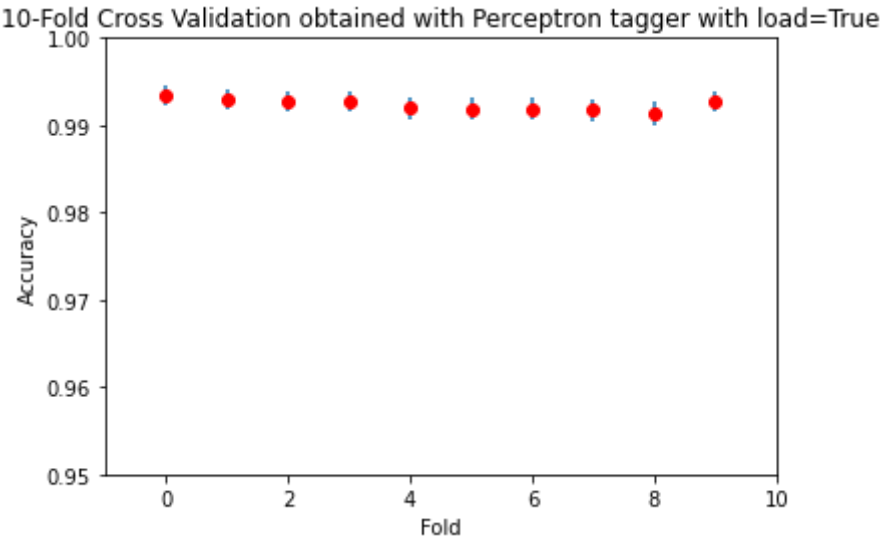
Etiquetador	Precisión (%)	Intervalo de confianza (%)
CRF	95.7	[95.5, 95.9]
	96.0	[95.8, 96.2]
	95.5	[95.3, 95.7]
	95.3	[95.1, 95.5]
	95.9	[95.7, 96.1]
	95.7	[95.5, 95.9]
	95.3	[95.1, 95.5]
	95.7	[95.5, 95.9]
	95.6	[95.4, 95.8]
	95.5	[95.3, 95.7]



Estos resultados son los mejores que se han obtenido a lo largo de la tarea de POS-tagging y en la tabla se puede observar que para todas las particiones de validación cruzada la precisión ronda el 95.5%.

Por último, utilizaremos el etiquetador Perceptrón, con el cual se obtiene estos resultados:

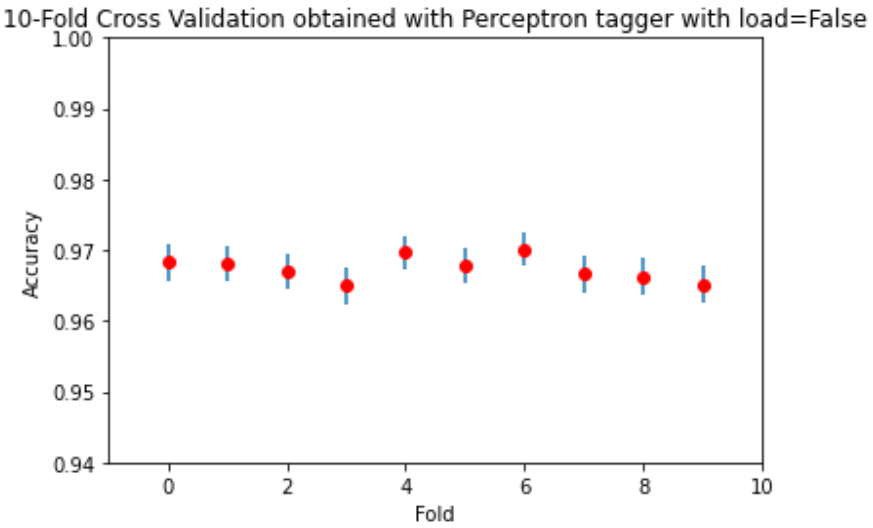
Etiquetador	**Precisión (%)**	**Intervalo de confianza (%)**
Perceptrón con load= True	99.3	[99.2, 99.5]
	99.2	[99.1, 99.3]
	99.2	[99.1, 99.3]
	99.2	[99.1, 99.3]
	99.1	[99.0, 99.2]
	99.1	[99.0, 99.2]
	99.1	[99.0, 99.2]
	99.1	[99.0, 99.2]
	99.1	[99.0, 99.2]
	99.2	[99.1, 99.3]



Esta tabla y gráfica nos muestra los mejores resultados de todos los etiquetadores que hemos utilizado a lo largo de las diferentes tareas, y hemos considerado que eran tan buenos, que al investigar un poco hemos encontrado que si no le indicas al etiquetador explícitamente que no cargue los pesos del modelo, este modelo se inicializará con unos pesos iniciales ya entrenados.

Debido a esto, finalmente hemos utilizado el mismo etiquetador, pero sin utilizar estos pesos iniciales y los resultados han sido:

Etiquetador	**Precisión (%)**	**Intervalo de confianza (%)**
Perceptrón con load= False	96.8	[96.6, 97.0]
	96.8	[96.6, 97.0]
	96.7	[96.5, 96.9]
	96.4	[96.2, 96.6]
	96.9	[96.7, 97.1]
	96.7	[96.5, 96.9]
	97.0	[96.8, 97.2]
	96.6	[96.4, 96.8]
	96.6	[96.4, 96.8]
	96.5	[96.3, 96.7]



Como podemos comprobar, al trabajar con el etiquetador sin unos pesos iniciales ya entrenados los resultados que se obtienen siguen siendo los mejores en comparación al resto de etiquetadores, pero ya no son tan altos como lo eran anteriormente.

Concretamente, los resultados con pesos iniciales eran de 99.2% y al trabajar sin pesos iniciales del modelo ya entrenados, la precisión es de 96.7%, que sigue siendo una precisión superior al resto de modelos en como mínimo un 1.7%.

Tarea 5: Evaluación del paquete Freeling

En esta tarea, se planea realizar un estudio sobre la herramienta *Freeling*.
De esta manera, se procede a evaluar diferentes aspectos de esta:

1. **Documentación**
   La documentación, aunque es muy descriptiva, parece un poco dispersa, pues en vez de darnos todo lo que debemos hacer en una sola página se nos redirecciona varias veces, haciendo que el usuario pierda el hilo de dónde estaba y ocultando toda la información en demasiados links. Sólo para encontrar las instrucciones de instalación debemos ir al link http://nlp.lsi.upc.edu/freeling/ -> Installing -> User Manual -> Freeling 4.2 user manual -> Table of Contents. Es decir, sólo para encontrar las diferentes opciones de instalado, se ha tenido que viajar por cinco páginas diferentes.

2. **Facilidad de instalación**
   La herramienta no nos ha parecido fácil de instalar. En Ubuntu 20.04 no se ha conseguido que funcione, ni utilizando paquetes precompilados ni intentando instalarlo desde la fuente. En Windows, los pasos para instalarlo resultan simples y claros, pero se ha descubierto que la herramienta no funciona con sistemas operativos de Windows 10 pirateados (asumimos que es a causa de la ausencia de algunas dlls).

3. **Facilidad de uso**
   En cuanto a uso, ha sido muy fácil de utilizar, simplemente se ha tenido que ejecutar la siguiente orden (siguiendo las instrucciones encontradas en la documentación):

   ```
   freeling\bin\analyze.bat -f es.cfg < Alicia_utf8.txt >
   .\results.txt
   ```

4. **Funcionalidad**
   Al ejecutar el comando previamente especificado obtenemos un fichero txt con la etiqueta morfosintáctica correspondiente a cada palabra y la correspondiente probabilidad de este en el modelo. Tras un preprocesamiento del resultado, adjuntamos a esta memoria el fichero *resultados.txt*, fichero que plasma los resultados en formato: palabra/etiqueta.

Conclusiones

Para finalizar, simplemente se quiere resaltar el trabajo realizado y algunas conclusiones sobre este.

Una de las primeras lecciones aprendidas, que se puede deducir por lógica, es que trabajar con menos etiquetas conseguirá obtener mejores resultados en el modelo. Además, cuantos más datos de entrenamiento le pasemos al modelo, generalmente va a dar mejores prestaciones.

También se ha aprendido que se puede entrenar un modelo de sufijos, y que este puede hacerse servir para suavizar otro modelo.

Por otro lado, en este trabajo también se ha experimentado con diferentes modelos hasta ahora no conocidos como Affix Tagger, Brill Tagger, CRF o Perceptrón. Ninguno consigue destacar especialmente sobre el resto a excepción del Perceptrón con pesos ya inicializados, aunque asumimos que generalmente el mejor método dependerá del corpus que queramos entrenar.

Antes de acabar, queremos resaltar dos preguntas que sintetizan parte del trabajo realizado:

¿Qué conclusiones se pueden extraer en la fase de entrenamiento de este trabajo?

Lo más evidente a destacar es que trabajar con más información generalmente sacará mejores resultados. No nos referimos sólo a datos de entrenamiento, pues por ejemplo el perceptrón con pesos pre entrenados es el modelo que más información disponía y el que mejor ha funcionado reentrenándolo con nuestros datos. Esto también se puede observar en la tarea 3, pues cuando añadimos al etiquetador tnt un modelo de suavizado entrenado a partir de un Affix Tagger con longitud de sufijo tres, aumentamos sus prestaciones. Y en realidad también estamos aportando más información al sistema.

¿Qué modelo de todos con los que se ha experimentado obtiene los mejores resultados?

El modelo que ha obtenido mejores resultados ha sido el etiquetador Perceptrón con pesos preentrenados. La justificación sigue la línea de la pregunta anterior.

Sin embargo, cabe destacar que estos resultados son relativos a la tarea de clasificación con la que se trabaja y con el corpus específico, de tal forma que para una tarea como puede ser POS-tagging con el corpus reducido, todos los modelos van a obtener mejores resultados, ya que la tarea es menos compleja.