Detector de Plagio

Alexis Herasimiuk

El siguiente es un archivo de justificaciones de las decisiones tomadas en el desarrollo del Trabajo Práctico N°2 de la materia Procesamiento del Lenguaje Natural a cargo de Mg. Ing. Hernán Borré.

Introducción

El trabajo práctico consiste en desarrollar en Python un sistema capaz de detectar plagio del trabajo práctico de un alumno con respecto a un dataset de trabajos prácticos de la misma materia, de la misma manera, se busca poder detectar plagio con respecto a contenido de páginas web o libros que no han sido citados correctamente. A su vez, debe reconocerse el tema del trabajo práctico, así como el alumno que lo ha confeccionado. Debe reportarse plagio cuando se detecta un 30% de contenido plagiado.

Desarrollo

En principio me enfoqué en la resolución del problema de la detección del tema del trabajo práctico, dado que el dataset no se encontraba etiquetado en su estado inicial, he optado por usar el algoritmo de *Latent Dirichlet Allocation* (Blei et al., 2003), un modelo probabilístico generativo, para colecciones de documentos de un corpus; en donde cada documento se representa con un conjunto de temas, y cada tema está caracterizado por un conjunto de palabras(Blei et al., 2003). De esta forma, conté manualmente la cantidad de temas (resultando ser 14), e inicialicé el modelo. Obtuve resultados no buenos, dado que había temas cuya cantidad de documentos era menor a 2 o 3. Por lo tanto, he decidido reducir la cantidad de temas a 10 para poder tener una distribución más acertada de los temas para cada documento. Aún así, las palabras encontradas no representaban de una manera sencilla el tema del trabajo práctico que permitieran a golpe de vista saber de qué tema se trataba. Así que, si bien se almacenaron los resultados para otro uso posterior, descarté el uso de LDA para clasificación de tópicos.

Dado que LDA no ha podido cumplir con el requerimiento de modelar los 14 temas, opté por clasificar los documentos utilizando el algoritmo de *Naive Bayes* (Stanford University, n.d.), clasificador basado en la *Regla de Bayes*, y en la representación de documentos mediante una *Bag of Words* (Stanford University, n.d.). Como este modelo requiere calcular la probabilidad de cada palabra en una clase, entonces una palabra nunca vista por el clasificador daría como resultado $P(w \mid c) = 0$, por lo tanto es necesario un suavizado, en este caso, he optado por *Laplace Smoothing* (Stanford University, n.d.), que consiste en agregar 1 ocurrencia a cada palabra, de manera que $P(w \mid c) \neq 0$. De esta forma es que la probabilidad se calcula de la siguiente manera:

$$\hat{P}(w \mid c) = \frac{count(w,c) + 1}{count(c) + |V|}$$

Donde w es la palabra, c la clase, V el vocabulario y |V| su cardinalidad.

Para cumplir con tal objetivo, he etiquetado el dataset mediante el uso de un script de Python que detectaba el tópico dentro del título del documento, o bien de forma manual accediendo al mismo. Finalmente se obtiene una efectividad de ~98%, entrenando con el 80% del dataset.

Una vez lograda la clasificación de tópicos, he utilizado las palabras otorgadas por el modelo de LDA para realizar Web Scrapping y obtener resultados web sobre cada tema. Y de la misma manera he buscado manualmente algunos de los libros mencionados en los trabajos prácticos para aumentar las fuentes de detección de plagio.

Como para detectar plagio hay que tener en cuenta no solo la sintaxis sino también la semántica, he decidido decantarme por el uso de *Word Embeddings* (Mikolov et al., n.d.), que permite representar palabras en el espacio vectorial dotando de una semántica a la representación, agrupando palabras similares y logrando por ejemplo, que vec('Madrid') – vec('Spain') + vec('France'), de como resultado un vector más cercano en el espacio a vec('Paris') que el resto de otros vectores. (Mikolov et al., n.d.).

Entonces, queda la detección del plagio en términos de la 'distancia' entre distintas palabras de un párrafo u oración, mediante el cálculo de *Word Mover's Distance* (Wu et al., n.d.). WMD es particularmente útil y preciso para medir la distancia entre documentos que son semánticamente parecidos pero que utilizan distintas palabras (Wu et al., n.d.).

Para lograr esto, los documentos han sido *lematizados* (Plisson et al., n.d.), que es el proceso en el cual se encuentra la forma normalizada de una palabra(Plisson et al., n.d.), y disminuir la 'variedad' de palabras en el corpus. Por ejemplo, si aparece 'clienta' o clientes', todas ellas serán normalizadas a 'cliente'.

De la misma manera, las *stop words,* han sido removidas del corpus para que no afecten en los procesos de clasificación o detección de plagio.

El nombre del alumno es obtenido del documento mediante *Named Entity Recognition* (Krishnan & Ganapathy, 2005) que es el proceso en el cual se obtienen nombres de personas, organizaciones, ubicaciones, expresiones de tiempo, cantidades, unidades monetarias, porcentajes, entre otros (Krishnan & Ganapathy, 2005). Y se filtran nombres de comunes tales como los que se mencionan en los libros, o bien el nombre de los docentes y ayudantes manera manual.

Luego se aumenta la precisión del detector de plagio filtrando oraciones y párrafos comunes tales como consignas o preguntas que aparecen frecuentemente que está claro que aparecerán en la mayoría de los trabajos prácticos del mismo tema.

Finalmente, se comparan documentos que el clasificador de Naive Bayes haya considerado del mismo tema, páginas web que hayan sido encontradas en base a las palabras clave encontradas por LDA, y libros que se han usado en la materia.

Referencias

Blei, D. M., Ng, A. Y., & Edu, J. B. (2003). Latent Dirichlet Allocation Michael I. Jordan. In *Journal of Machine Learning Research* (Vol. 3).

Krishnan, V., & Ganapathy, V. (2005). *Named Entity Recognition*. http://nlp.stanford.edu/javanlp/

Mikolov, T., Chen, K., Corrado, G., & Dean, J. (n.d.). *Distributed Representations of Words and Phrases and their Compositionality*.

Plisson, J., Lavrac, N., & Mladenic, D. (n.d.). *A Rule based Approach to Word Lemmatization*.

Stanford University. (n.d.). *Text Classification and Naïve Bayes The Task of Text Classifica1on*. Retrieved October 26, 2020, from https://web.stanford.edu/class/cs124/lec/naivebayes.pdf

Wu, L., En-Hsu Yen, I., Xu, K., Xu, F., Balakrishnan, A., Chen, P.-Y., Ravikumar, P., & Witbrock, M. J. (n.d.). *Word Mover's Embedding: From Word2Vec to Document Embedding*.

Modelo preentrenado de Word Embeddings

Aitor Almeida, & Aritz Bilbao. (2018). Spanish 3B words Word2Vec Embeddings (Version 1.0) [Data set]. Zenodo. http://doi.org/10.5281/zenodo.1410403

Bilbao-Jayo, A., & Almeida, A. (2018). Automatic political discourse analysis with multi-scale convolutional neural networks and contextual data. International Journal of Distributed Sensor Networks, 14(11), 1550147718811827.