Algoritmo de porter

Por: Jordan Jimenez Gajardo

Profesor: Ricardo Corbinaud Perez

Ayudante: Cristobal Saldias rojas

Año: 2019

Indice

Objetivos 3

Introduccion - marco teorico 4

Elementos de un automata celular 5

Regex 6

Algoritmo de Porter(A.P) 7

Aplicaciones de A.P 8

Text mining 9

Avance 1 y 2 10

Funciones relevantes 15

Comparación Prefijos v/s Sufijos 16

Conclusiones 17

Bibliografia 18

Objetivos

* Dar a conocer el algoritmo de porter
* Analizar autómatas y regex
* Compara resultados de eliminación de sufijos y prefijos
* Fundamentar sobre el algoritmo y sus capacidades

Introduccion

En este trabajo se abordará el tema de el algoritmo de porter, de una manera general y específica,, se dará a conocer algunas aplicaciones de los autómatas celulares, la utilización y que son los regex para que nos sirven información sobre el algoritmo de porter en qué consiste cómo abordamos el tema y cada uno de los pasos que seguimos para llegar al resultado final el cual se dará a conocer por medio de una serie de códigos y funciones en el lenguaje java.

Marco historico

Los autómatas celulares surgen en la década de 1940 con el pionero de la informática John Von Neumann, el cual investigó la cuestión del origen de la vida en intento diseñar una máquina que fuera capaz de autorreproducirse. La idea en el diseño de auto reproducción llevó a Neumann a inventar un sistema llamado Autómatas celulares, capaces de construir cualquier autómata a partir de un conjunto apropiado de instrucciones codificadas.

En cómputo teórico y teoría de lenguajes formales una expresión regular, también conocida como regex, es una secuencia de caracteres que forman un patrón de búsqueda, principalmente utilizada para la búsqueda de patrones de cadenas de caracteres u operaciones de sustituciones.La mayoria de las formalizaciones proporcionan los siguientes constructores: una expresión regular es una forma de representar los lenguajes regulares(finitos o infinitos)y se construye utilizando caracteres del alfabeto sobre cual se define el lenguaje.

En informática, las expresiones regulares proporcionan una manera muy fácil de buscar o reconocer cadenas de texto.

Elementos de un autómata

Arreglo regular: Ya sea un plano de 2 dimensiones o un espacio n-dimensional, este es el espacio de evoluciones y cada división homogénea de arreglo es llamada célula.

Conjunto de estados: Es finito y cada elemento o célula del arreglo toma un valor de este conjunto de estados. También denominado alfabeto. Puede ser expresado en valores o colores.

Configuración inicial: Consiste en asignar un estado a cada una de las células del espacio de evolución inicial del sistema.

Vecindades: Define el conjunto contiguo de células y posición relativa respecto a cada una de ella. A cada vecindad diferente corresponde un elemento del conjunto de estado.

Función local: Es la regla de evolución que determina el comportamiento del autómata celular. Se conforma de una célula central y sus vecindades. Define cómo debe cambiar de estado cada célula dependiendo de los estados anteriores. Esta puede ser una expresión algebraica o un grupo de ecuaciones.

Además, se componen de condición inicial y condiciones frontera.

Condición inicial: Es el estado de cada una de las celdas en el tiempo=0.

Expresiones Regulares (Regex)

Las expresiones regulares proporcionan un método eficaz y flexible para procesar texto. La notación extensiva de búsqueda de patrones coincidentes de las expresiones regulares permite analizar rápidamente grandes cantidades de texto para buscar patrones de caracteres específicos; para validar un texto con el fin de asegurar que se corresponde con un patrón predefinido (por ejemplo, una dirección de correo electrónico); para extraer, editar, reemplazar o eliminar subcadenas de texto; y para agregar las cadenas extraídas a una colección con el fin de generar un informe. Para muchas aplicaciones que usan cadenas o analizan grandes bloques de texto, las expresiones regulares son una herramienta indispensable.

Algoritmo de porter

Porter publicó en 1980 un algoritmo para el método de Stemming que fue tomado como base por muchos investigadores. El algoritmo lee un archivo, toma una serie de caracteres y de esa serie, una palabra; luego la válida verificando que todos los caracteres involucrados sean letras, de ser así, aplica Stemming sobre ella. La aplicación de Stemmer consiste en hacer pasar esta palabra a través de varios conjuntos de reglas, cada conjunto de reglas está formada por n reglas y cada regla por:

1. Un identificador de regla

2. El sufijo a identificar

3. El texto por el cual debe ser reemplazado al encontrar el sufijo

4. El tamaño del sufijo

5. El tamaño del texto de reemplazo

6. El tamaño mínimo que debe tener la raíz resultante luego de aplicar la regla (esto es a los efectos de no procesar palabras demasiado pequeñas)

7. Una función de validación (una función que verifica si se debe aplicar la regla una vez encontrado el sufijo)

Aplicaciones de algoritmo de porter

La frecuencia de una palabra en un texto puede ser útil para muchas tareas, por ejemplo para clasificación de textos o *clustering*, recuperación de información, generación de resúmenes, etc.

He ahí donde radica la importancia del algoritmo de Porter, pues este algoritmo asegura que la forma de las palabras no penalice la frecuencia de éstas. Es decir, una palabra puede estar conjugada en cualquier género, número, persona, etc… y solo se considerará (en muchas ocasiones) como un solo término.

CLUSTERING

Un algoritmo de agrupamiento (en inglés, **clustering**) es un procedimiento de agrupación de una serie de vectores de acuerdo con un criterio. Esos criterios son por lo general distancia o similitud

El algoritmo de porter además de ser muy útil para determinar frecuencias en determinadas frases o textos como por ejemplo *:“Aquel es un caballo de la caballería militar, los otros caballos no.”* La frecuencia de la palabra caball (Que hace referencia al caballo), es de 3.

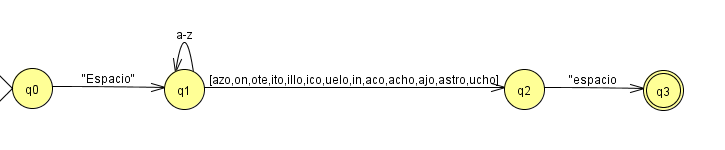
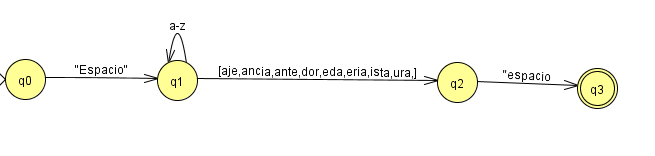
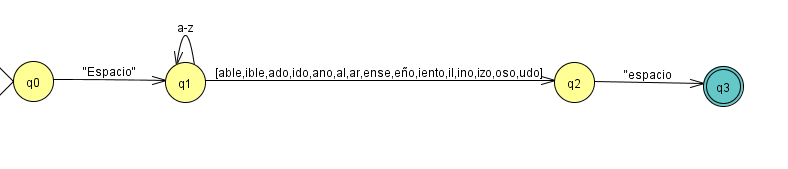
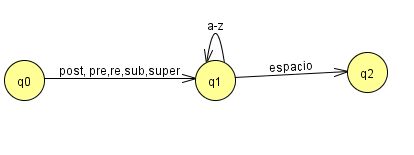
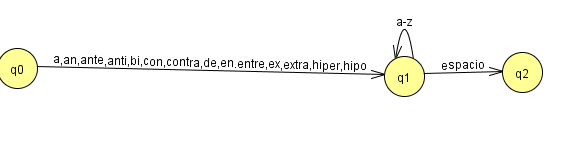
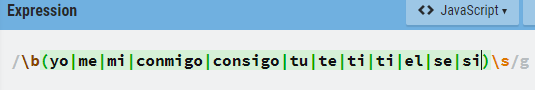
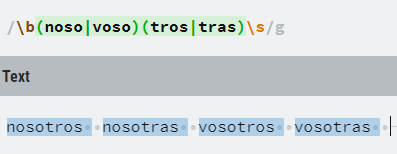
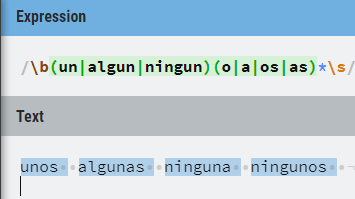
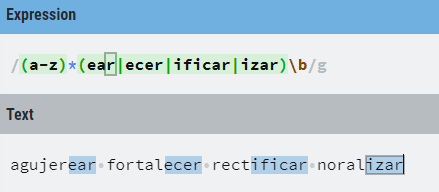
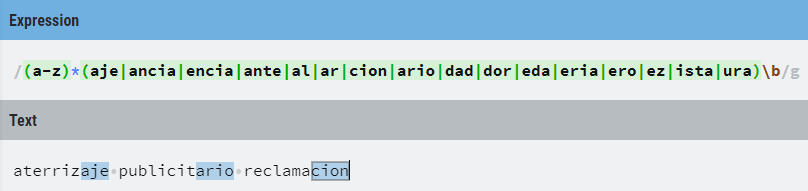
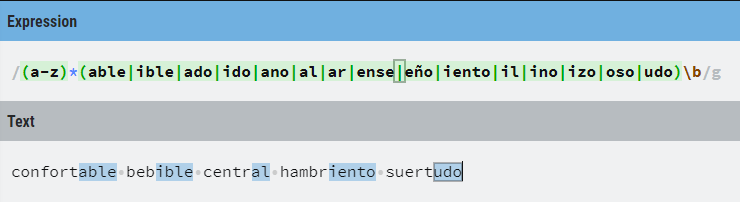
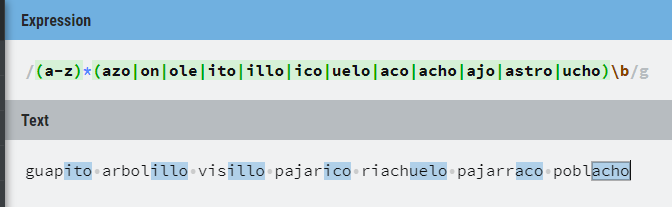
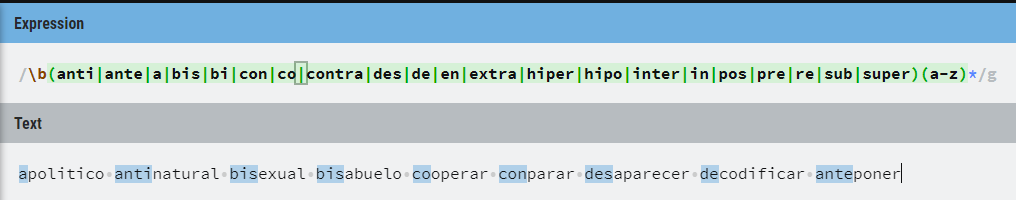
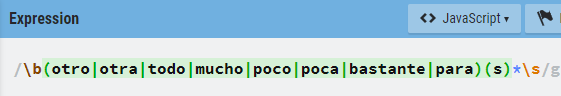
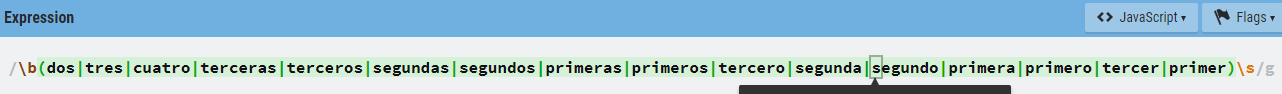
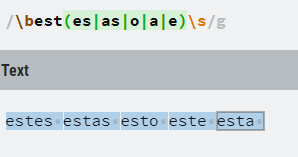
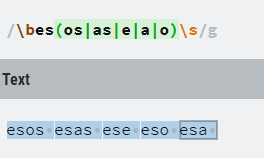
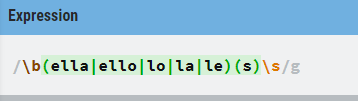
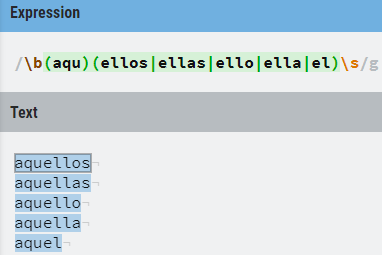
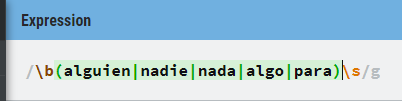
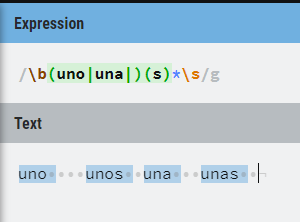
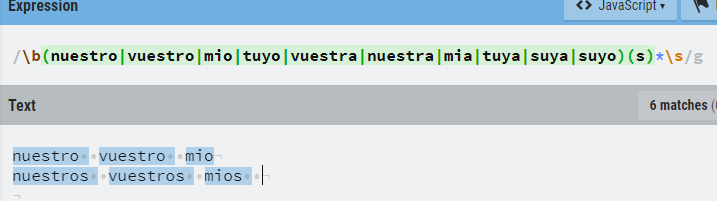
Este algoritmo nos permite realizar stemming, que es un método para reducir una palabra a su raiz, y asi poder comprender las palabras más utilizadas las que se repiten con más frecuencia para lograr ver mejor el contexto o de qué va el texto.

Text mining

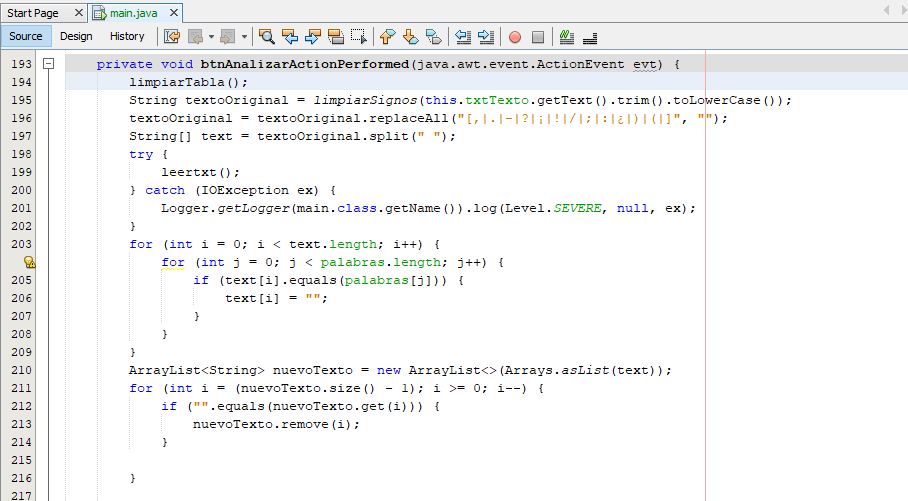
El Text-Mining surgió a principios de los 80 cuando los textos empezaban a necesitar una gran cantidad de esfuerzo humano. Pero los avances tecnológicos han permitido que este área progrese de manera muy rápida en la última década.

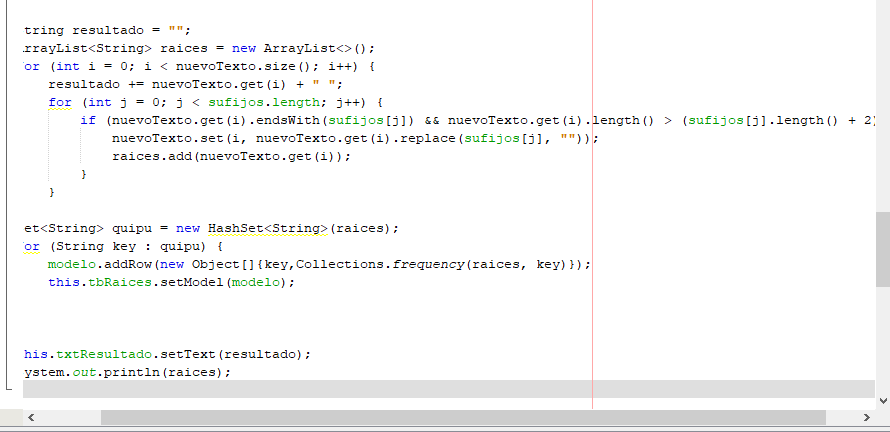
El text-Mining es el proceso de analizar colecciones de materiales textuales con el fin de capturar los conceptos y temas claves y descubrir relaciones y tendencias ocultas sin requerir que conozca las palabras o términos que los autores han usado para expresar esos conceptos.

Avance 1 y 2

Funciones relevantes





Comparación sufijos v/s prefijos

La gran diferencia que pudimos notar en la compilación del algoritmo los sufijos al ser patrones que se encuentran al final de la palabra, se hace más fácil su comprensión ya que el raciocinio humano puede reconocer mejor e interpretar de una manera adecuada la palabra que viene, para ocupar en base a información.

En el caso de los prefijos la eliminación es al principio del texto , lo cual se hace menos comprensible las palabras que queremos utilizar por ende a nuestro parecer es mas entendible o más legible la interpretación de este algoritmo aplicando sólo los sufijos.

Conclusión

El algoritmo de Porter nos permite realizar *stemming*, esto es remover los sufijos comunes morfológicos de palabras literalmente diferentes pero con un *stem* común, que pueden ser consideradas como un sólo término.

Este algoritmo requiere de un conjunto de pasos para llegar al *stem*.

Logramos el objetivo de analizar un texto mediante sus raíces y el algoritmo funcionó.

Fue posible implementar el algoritmo de porter para el texto dado.

Gracias a este desafío logramos acercarnos más al lenguaje java.

La limitada cantidad de palabras al aplicar el algoritmo de porter nos hace difícil la comprensión y la extracción total de todo el texto, además por medio de la utilización de los dos textos mostrados en el ppt también se puede decir que en los textos cortos se hace más dificultoso la comprensión de su totalidad de texto en cambio en textos más largos se pueden expresar mejor la interpretación de las palabras y poder razonar mejor el contenido de las palabras

Bibliografia

<https://medium.com/@roquelopez/algoritmo-de-porter-para-el-espa%C3%B1ol-en-java-dd44ea7b0a10>

<http://www.tyr.unlu.edu.ar/TYR-publica/debaja-Varia-Morfo.pdf>