UNIVERSIDAD DE ZARAGOZA
ESCUELA DE INGENIERÍA Y ARQUITECTURA

Modelos de lenguaje

Nombre estudiante: *Alberto Mur López*

Curso: *Sistemas inteligentes*
Docente: *David Abadía*
Fecha de entrega: *November 24, 2023*

INTRODUCCIÓN

En esta práctica nos introducen el concepto de modelo de lenguaje, que son modelos que asignan probabilidades a secuencias de palabras. En este caso trabajaremos con n-gramas, en concreto con trigramas y bigramas.

TRIGRAMAS

Los trigramas son secuencias de tres palabras. Y el modelo lo que nos dice es la probabilidad de esa secuencia. En la imagen 1 se detalla el modelo para dos palabras *a* y *b* y sus probabilidades.

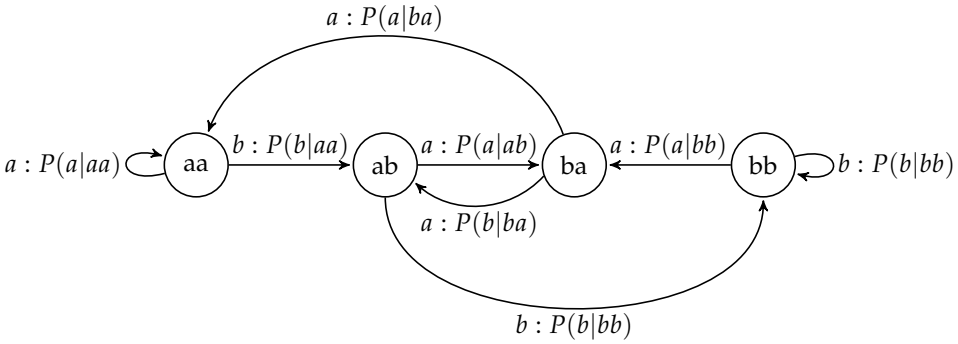

Figure 1: Modelo de trigrama

¿Puedes explicar qué y con qué finalidad se hace el código de la construcción del modelo y en su transformación a probabilidades?

Para poder generar un modelo de lenguaje basado en trigramas, debemos recorrer un texto y calcular para cada par de palabras la probabilidad de la palabra que aparece después. Con ello obtenemos un árbol de probabilidades y estados.

Una estrategia es utilizar un diccionario utilizando como clave el par de palabras. El valor de cada entrada del diccionario será el conteo de las diferentes palabras que pueden aparecer después. Tras haber recorrido todo el texto y obtener el conteo completo para cada estado, calcularemos la probabilidad de que aparezca cada una de las

palabras. Esto nos permitirá determinar que palabra es más probable que aparezca después de un conjunto de palabras.

¿Por qué es necesaria la preparación del texto?

Para generar el modelo de esta frase u otras, es importante preparar el texto para generar los tokens adecuados y eliminar el posible ruido. Por ejemplo, separar los signos de puntuación de las palabras o eliminar partes concretas del mismo.

Generación del modelo

Por ejemplo, para el texto:

> Hugo tuvo un tubo, pero el tubo que tuvo se le rompió. Para recuperar el tubo que tuvo, tuvo que comprar un tubo igual al tubo que tuvo.

El modelo generado es el mostrado en la Figura 2. En dicho modelo, de cada nodo salen las probabilidades de la siguiente palabra teniendo en cuenta el origen de llegada a dicho nodo entre paréntesis.

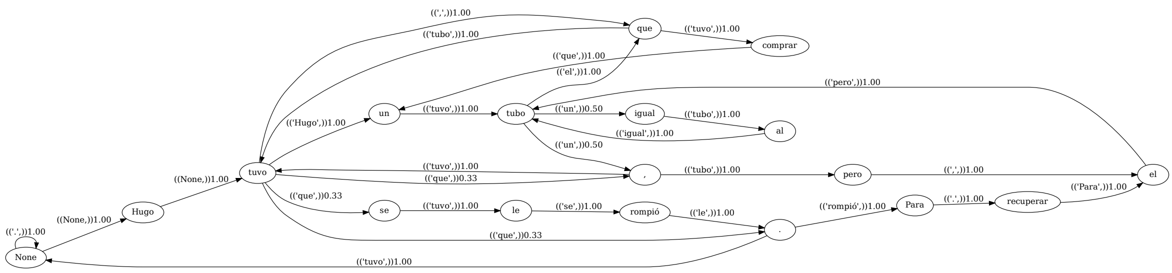

Figure 2: Modelo de trigrama generado

¿Por qué aparecen `None` en los trigramas?

Esto se debe a que para generar los estado válidos y consistentes del inicio y final del texto es necesario rellenar los huecos donde no existe ninguna palabra con algún valor, en este caso `None`.

BIGRAMAS

El modelo de bigramas es el más sencillo de todos, pues solo tiene en cuenta la palabra anterior, y por tanto se simplifican los caminos posibles. La figura 3 representa el modelo de bigrama, en el que se observa como para dos palabras a y b las posibles transiciones son inferiores que en el caso del trigrama.

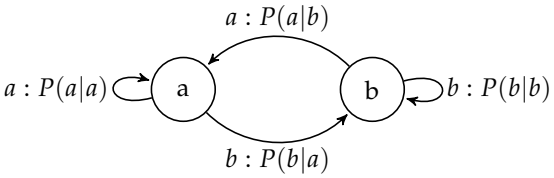

Figure 3: Modelo de birama

Generación del modelo

Para la frase utilizada para generar el modelo de trigrama obtenemos el modelo representado en la figura 4.

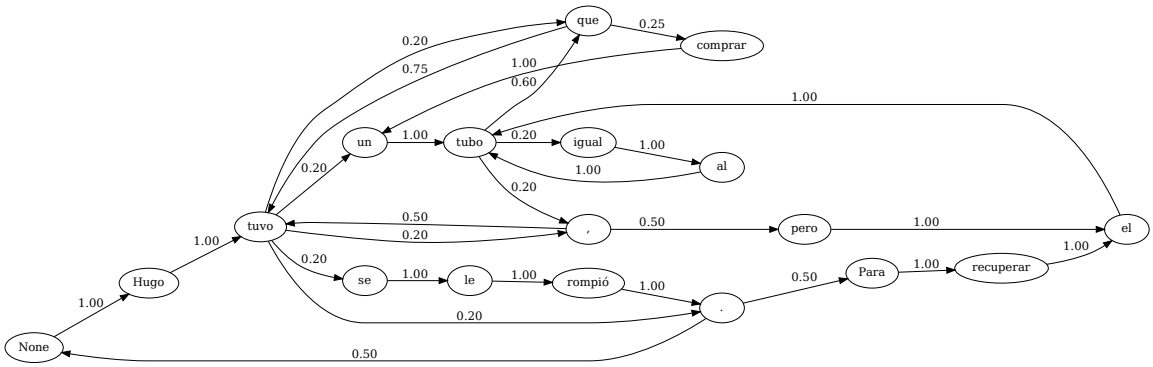

Figure 4: Modelo de bigrama generado

Se puede observar a simple vista como un modelo de trigramas tiene mayor capacidad para generar frases más complejas y diferentes.

EVALUACIÓN

Una manera de evaluar el modelo de lenguaje es a través de lo que se llama perplejidad. Con esta medida se pueden comparar diferentes modelos de lenguaje que usan el mismo vocabulario. Como la limpieza de los textos del proyecto gutenberg era algo tediosa y compleja, se han descargado transcripciones de charlas de TED disponibles en Kaggle[1]. En concreto, las transcripciones en español.

Como experimento adicional se han generado también cuatrigramas. Y a continuación mostramos algunas de las frases arrojadas por nuestros modelos.

Si iniciamos cada uno de los modelos con la frase "me gustaría mucho" obtenemos las siguientes frases generadas para el modelo de bigramas, trigramas y cuatrigramas respectivamente:

> *me gustaría mucho , cocineros , te lanzas esa enfermedad del mundo tridimensional y no solo quería construir soluciones ? Por desgracia , la planta que le gusta . Nuestra retina .*

[1] https://www.kaggle.com/datasets/miguelcorraljr/ted-ultimate-dataset

me gustaría mucho escuchar sus palabras y maldiciones del pueblo congoleño . Ella es veterinaria .

me gustaría mucho aprender de este diagrama en 24 segundos . Y ella entendía algo básico sobre los jóvenes y crear una televisión realmente democrática y global .

Si iniciamos el cada uno de los modelos con la frase "Porque somos parte" obtenemos las siguientes frases generadas para el modelo de bigramas, trigramas y cuatrigramas respectivamente:

Porque somos parte de comprender otras guerras , podemos utilizar ese dinero solo estamos trabajando , 106 000 años , una breve historia . .

Porque somos parte de un lugar más divertido . sí , el oftalmólogo se dio cuenta de que imponerme no iba a lanzarla , la velocidad de la humanidad .

Porque somos parte de la solución ; debe tratarse realmente de un problema como autodidacta de la programación , en la segunda mitad de los recortes de prensa de Donald Trump y Bernie Sanders hubieran decidido no hacer añicos el status quo . Enviar a dos leones esperando en la cola de esa gran distancia de las ciudades en desarrollo puedan recibir personas y proporcionar una oportunidad para afrontar los desafíos del liderazgo mundial , así como los medios para intentar hacer frente a experiencias peligrosas .

Se puede observar que la capacidad para generar frases con un mayor sentido y contexto aumenta de acuerdo a un mayor grado del n-grama.

Esto se puede comprobar calculando la perplejidad para algunas frases (tablas 1 y 2). En general se corresponde que la perplejidad del modelo de trigramas es menor que en el modelo de bigramas. Esto quiere decir que cuanta más información nos da el n-grama sobre la secuencia de palabras, menor es la perplejidad.

Bigrama	**Trigrama**	**Cuatrigrama**
44.70	13.65	2.57

Table 1: Perplejidad: "El propósito y la alegría de la ciencia y la ingeniería ."

Bigrama	**Trigrama**	**Cuatrigrama**
77.62	8.31	1.34

Table 2: Perplejidad: "Mira , tengo estos nuevos submarinos nucleares que estoy construyendo"

N-Gramas con Elixir

Como trabajo adicional se ha implementado una versión de este algoritmo en Elixir. Para su implementación y ejecución hemos utilizado Livebook[2]. Livebook es un entorno de ejecución para Erlang y Elixir con sabor a los cuadernos de Jupyter para Python. Elixir es un lenguaje de programación funcional que se ejecuta en la máquina

[2] `https://livebook.dev/`

virtual de Erlang. El reto ha sido implementar este algoritmo con un paradigma funcional.

Uno de las principales dificultades ha sido realizar el modelo utilizando las funciones ‘map‘ y ‘reduce‘ para cada nodo del diagrama y la generación de frases utilizando ‘tail-recursion‘.