Procesamiento de Lenguaje Natural
Tarea 2

Rayo Mosquera, Jhon Stewar
j.rayom@uniandes.edu.co

De La Rosa Peredo, Carlos Raul
c.delarosap@uniandes.edu.co

Mario Garrido Córdoba
m.garrido10@uniandes.edu.co

Septiembre 2024

1 Introducción

El objetivo de esta tarea es construir y evaluar modelos de N-Gramas utilizando los siguientes dos conjuntos de datos.

- **20 Newsgroups (20N)**: Una colección de aproximadamente 18,800 documentos de grupos de noticias, categorizados en diferentes temas.
- **Blog Authorship Corpus (BAC)**: Una colección de publicaciones de blog de más de 19,300 autores.

Estos conjuntos de datos se utilizan para construir modelos de unigramas, bigramas y trigramas, que se evalúan utilizando la métrica de perplejidad y se emplean para generar oraciones.

2 Implementación

2.1 Preprocesamiento de Datos

El primer paso fue leer y combinar los documentos de los conjuntos de datos 20N y BAC. Los documentos se concatenaron en dos archivos grandes: uno para 20N y otro para BAC.

Para cada conjunto de datos, se implementaron los siguientes pasos de preprocesamiento.

- **Tokenización por Oraciones**: Utilizamos la función `nltk.sent_tokenize()` para dividir el texto en oraciones individuales. Cada oración se envolvió con etiquetas de inicio ("<s>") y fin ("</s>") para facilitar el entrenamiento de los modelos de N-Gramas.
- **Normalización**: Todos los números en el texto fueron reemplazados con el token "NUM". Los tokens con frecuencia unitaria se modelaron como "<UNK>".

2.2 Construcción de Modelos de N-Gramas

Después del preprocesamiento de los datos, cada conjunto de datos se dividió en dos conjuntos.

- **80% para entrenamiento**: Estas oraciones se utilizaron para construir los modelos de N-Gramas.

- **20% para prueba**: Estas oraciones se reservaron para la evaluación de los modelos (i.e., el cálculo de su perplejidad).

Luego, construimos tres modelos de N-Gramas diferentes (unigramas, bigramas y trigramas) para cada conjunto de datos.

- **Unigramas**: Representan palabras individuales como unidades independientes.
- **Bigramas**: Capturan pares de palabras para identificar el contexto inmediato de una palabra.
- **Trigramas**: Consideran una secuencia de tres palabras y pueden capturar dependencias más largas entre palabras.

Es crucial destacar que, debido a la enorme cantidad de datos, se precisó una estructura de datos capaz de almacenar y consultar la información eficazmente. Para ello, se emplearon diccionarios donde se guardaron los tokens junto con sus respectivas frecuencias, facilitando así el cálculo de probabilidades. Este método resulta significativamente más eficiente que el uso de matrices dispersas, ya que éstas, dado el tamaño del corpus, requerirían decenas de gigabytes de memoria.

3 Resultados

3.1 Evaluación de la Perplejidad

La perplejidad de los modelos de unigramas, bigramas y trigramas se calculó utilizando los conjuntos de prueba. La **perplejidad** mide qué tan bien un modelo probabilístico predice una muestra, con valores más bajos indicando mejores predicciones. Para manejar el desbordamiento durante los cálculos de perplejidad, se usaron probabilidades en espacio logarítmico aplicando **suavizado de Laplace** a cada modelo de N-Gramas.

Resultados de Perplejidad para los Modelos de 20N:

- **Unigrama**: 1.0
- **Bigrama**: 4683.49
- **Trigrama**: 168769.46

Resultados de Perplejidad para los Modelos de BAC:

- **Unigrama**: 1.0
- **Bigrama**: 1596.41
- **Trigrama**: 267404.24

Observaciones:

- Los **modelos de unigrama** para ambos conjuntos de datos arrojaron una perplejidad de 1.0, lo que sugiere que el modelo había visto la mayoría de las palabras en el conjunto de prueba. Sin embargo, esto probablemente se deba a la alta prevalencia de palabras comunes y a la simplicidad del modelo de unigrama, que no captura dependencias entre palabras.
- Los **modelos de bigrama** tuvieron un aumento significativo en la perplejidad en comparación con los de unigrama, reflejando la dificultad del modelo para predecir pares de palabras, especialmente en el conjunto de datos 20N, que presenta combinaciones no vistas. Sin embargo, el modelo de bigrama de BAC mostró un mejor rendimiento debido a un estilo más consistente de lenguaje en las publicaciones de blogs.

- Los **modelos de trigramas** tuvieron la perplejidad más alta, lo que indica que los modelos tuvieron dificultades para predecir secuencias de tres palabras. Este resultado es esperado, ya que los modelos de trigrama requieren más datos de entrenamiento para capturar con precisión las dependencias largas entre palabras.

3.2 Generación de Oraciones

Después de la evaluación de los modelos, generamos oraciones utilizando los **modelos de bigrama**. La función toma una palabra inicial y genera una secuencia basada en las probabilidades aprendidas.

Ejemplos de Oraciones Generadas (Modelo de 20N):

- **Palabra inicial "<s>"**: "<s> i have a <UNK> <UNK> <UNK>..."

Ejemplos de Oraciones Generadas (Modelo de BAC):

- **Palabra inicial "book"**: "book and i was a little bit of the same time"
- **Palabra inicial "tonight"**: "tonight i was a little bit of the same time to"

Observaciones:

- Las oraciones generadas por el **modelo de bigrama de 20N** tienden a ser repetitivas y frecuentemente contienen tokens desconocidos ("<UNK>"), lo cual es probablemente causado por la variedad de temas y la dispersión de los datos en el conjunto 20N.
- Las oraciones generadas por el **modelo de bigrama de BAC** son más coherentes y estructuradas, probablemente debido a un estilo de lenguaje más consistente en las publicaciones de blogs. Sin embargo, las oraciones aún carecen de contexto significativo y exhiben cierta repetición, como la frase "a little bit of the same time".

4 Conclusión

En esta tarea, implementamos y evaluamos con éxito modelos de N-Gramas utilizando los conjuntos de datos 20 Newsgroups (20N) y Blog Authorship Corpus (BAC). Los modelos variaron desde simples unigramas hasta bigramas y trigramas más complejos.

Principales conclusiones:

- Los **modelos de unigramas** tuvieron la perplejidad más baja, pero no capturaron relaciones significativas entre palabras.
- Los **modelos de bigramas** mostraron un rendimiento intermedio, con un aumento en la perplejidad debido a la dificultad de predecir pares de palabras.
- Los **modelos de trigramas** presentaron la mayor dificultad para predecir secuencias largas, reflejada en su alta perplejidad.

En conclusión, aunque los modelos de N-Gramas son efectivos para la generación y predicción básica de texto, su rendimiento depende en gran medida del tamaño y la consistencia del conjunto de datos. La tarea destacó las limitaciones de estos modelos, particularmente para capturar dependencias a largo plazo y manejar la dispersión de datos. El trabajo futuro podría involucrar la exploración de técnicas de modelado del lenguaje más avanzadas, como redes neuronales o modelos basados en transformadores,

para superar estas limitaciones y mejorar tanto la perplejidad como la calidad de la generación de oraciones.

Los archivos adjuntos representan los modelos conseguidos. Se excluye el modelo de trigramas para el conjunto BAC pues pesa cerca de 1GB y tuvimos dificultad para subir el archivo, este se puede generar ejecutando ambos notebooks. Tambien se puede referenciar el siguiente repositorio de Github: https://github.com/oyar99/nlp/tree/main/HW02