T4: Modelos de Lenguaje Estadísticos

CIMAT, Procesamiento de Lenguaje Natural, Ciencias de la Computación

Profesor: Dr. Adrián Pastor López Monroy

Entregar: Viernes 11 de Marzo de 2022 antes de las 23:59:59

1 Instrucciones

Realiza los siguientes puntos en un notebook de Python *lo mejor organizado y claro posible*. Ponga su nombre completo al archivo de entrega (e.g., adrian_pastor_lopez_monroy.ipynb) y también en la primera celda del notebook junto con el número de Tarea. Al entregar la tarea, sube al classroom el notebook como un archivo (NO zip). El notebook deberá haber sido ejecutado en tú máquina (o colab) y mostrar el resultado en las celdas.

Para cada punto de esta tarea deberá agregar un comentario BREVE de máximo un párrafo explicando que hizo y la principal conclusión del punto. No hacerlo te restará puntos.

1.1 Se vale pedir ayuda y "copiar"

Se vale pedir ayuda y/o copiar con atribución entre los miembros de la clase y apegándose estrictamente a los siguientes puntos:

1. Del total de actividades que se solicitan hacer (8 con valor para esta tarea) solo puedes pedir ayuda/copiar en un total de **una**.
2. Para los puntos dónde se pide ayuda, brevemente escribe en qué pediste ayuda y a quién.
3. Si tuviste que reusar alguna parte de código que no es tuyo, deja claro dos cosas: 1) brevemente porque tuviste dificultad para hacerlo, 2) cómo lo resolvió tu compañero.
4. Se darán hasta 5pts/100pts extra en esta tarea **(no son acumulables, es decir la máxima nota es 100)** si ayudaste a algún compañero y se comenta en ambas tareas. Se pueden ayudar mutuamente en diferentes puntos y ganar cada quién los 5pts/100pts extra. Siempre respetando el punto 1.

2 Modelo de Lenguaje y Evaluación (50pts)

1. **(5pts)** Preprocese todos los tuits de agresividad (positivos y negativos) según su intuición para construir un buen corpus para un modelo de lenguaje (e.g., solo palabras en minúscula, etc.). Agregue tokens especiales de *<s>* y *</s>* según usted considere (e.g., al inicio y final de cada tuit). Defina su vocabulario y enmascare con *<unk>* toda palabra que no esté en su vocabulario.

2. **(20pts)** Entrene tres modelos de lenguaje sobre todos los tuits: $P_{unigramas}(w_1^n)$, $P_{bigramas}(w_1^n)$, $P_{trigramas}(w_1^n)$. Para cada uno proporcione una interfaz (función) sencilla para $P_{n-grama}(w_1^n)$ y $P_{n-grama}(w_1^n|w_{n-N+1}^{n-1})$. Los modelos deben tener una estrategia común para lidiar con secuencias no vistas. Puede optar por un suavizamiento *Laplace* o un *Good-Turing discounting*. Muestre un par de ejemplos de como funciona, al menos uno con una palabra fuera del vocabulario.

3. **(25pts)** Construya un modelo interpolado con valores λ fijos:

$$\hat{P}(w_n|w_{n-2}w_{n-1}) = \lambda_1 P(w_n|w_{n-2}w_{n-1}) + \lambda_2 P(w_n|w_{n-2}w_{n-1}) + \lambda_3 P(w_n)$$

   Para ello experimente con el modelo en particiones **estratificadas** de 80%, 10% y 10% para entrenar (*train*), ajuste de parámetros (*val*) y prueba (*test*) respectivamente. Muestre como bajan o suben las perplejidades en validación, finalmente pruebe una vez en test. Para esto puede explorar algunos valores $\vec{\lambda}$ y elija el mejor, i.e., [1/3, 1/3, 1/3], [.4, .4, .2], [.2, .4, .4], [.5, .4, .1] y [.1, .4, .5].

3 Generación de Texto (50pts)

Para esta parte reentrenará su modelo de lenguaje interpolado para aprender los valores λ:

$$\hat{P}(w_n|w_{n-2}w_{n-1}) = \lambda_1 P(w_n|w_{n-2}w_{n-1}) + \lambda_2 P(w_n|w_{n-2}w_{n-1}) + \lambda_3 P(w_n)$$

Realice las siguientes actividades:

1. **(15pts)** Proponga una estrategia con base en *Expectation Maximization* para encontrar buenos valores de interpolación en $\hat{P}$ usando todo el dataset de agresividad. Para ello experimente con el modelo en particiones de 80%, 10% y 10% para entrenar (*train*), ajustar parámetros (*val*) y probar (*test*) respectivamente. [1] Muestre como bajan las perplejidades en 5 iteraciones que usted elija (de todas las que sean necesarias de acuerdo a su EM) en validación, y pruebe una vez en test. Sino logra hacer este punto, haga los siguientes dos con el modelo de lenguaje con λ fijos.

2. **(15pts)** Haga una función "tuitear" con base en su modelo de lenguaje $\hat{P}$ del último punto. El modelo deberá poder parar *automáticamente* cuando genere el símbolo de terminación de tuit al final (e.g., "</s>"), o 50 palabras. Proponga algo para que en los últimos tokens sea más probable generar el token "</s>". Muestre al menos cinco ejemplos.

3. **(10pts)** Use la intuición que ha ganado en esta tarea y los datos de las mañaneras para entrenar un modelo de lenguaje *AMLO*. Haga una un función "dar_conferencia()". Generé un discurso de 300 palabras y detenga al modelo de forma abrupta.

4. **(5pts)** Calcule el estimado de cada uno sus modelos de lenguaje (el de tuits y el de amlo) para las frases: *"sino gano me voy a la chingada"*, *"ya se va a acabar la corrupción"*.

5. **(5pts)** Para cada oración del punto anterior, haga todas las permutaciones posibles. Calcule su probabilidad a cada nueva frase y muestre el top 3 mas probable y el top 3 menos probable (para ambos modelos de lenguaje). Proponga una frase más y haga lo mismo.

[1] Note que cada partición tiene tanto tuits agresivos y no agresivos

4 (Opcional) El ahorcado (5pts) acumulables para tareas: e.g.; 105/100

(2.5pts) Para esta parte estudie y comprenda el funcionamiento de la estrategia propuesta por Norvig http://norvig.com/spell-correct.html. Siéntete libre de adaptar y/o extender parcial o totalmente el código de Norvig para esta tarea.

Diseñe una función que sea capaz de encontrar los caracteres faltantes de una palabra. Para ello proponga una adaptación simple de la estrategia de corrección ortográfica propuesta por Norvig. La función de el ahorcado debe poder tratar con hasta 4 caracteres desconocidos en palabras de longitud arbitraria. La función debe trabajar en tiempo razonable (≈ 1 minuto en una laptop o menos). La función debe trabajar como sigue con 10 ejemplos:

```
>>> hangman("pe_p_e")
'people'

>>> hangman("phi__sop_y")
'philosophy'

>>> hangman("si_nif_c_nc_")
'significance'
```

Puede resolver este punto con una extensión muy simple de la estrategia de Norvig, o alguna forma más eficiente con distancias de edición (e.g., Levenshtein) o de subcadenas (e.g., Karp Rabin, Aho-Corasick, Tries, etc.).

(2.5pts) Comente brevemente como integraría un modelo de lenguaje con el modelo de Norvig para tratar de resolver errores gramaticales de más alto nivel, o errores dónde el error sea una palabra que si está en el diccionario, por ejemplo: *"In the science off Maths ..."*.