BENEMÉRITA UNIVERSIDAD AUTÓNOMA DE PUEBLA

FACULTAD DE CIENCIAS DE LA COMPUTACIÓN

DESARROLLO DE UN MODELO PARA DETECTAR LA SIMILITUD SEMÁNTICA ENTRE TEXTOS DE DIFERENTE TAMAÑO PARA EL IDIOMA INGLÉS

TESIS PRESENTADA COMO REQUISITO PARA

OBTENER EL TITULO DE

LICENCIATURA EN CIENCIAS DE LA COMPUTACIÓN

PRESENTA:

DANTE LÓPEZ ROSAS

ASESOR:

Dra. DARNES VILARINO AYALA

NOVIEMBRE 2015

Índice de contenido

Índice de Figuras

Índice de tablas

Resumen

Una de las tareas en el procesamiento de lenguaje natural es encontrar la similitud semántica que existe entre diferentes textos, siendo que para ello se aplican modelos de similitud usando características que comparten ambos textos; sin embargo esta tarea se complica cuando estos presentan diferentes longitudes haciendo que sea necesario aplicar formas o métodos con los que sea posible evaluarlos aún con diferente tamaño. En este trabajo se presenta el desarrollo de un modelo de expansión de palabras usando contenido extraído de la web para posteriormente aplicar un modelo de similitud semántica basado en características. Para ello se hace uso de herramientas que facilitan el procesado de información textual como son Python, AWK o Wordnet. Además de hacer uso de software de aprendizaje automático para la clasificación de los datos, siendo en este caso Weka. Los resultados obtenidos en este trabajo muestran una mejora significativa en comparación a los reportados en el Semeval 2014, sin embargo siguen sin ser relevantes debido a que no se alcanza a clasificar correctamente más de la mitad de los datos.

Capítulo 1: Introducción

En este primer capítulo se explica la problemática que hay al calcular la similitud entre textos de diferente tamaño, así como las soluciones que han sido planteadas para resolver dicho problema. De igual modo se describe el objetivo que busca alcanzar este trabajo.

1.1. Planteamiento del problema

Encontrar la similitud entre pares de sentencias es muy importante en aplicaciones de procesamiento de lenguaje natural. Tareas como recuperación de información, desarrollo de resúmenes automáticos, atribución de autoría, verificación de autoría e implicación textual necesitan conocer de manera correcta un valor numérico que mida el grado de similitud entre dos textos de entrada, para inferir si ambos poseen el mismo significado.

Ahora bien, cuando los textos de entrada involucrados tienen diferente tamaño se busca encontrar el grado de similitud entre:

- Párrafo-Sentencia
- Sentencia-frase
- Frase-palabra
- Palabra-sentido

Esta problemática fue planteada por primera vez a la comunidad de investigación en procesamiento de Lenguaje Natural en el marco del Semeval 2014[1]. Se han desarrollado modelos para resolver el problema. Los resultados alcanzados cuando se hace necesario determinar el grado de similitud entre párrafo-sentencia y sentencia-frase son relevantes, ya que en el Semeval se reportan resultados de prácticamente un 80% de precisión, sin embargo no se pudo detectar correctamente el grado de similitud entre frase- palabra y mucho menos entre

[1] Semeval 2014: http://alt.qcri.org/semeval2014/

palabra –sentido, es por ello que en este trabajo se busca desarrollar un modelo que permita detectar correctamente dichos grados de similitud.

Detectar el sentido que poseen ambas sentencias es complicado cuando la mayoría de los modelos desarrollados hasta la actualidad no poseen inferencia, es decir, se buscan las características léxicas, sintácticas y semánticas que se comparten, y eso es difícil de resolver cuando ambas sentencias ofrecen diferente cantidad de información.

Para darle solución a este problema es necesario estudiar y proponer modelos que permitan expandir los textos de menor tamaño, sin incluir ambigüedad léxica, uno de los principales objetivos del presente trabajo de tesis.

1.2. Antecedentes

Hasta el año 2013, la comunidad de investigadores se dio a la tarea de encontrar la similitud semántica entre sentencias, de tamaño similar. En este sentido se han propuesto diferentes modelos basados en estrategias totalmente diferentes. Los modelos desarrollados han hecho énfasis fundamentalmente en las características que logran detectar en ambas sentencias, es decir, se ha buscado encontrar aquellos elementos comunes entre ambas sentencias que permiten medir el grado de similitud. En ese sentido se discuten los trabajos que a continuación se presentan:

En el trabajo desarrollado por **(Roth, A., Frank, A. , 2013)**, se busca establecer un modelo semántico que permita identificar los argumentos implícitos en los textos. Ya que a pesar de ser una tarea fácil para un lector humano, resulta complicado para las computadoras, debido a que no hay una manera de indicarle a éstas, que un argumento puede ser inferido varias veces en el texto. En este trabajo se propone una aproximación por inducción, que explota la información complementaria obtenida de un par de textos comparables. Esto significa que se

desarrollan modelos que permiten alinear ambos textos, esta es una técnica que se ha utilizado para el desarrollo de diccionarios estadísticos para el aprendizaje automático. Esta técnica ha permitido encontrar la similitud con un grado de precisión de un 83%, sin embrago se parte de textos que poseen aproximadamente el mismo tamaño.

Otro trabajo que es importante destacar es el desarrollado por **(Beltagy, A., .et. al., 2013)**, en el mismo se propone un modelo Markoviano para determinar la cantidad de información que ambas sentencias comparten. En el marco de esta investigación se proponen un conjunto de reglas que permiten inferir cuando dos textos poseen el mismo significado, la precisión reportada para los datos del test a partir de los modelos construidos fue de 73 %, con los datos de entrenamiento y prueba ofrecidos en el marco de la conferencia SemEval 2013.

De igual modo, se hace mención al trabajo **(Shutova, 2013)**, en el cual se muestra el primer método computacional que identifica las metáforas en textos sin restricción respecto a su interpretación, es decir se coinvierte al problema de encontrar el parecido entre textos, como encontrar si un texto es el parafraseo del otro. Se define a la interpretación de metáforas como una tarea de encontrar una paráfrasis literal para una palabra usada metafóricamente, además de introducir el concepto de parafraseado inverso simétrico como un criterio para la identificación de metáforas. Esto lo logra haciendo experimentos en los que se manejan relaciones de verbo-sujeto y sujeto-objeto indirecto, sin embargo la precisión que logran usando los datos del test y training en el SemEval 2013 es solamente del 66%.

En el trabajo desarrollado por **(Salehi, B., Cook, P., 2013)**, se plantea que para resolver este problema es necesario representar los textos no con términos unipalabras, sino con términos multipalabras. Para encontrar la similitud en este tipo de representación acuden a traducciones automáticas, para lo que usan a la herramienta PanLex, la cual le permite la creación de un diccionario estadístico.

Si la traducción es posible, esto quiere decir que es equivalente un término en un texto, con una expresión multipalabra en el otro. El sistema ofreció una precisión del 66 % en el marco del Semeval 2013.

Otro trabajo al que se puede hacer mención es el desarrollado por **(Schwartz, H., et.al., (2013)),** cuyo objetivo es caracterizar los errores que se producen al realizar conteo de palabras, además de mostrar que los conteos léxicos pueden ser significativamente mejorados mediante el empleo de medidas de ambigüedad. Es decir encontrar sobre el texto aquellas palabras ambiguas, eliminar aquellas que no están totalmente relacionadas con el contexto. Esta propuesta no brindó resultados significativos, ya que solamente alcanzó el 50% de precisión.

Por otro lado, en el trabajo desarrollado por **(Horbach, A., et.al., 2013)** se maneja el concepto de los ejercicios de traducción en la enseñanza de los idiomas, por lo que se explora el papel que tiene el texto para evaluar las respuestas cortas, es decir, una oración es la pregunta y la otra es la respuesta, el grado de similitud es medido en función si la respuesta logra responder a la pregunta. Ver este problema con el enfoque de Question Answering les permitió obtener un 73 % de precisión.

A pesar de todos los esfuerzos realizados ninguna de las propuestas ha logrado superar el 85 % de precisión. Esta situación se complica aún más, cuando los textos a comparar no poseen el mismo tamaño, es decir una cantidad de información similar, es por ello que en la presente investigación se pretende desarrollar un modelo que permita encontrar la similitud semántica entre textos de longitud totalmente diferente y con un sentido distinto, como es palabra-sentido.

1.3. Objetivos

- **Objetivo General**:
  Desarrollar un modelo para determinar la similitud semántica que existe entre textos de diferente tamaño.
- **Objetivos Específicos:**
  - Estudiar los trabajos publicados en el marco teórico del SemEval 2012 y SemEval 2013.
  - Determinar las características a extraer para cada par de textos.
  - Desarrollar un modelo de expansión para encontrar la similitud entre frase-palabra.
  - Desarrollar un mecanismo para lograr detectar la similitud entre palabra sentido, incluyendo métodos que impidan introducir ambigüedad en los textos.
  - Desarrollar un modelo supervisado utilizando los datos Training, dados en el marco del SemEval 2014.
  - Validar el modelo confeccionado con los datos del test ofrecidos en el marco de la misma conferencia.

1.4. Justificación

Como es mencionado en el libro Natural Language Processing whit Python **(Bird, S., et al, 2009)** , las tecnologías basadas en Procesamiento de Lenguaje Natural están siendo cada vez más generales. Por ejemplo teléfonos o dispositivos de bolsillo ahora soportan predicción de texto y reconocimiento de escritura; la ingeniería de búsqueda en la web proporciona acceso a información que se encuentra dentro de texto no estructurado; así mismo las máquinas traductoras nos permiten leer texto entre diferentes idiomas. Por lo anteriormente mencionado, aunado a otras aplicaciones, el procesamiento de lenguaje juega un rol importante en el tratamiento de información multilenguaje en la sociedad.

Ahora bien, sabiendo que la mayor parte de la información se almacena y maneja en forma de texto, se diseñan diferentes formas y medios para transmitirla e interpretarla, usando para esto último el procesamiento de lenguaje natural, sin embargo a pesar de los grandes avances tecnológicos esta área sigue siendo complicada puesto que el tratamiento de la información textual abarca aplicaciones como sistemas de recomendación, detección de plagios, elaboración de resúmenes automático, detección de palabras clave, extracción de conocimiento web, entre otros, y para las cuales hay que realizar tareas como encontrar la similitud que existe en los textos, sin embargo a pesar de que un humano puede hacer esta tarea de forma intuitiva para el ámbito computacional es necesario indicar los métodos que se han de usar para detectar dichas similitudes, es por ello que se diseñan modelos de similitud basándose en características específicas, por lo que la problemática de esta tarea se presenta debido a la complejidad que existe en los idiomas y en la cantidad de texto que se tiene para calcular la similitud. Es por ello que el presente trabajo tiene el fin desarrollar un modelo de expansión así como un modelo de similitud semántica textual que permita medir la similitud entre textos de diferente tamaño.

1.5. **Organización de la Tesis**

El presente trabajo de tesis está formado por cuatro capítulos que se enuncian a continuación:

1. Introducción: este capítulo se describe de forma general el problema que se quiere resolver, las soluciones o antecedentes que han sido propuestos y los objetivos de este trabajo.
2. Marco Teórico: en este capítulo se describen las herramientas y bases teóricas usadas para desarrollar este trabajo.
3. Metodología: se describe como se usaron las herramientas y los pasos usados para este trabajo.
4. Análisis de Resultados: se hace una comparativa con los resultados arrojados por este trabajo y con los reportados por el Semeval 2014.
5. Conclusiones y recomendaciones: se explican posibles mejoras y trabajos futuros para este trabajo.

Capítulo 2: Marco Teórico

En este capítulo se describen las herramientas y los elementos teóricos que sirvieron de base para desarrollar el presente trabajo de investigación.

2.1. Herramientas y recursos

Para el desarrollo del presente trabajo se hizo uso de herramientas cuyas funciones permitieran el procesamiento de información textual, las cuales se describen a continuación:

2.1.1. Python

Existen múltiples lenguajes de programación, muchos de ellos pueden ser usados en variadas tareas, sin embargo su facilidad de uso en una tarea en específico es lo que muchas veces marca la pauta de cuál lenguaje debe de ser usado, y dentro del área del procesamiento de lenguaje natural, Python es uno de los mejores que pueden ser usados. El logotipo del lenguaje se aprecia en **Figura 2.1**

Figura 2.1 Logotipo del lenguaje de programación Python

Python es un lenguaje de programación de alto nivel, cuyo enfoque se centra en la productividad del programador y la legibilidad del código. Este lenguaje de programación soporta diferentes paradigmas de programación como la orientada a objetos, la imperativa o la funcional.

Una de las grandes ventajas que tiene Python es la definición de objetos incorporados como son listas, tuplas, tablas hash; además de permitir el manejo de cadenas de forma sencilla.

Python fue publicado por primera vez por Guido van Rossum en 1991. El lenguaje tiene un modelo abierto de desarrollo basado en la comunidad administrado por la organización sin fines de lucro Python Software Foundation. **(Di Pierro, 2011).**

Los motivos por los cuales se seleccionó este lenguaje de programación para trabajar son debido a que como menciona **(Bird, S., et al, 2009)** "Python es una curva de aprendizaje superficial, su sintaxis y semántica son transparentes, y tiene un buen manejo de funcionalidades con cadenas".

De igual modo, Python provee de un gran número de librerías que brindan funciones para diferentes aplicaciones como web, gráficas o textuales. Así mismo las estructuras de datos son de fácil manejo, siendo esto de gran utilidad ya que en el procesamiento de texto es común almacenarlo en diferentes estructuras dependiendo de la tarea que se esté realizando.

2.1.2. Natural Language Toolkit (NLTK)

NLTK es la plataforma líder para construir programas en Python y trabajar con datos del lenguaje humano. Provee interfaces fáciles de usar de más de 50 corpus y recursos léxicos como WordNet, junto con una suite de librerías de procesamiento de texto para clasificación, tokenización, lematización, etiquetado, parsing, y razonamiento semántico. **(NLTK Project, 2015).** Originalmente fue diseñado para la enseñanza, sin embargo ha sido adoptado en la industria para la investigación y desarrollo debido a su utilidad.

Algunas de las tareas que se pueden realizar con NLTK son las siguientes:

- Tokenización de grandes textos en oraciones.

- Tokenización de oraciones en sus palabras.
- Filtrado y creación de conjuntos de palabras cerradas en diferentes idiomas
- Mejoramiento en los lemas y sinónimos en WordNet

Existen más funciones para el procesamiento de lenguaje natural, sin embargo en este trabajo se hará principalmente uso de las mencionadas hace un momento.

2.1.3. Wordnet

WordNet es una base de datos léxica electrónica para el idioma Ingles; ésta fue realizada en 1986 en la Universidad de Princeton. Sustantivos, verbos, adjetivos y adverbios son agrupados dentro de conjuntos cognitivos (synsets), siendo que cada uno expresa un concepto diferente. Dichos synsets son interrelacionados por significados de conceptos semánticos y relaciones léxicas. De igual modo, la estructura que posee WordNet lo hace una herramienta muy útil para tareas computacionales lingüísticas y para el procesamiento de lenguaje natural.

WordNet superficialmente parece un diccionario, en el que sus grupos de palabras están basadas en sus significados, sin embargo hay dos importantes diferencias. Primero, WordNet interrelaciona no sólo la forma de las palabras, sino también el sentido específico de ellas. Como resultado, las palabras que son encontradas con una cercana proximidad entre una y otra son semánticamente desambiguas. Segundo, WordNet etiqueta las relaciones semánticas entre palabras, mientras el agrupamiento de palabras en el diccionario no sigue ningún patrón en específico más que similitud de significado. **(Princeton University, 2015).**

En este trabajo se usó principalmente para los modelos de similitud semántica, aparte de la lematización en el modelo de expansión.

2.1.4. Beautiful Soup

Beautiful Soup es una librería de Python diseñada para proyectos que utilizan contenido web HTML o XML. Tres características lo hace una herramienta poderosa:

1. Beautidul Soup provee de métodos simples e instrucciones para la navegación, búsqueda, y modificación de árboles de búsqueda.
2. Beautiful Soup convierte automáticamente los documentos entrantes a formato Unicode para posteriormente devolverlos a un formato UTF-8, con lo cual se evita problemas con el codificado de los documentos.
3. Beautiful Soup forma parte de los principales parsers de Python, por lo que es posible probar diferentes estrategias de parseo o velocidad. **(Crummy, 2015).**

Debido a la gran cantidad de información que fue extraída de la web, se hizo uso de esta librería para obtener textos planos sin ninguna etiqueta que pudiera afectar el modelo de expansión.

2.1.5. Weka

Weka es una extensa colección de algoritmos de máquinas de conocimiento desarrollado por la universidad de Waikato (Nueva Zelanda) implementados bajo el lenguaje Java, útiles para ser aplicados sobre datos mediante interfaces que ofrece o para embeberlos dentro de cualquier aplicación. Además Weka contiene herramientas necesarias para realizar transformaciones sobre los datos, tareas de clasificación, regresión, clustering, asociación y visualización. Weka está diseñado como una herramienta orientada a la extensibilidad por lo que añadir nuevas funciones es una tarea sencilla.

La licencia de Weka es GPL, lo que significa que este programa es de libre distribución y difusión. Además, ya que Weka está programado en Java, es

independiente de la arquitectura, ya que funciona en cualquier plataforma sobre la que haya una máquina virtual de Java disponible. **(Garcia, s.f).**

Figura 2.2 Logotipo de Weka

Para este trabajo se utilizó para la clasificación de los resultados usando la técnica de regresión lineal. El logotipo de Weka puede se puede apreciar en la **Figura 2.2**.

2.1.6. AWK

AWK es un lenguaje de programación que está especialmente diseñado para trabajar con archivos estructurados y patrones de texto. Dispone de características internas para descomponer líneas de entrada en campos y comparar estos campos con patrones que se especifiquen. Debido a estas posibilidades, resulta particularmente apropiado para trabajar con archivos que contienen información estructurada en campos, como inventarios, listas de correo, y otros archivos de bases de datos simples. […]

La función básica de AWK es buscar líneas en ficheros (u otras unidades de texto) que contienen ciertos patrones. Cuando en una línea se encuentra un patrón, AWK realiza las acciones especificadas para dicho patrón sobre dicha línea. AWK sigue realizando el procesamiento de líneas de entrada de esta forma hasta que se llega al final del fichero. **(Vidal, J., 2002).**

Para este trabajo AWK tuvo una vital importancia, puesto que fue gracias a este lenguaje que se pudo extraer contenido web, además de que algunas características de similitud fueron programadas bajo este lenguaje de programación.

2.2. Selección de características

En general a lo largo de la literatura se ha resuelto el problema extrayendo características que comparten ambas sentencias, con el objetivo de lograr descubrir el grado de similitud que poseen. Entre las características léxicas más relevantes se tienen n-gramas (predicción de la próxima palabra de una oración basándose en las n-1 anteriores) y skip-gramas (predicción de la próxima palabra de una oración basándose en las anteriores, pero omitiendo aleatoriamente algunas) tanto de caracteres como de palabras, y la presencia o ausencia de sinónimos e hiperónimos.
Otra de las medidas consideradas es el coeficiente de similitud de Jaccard entre dos textos y la similitud coseno entre los dos textos representados cada uno de ellos por la bolsa de los n-gramas y la bolsa de los skip-gramas de caracteres, respectivamente.

El segundo conjunto de características considera las 6 medidas de similitud de palabras ofrecidas por la herramienta NLTK: Leacock & Chodorow **(Leacock, C., Chodorow, M., 1998)**, Lesk **(Lesk, M., 1986)**, Wu & Palmer **(Wu, Z., Palmer, M., 1994)**, Resnik **(Resnik,P., 1995)**, Lin **(Lin, D., 1998)**, y Jiang & Conrath **(Jiang, J., Conrath, D., 1997)**. En este caso, se determina la similitud semántica entre dos textos como el máximo valor de similitud obtenido entre los pares de palabras.

El tercer conjunto de características considera dos medidas basadas en corpus, ambas métricas utilizan la métrica de similitud semántica textual ofrecida por Rada Mihalcea **(Mihalcea, R., et al., 2006)**. La primera usa información mutua (PMI) **(Turney, P., 2001)** para el cálculo de la similitud entre pares de palabras,

mientras que la segunda utiliza análisis semántico latente (LSA) **(Landauer, P., et.al., 1998)**.

En la **Tabla 2.1** se listan las características usadas para encontrar la similitud semántica textual

Tabla 2-1 Caracteristicas Usadas

Característica	Tipo
n-gramas de caracteres (n=2,…,5)	Léxica
Skip-gramas de caracteres (skip=2,…,5)	Léxica
Número de palabras que comparten	Léxica
Número de sinónimos que comparten	Léxica
Número de hiperónimos que comparten	Léxica
Coeficiente de Jaccard con expansión de sinónimos	Léxica
Similitud coseno con n-gramas y skip-gramas de caracteres	Léxicas
Similitud de palabra de Leacock & Chodorow	Basada en conocimiento
Similitud de palabra de Lesk	Basada en conocimiento
Similitud de palabra de Wu & Palmer	Basada en conocimiento
Similitud de palabra de Resnik's	Basada en conocimiento
Similitud de palabra de Lin	Basada en conocimiento
Similitud de palabra de Jiang & Conrath	Basada en conocimiento
Métrica de Rada Mihalcea usando información mutua	Basada en Corpus
Métrica de Rada Mihalcea usando LSA	Basada en Corpus

2.3. Técnica de clasificación seleccionada

El tratar con el procesamiento de lenguaje natural implica a su vez el uso o creación de las conocidas máquinas de aprendizaje automático de la inteligencia artificial, siendo el usado en este trabajo el aprendizaje supervisado; sin embargo antes de explicar a detalle el tipo de técnica que se usó, es conveniente aclarar algunos conceptos que permitan su correcta comprensión.

2.3.1. Máquinas de aprendizaje automático

Dentro de la inteligencia artificial hay una gran cantidad de categorías de problemas, siendo uno de estos las máquinas de aprendizaje automático, cuyo principal objetivo es diseñar algoritmos que permitan a la computadora aprender. **(aihorozin, 2015).**

Estas máquinas se dividen principalmente en las de aprendizaje supervisado y las de aprendizaje no supervisado. De forma muy intuitiva podría deducirse que su principal diferencia es que en una interviene el ser humano, mientras que en la otra no, sin embargo no es del todo así. A continuación se describen sus características principales para tener un concepto más claro de estas:

- **Máquinas de aprendizaje supervisado**

  El aprendizaje supervisado es bastante común en los problemas de clasificación debido a que el objetivo es conseguir que la computadora aprenda un sistema de clasificación que se ha creado. De forma más general el aprendizaje por clasificación es apropiado para cualquier problema donde deducir una clasificación sea útil y que su clasificación sea fácil de determinar.

  De igual modo el aprendizaje supervisado es la técnica más común para entrenar redes neuronales y árboles de decisión. Ambas técnicas son altamente dependientes de la información dada por clasificaciones pre-

determinadas. En el caso de las redes neuronales, la clasificación es usada para determinar el error de la red y después ajustar la red para minimizarlo; por otro lado, en los árboles de decisión, las clasificaciones son usadas para determinar qué atributos proveen la mayor información que pueda resolver el problema del rompecabezas de la clasificación.

Otra técnica muy usada en los problemas de clasificación son las máquinas de soporte vectorial, en su texto, **Colmenares (s.f.)** nos menciona que la teoría de las máquinas de soporte (abreviadas regularmente como SVM por sus siglas en inglés Support Vector Machine) fue desarrollada por Vapnik basado en la idea de minimización del riesgo estructural (SRM).

Así mismo nos menciona que algunas de las aplicaciones de clasificación o reconocimiento de patrones son: reconocimiento de firmas, reconocimiento de imágenes, la categorización de textos, entre otros.

De igual modo nos indica que a diferencia de las redes neuronales, las cuales utilizan durante su entrenamiento el principio de minimización de riesgo empírico, las máquinas de soporte vectorial se basan en el principio de minimización de riesgo estructural, es decir, minimizan un límite superior al riesgo esperado en vez de minimizar el error sobre los datos de entrenamiento.

Ahora bien, mencionando otra técnica de clasificación, se tiene la regresión lineal, la cual es la usada en este trabajo, por lo que se explicara a más detalle posteriormente.

Por último hay que destacar que en los problemas de clasificación, el objetivo de los algoritmos es minimizar el error respecto a las entradas dadas, las cuales frecuentemente son llamadas “training-set”, que en si son ejemplos de los cuales se trata de aprender; en este trabajo eso vendría siendo el conjunto de datos brindado por SemEval.

Un problema presente en este tipo de máquinas de aprendizaje y sus training-sets es el hecho de que si estos no están clasificados correctamente pueden surgir diferentes dificultades, incluso si el algoritmo esta optimizado para casos especiales. **(aihorizon, 2015).**

- **Máquinas de aprendizaje no supervisado**

  El aprendizaje no supervisado parece aún más difícil, puesto que su objetivo es enseñar a la computadora a hacer algo sin que se le diga cómo, siendo esto la principal diferencia que existe con el aprendizaje supervisado.

  Existen dos formas principales en las que este tipo de máquinas son entrenadas; la primera se basa en la premiación, es decir, la máquina es premiada dependiendo de las acciones que tome, logrando de este modo que se aprendan las formas para obtener cierto premio. De igual modo con este tipo de entrenamiento se pueden aplicar castigos por las acciones. Lo anterior lleva a que debido a que se conoce la función que brinda cierto premio, la maquina simplemente aprende qué hacer sin necesidad de algún procesamiento, lo cual es benéfico debido a que se ahorran el cálculo de todas las posibilidades, sin embargo, al mismo tiempo, puede resultar tardado debido a que se aprende a prueba y error.

  El segundo tipo, son los famosos clusterings, cuyo objetivo no es maximizar la función de utilidad, en vez de eso la simplifica para encontrar similitudes en los datos de entrenamiento.

  De cualquier modo, en el aprendizaje no supervisado, se requiere de la intervención humana más que nada para la interpretación de resultados, puesto que, como se mencionó anteriormente, simplemente se le ha indicado a la maquina hacer algo, mas no cómo hacerlo, por lo que se necesita verificar si el resultado obtenido es el deseado. **(aihorizon,2015).**

Con la información brindada hasta el momento ya es posible la comprensión de las máquinas de aprendizaje automático, por lo que a continuación se procederá a explicar la técnica que se usó para la clasificación.

2.3.2. Regresión lineal

Para este trabajo se usó regresión lineal, dicha técnica es del tipo supervisado, puesto que necesita de datos de entrenamiento pre-etiquetados que le permitan aprender a clasificar las características de los datos.

El análisis de regresión es una técnica para investigar y modelar la relación entre variables. Tiene diferentes aplicaciones y ocurre en diferentes áreas como la física, las ciencias sociales, economía, entre otras.

Entre las utilidades que presenta esta técnica, **Mendoza H, et.al. (2015)** nos menciona las siguientes:

- Descripción de datos: frecuentemente se utilizan ecuaciones para resumir un conjunto de datos.
- Estimación de parámetros: existen varios casos en los que es necesario estimar parámetros, uno de estos podría ser en un circuito eléctrico, donde se tiene una resistencia de *R* ohms, además de que pasan diferentes corrientes, por lo que el voltaje es medido. Usando un diagrama de dispersión se podría indicar que el voltaje y la corriente están relacionados por una línea recta que pasa por el origen con pendiente *R*. Haciendo un análisis de regresión es posible estimar el valor de la resistencia desconocida ajustando el modelo para lograrlo.

  Lo anterior es sólo un ejemplo, sin embargo puede ser aplicado en diferentes áreas dependiendo de lo que se desee saber.
- Predicción y estimación: al igual que el caso anterior, depende de lo que se desea saber, por ejemplo para saber la respuesta de un cultivo al variar la cantidad de fertilizantes con el objetivo de establecer una relación o predecir la combinación optima de fertilizantes.

Ahora, qué es exactamente regresión lineal; pues bien en el análisis de datos, se denomina regresión al análisis de la relación entra una variable ***y*** denominada

variable dependiente y un conjunto de otras variables denominadas independientes $x_1, x_2 \ldots x_p$.

Su objetivo es predecir el valor de la variable dependiente en función de la variable independiente. De igual modo se supone que ***x*** e ***y*** son variables continuas, siendo que si se conoce ***x*** es posible conocer el valor de ***y*** a excepción de un posible ruido aleatorio ***ε.***

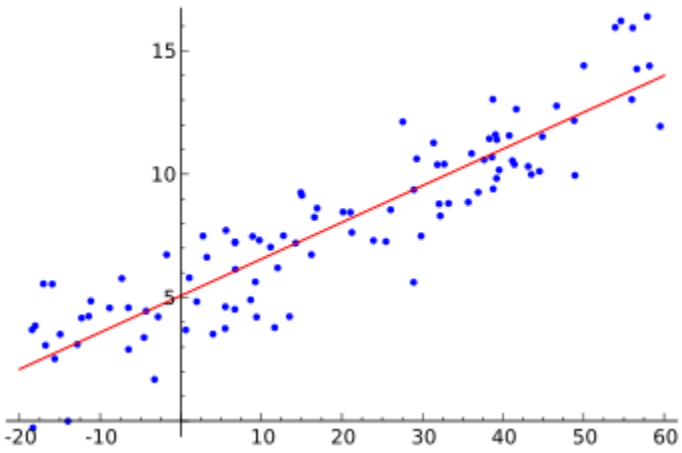
Con lo anterior mencionado se tiene el siguiente modelo:

$$y = f(x) + \varepsilon$$

Regularmente en un modelo de regresión lineal se supone que la relación ***f*** entre ***x*** e ***y*** es lineal, es decir con una sola variable independiente.

Un ejemplo de cómo se representa la regresión lineal gráficamente se ve en la **Figura 2.3.**

Figura 2.3 Ejemplo de representación de regresión lineal



En su texto, **Ñanculef (2007)**, nos menciona que la clasificación mediante la regresión lineal es posible debido a que la idea para llevarlo a cabo es:

- Asignar labels numéricos a las clases (etiquetar las clases)

- Aplicar la regresión lineal usando como variable independiente estos labels
- Ajustado el modelo ***f***, la regla de clasificación consiste en elegir el label más cercano a la predicción del modelo ***f(x)***.

Es necesario tener en cuenta que los números que se asignan a cada clase son arbitrarios, por lo que en la naturaleza rígida de la regresión lineal sólo codificaciones especiales darían buenos resultados.

Teniendo en cuenta lo anterior, es necesaria una re-codificación, para ello las clases se codifican mediante vectores binarios. Si la clase asociada a un x_i es la clase k entonces la variable independiente que se le asocia es:

$$y_i = \begin{pmatrix} 0 \\ 0 \\ \dots \\ 1 \\ \dots \\ 0 \end{pmatrix} \rightarrow \boldsymbol{en\ la\ k-}\text{ésima }\boldsymbol{posición}$$

Es decir, se tiene que $y_i=1$ siempre y cuando x_i sea de la clase j.
Con lo anterior se obtiene un modelo de regresión lineal multivariada con tantas salidas como clases se tengan, por lo que se tiene un modelo lineal f_k por cada clase.

Una vez que se ajusta el modelo, la regla de clasificación consiste en elegir la clase k tal que $f_k(x)$ es máximo, es decir donde alcanza su máximo valor.

$$clase\ para\ x = argmax_k f_k(x)$$

Hablar acerca de la regresión lineal comprende introducirse más a profundidad en el tema, lo cual no es el objetivo de este trabajo, por lo que se espera que con lo explicado sea un poco más comprensible esta técnica.

Capítulo 3: Metodología

En este capítulo se describe la metodología propuesta para este trabajo, la cual tiene como objetivo lograr expandir el tamaño de las sentencias de menor longitud. La idea consiste principalmente en la expansión de texto usando información extraída de la web, siendo que en el caso de la similitud entre frase-palabra, se extrae contenido relacionado a la palabra y posteriormente se expande con el mismo; mientras que en la similitud entre palabra-sentido, se extrae contenido tanto de la palabra, como del sentido para después expandir.

3.1. Modelo de expansión de palabras

Como se mencionó anteriormente, uno de las principales dificultades al momento de buscar similitud semántica entre dos textos es la cantidad de información que se tiene de ellos para compararlos. Para resolver este problema se planteó diseñar un modelo de expansión que fuera capaz de ampliar la información de una palabra a través de contenido web.

De forma muy general las fases usadas para la ampliación de información fueron las siguientes:

- Extracción de las palabras a expandir de los datos de prueba brindados por SemEval.
- Búsqueda y extracción de contenido web
- Limpieza del contenido web
- Aplicación del modelo expansión
- Creación de nuevo corpus

A continuación se explica cada una de las fases:

3.1.1. Extracción de las palabras a expandir de los datos de prueba brindados por SemEval

Se han usado 2 corpus correspondientes a la tarea Cross Level Semantic Similarity del SemEval 2014. De cada uno de los corpus se dispone de 500 pares, es decir se tiene un corpus para detectar la similitud entre palabra-sentido, y frase-palabra. Se realiza la expansión de cada uno de los pares, buscando que el tamaño de los textos que es necesario evaluar sea de tamaño similar y se maneje mayor cantidad de información.

3.1.2. Búsqueda y extracción de contenido web

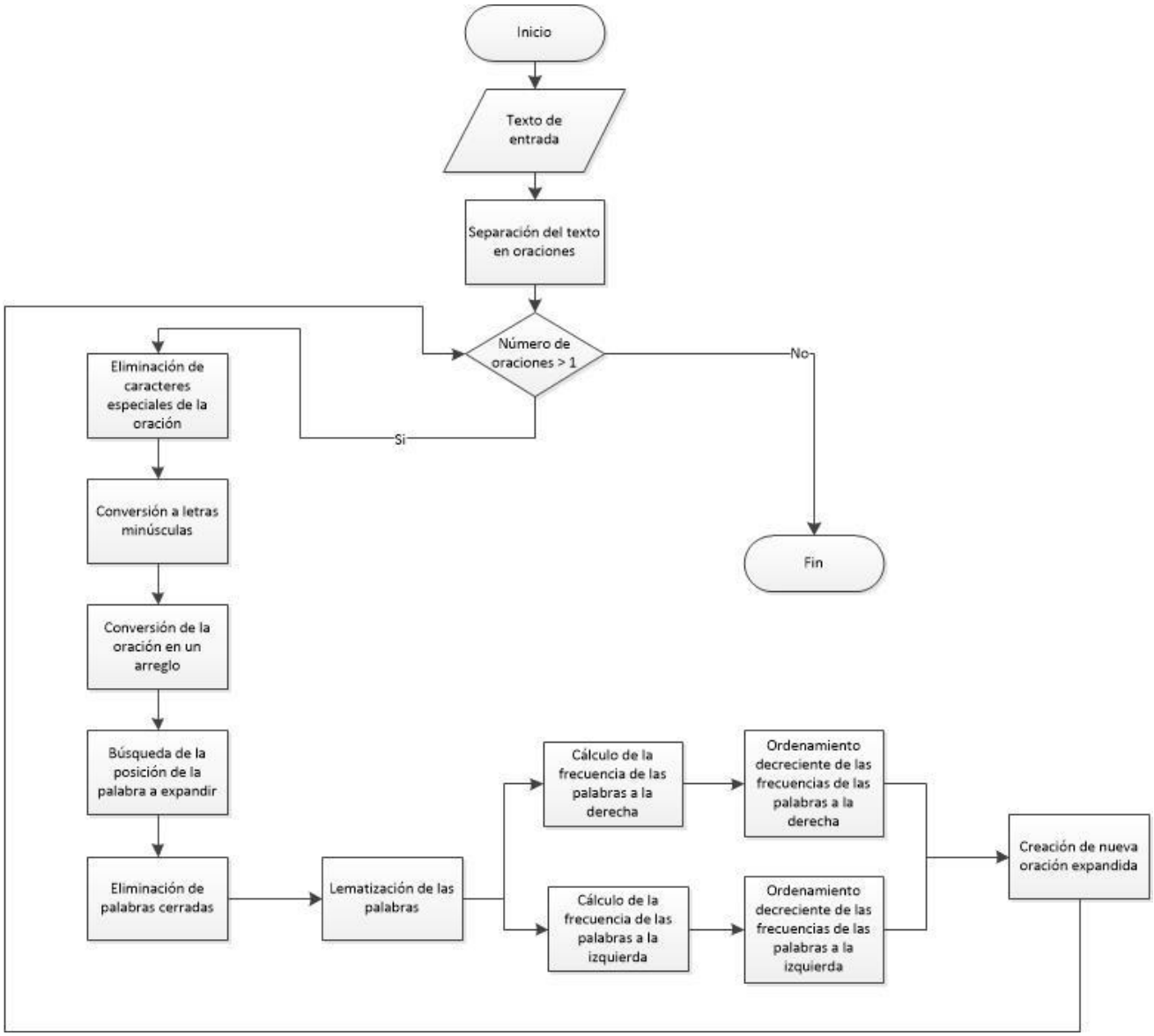
Una vez que se hizo la separación del texto de los corpus, se procedió a la extracción de contenido web, para ello se realizó un programa en Awk que permitiera buscar y descargar un conjunto de páginas en las que apareciera la palabra a buscar, esto se llevó a cabo para cada una de las palabras que forman parte de cada uno de los corpus y que es necesario expandir.

3.1.3. Limpieza del contenido web

Teniendo ya la información extraída de la web, se procedió a la limpieza de la misma, para ello se realizó un programa en Python haciendo uso de la librería Beautiful Soup y así quitar todo el etiquetado de las páginas, además de dar el formato deseado para aplicar la expansión a las palabras.

3.1.4 Aplicación del modelo de expansión

Figura 3.1 Diagrama que muestra el proceso de expansión



Una vez que el texto estuvo libre de etiquetado se procedió a aplicar el modelo de expansión, el cual está diseñado para crear una nueva oración o frase basándose en las palabras más frecuentes que aparecen junto con la palabra a expandir.

La idea del modelo de expansión es tomar las cinco palabras más frecuentes a la derecha y las cinco más frecuentes a la izquierda de la palabra a expandir, para

después formar una nueva oración con la que se tendrá más información y con ello sea posible aplicar el modelo de similitud semántica.

En la **Figura 3.1** se presenta de forma muy general la idea del funcionamiento del modelo de expansión.

A continuación se describe de forma detallada lo que se realiza en el modelo de expansión:

3.1.4.1. Separar los textos extraídos

Al momento de la eliminación del etiquetado de las páginas extraídas de las web, se deja preparado el texto con un formato necesario para aplicar correctamente el modelo de expansión. Dicho formato consiste en dejar la información de las páginas en una sola línea separadas por el caracter **"|"**. Esto permite que posteriormente el programa pueda separar el contenido en cadenas tomando como referencia dicho carácter.

3.1.4.2. Eliminación de espacios y caracteres

Una vez separado el contenido en cadenas, para cada una de ellas se elimina los espacios al inicio y al final, así como símbolos y signos de puntuación que pueden alterar la expansión.

3.1.4.3. Cambio a minúsculas y separación a palabras

Posteriormente cada cadena es transformada a letras minúsculas, con el objetivo de que el programa no tome como diferentes a palabras que estén escritas de igual manera, pero que se diferencien por alguna letra mayúscula. Luego la cadena resultante es separada nuevamente tomando como referencia el espacio

en blanco, es decir, se separada en sus palabras, las cuales son guardadas en un arreglo.

3.1.4.4. Determinar posición de la palabra a expandir

Inmediatamente después se busca en el arreglo la posición de la palabra que se desea expandir. Esto para que el programa pueda buscar hacia ambos lados posteriormente. Si llegase a existir en diferentes posiciones la palabra a extender se toma la primera ocurrencia. Esto, en cierta forma es una limitación del programa puesto que es posible que la palabra aparezca en las primeras posiciones del arreglo, lo que limitaría la expansión.

3.1.4.5. Expansión de la palabra

Una vez que se ha encontrado la posición, se procede a buscar la frecuencia de palabras hacia la izquierda y la derecha. Para asegurar que dichas palabras no se repitan se hace uso de diccionarios, uno para cada lado, en los que se guardan y contabilizan para determinar su frecuencia de aparición. Algo que hay que destacar es que antes de guardar las palabras se verifica que no sean cerradas, es decir, palabras que carecen de significado por si solas (artículos, pronombres, preposiciones, etc.), un ejemplo de dichas palabras se muestra en la **Figura 3.2**.

Para lograr esto se hace uso del conjunto de palabras cerradas en inglés que ofrece Wordnet, es decir, si la palabra a guardar existe en el conjunto de palabras de dicha librería, no son agregadas a los diccionarios.

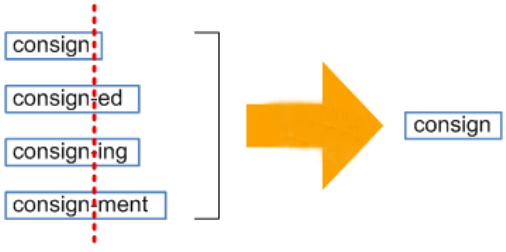
Figura 3.2 Ejemplo de lista de palabras cerradas en inglés

Stopwords		
a	it	these
about	its	they
again	itself	this
all	just	those
almost	kg	through
also	km	thus
although	made	to
always	mainly	upon
among	make	use
an	may	used
and	mg	using
another	might	various
any	ml	very
are	mm	was
as	most	we
at	mostly	were

De igual modo se considera que las palabras se encuentran en alguna de sus formas flexionadas, es decir que posee prefijos, sufijos, infijos, etc. de modo que se altera su significado (singular, plural, masculino, femenino), por lo que es necesario recurrir a la lematización, es decir, se hace un proceso de reducción para hallar el lema, el cual es la forma canónica en la que se reduce todo un paradigma flexivo y que se toma como representante de todas las variaciones morfológicas de la palabra. **(Hispanoteca, 2015).**

Un ejemplo del proceso de lematización se muestra en la **Figura 3.3.** Para lograr esto dentro del código se hace uso del paquete nltk.stem, el cual incluye un lematizador basado en Wordnet y que regresa el lema de menor longitud.

Figura 3.3 Ejemplo de lematización de una palabra respecto a sus formas flexionadas



Una vez que se ha terminado de analizar, procesar y contabilizar las palabras, se procede a verificar cuáles son las cinco palabras que tienen una aparición más frecuente hacia la derecha y cuáles son las cinco palabras que tienen una aparición más frecuente hacia la izquierda, por lo que se hace un ordenamiento de los diccionarios en modo decreciente de aparición de las palabras para que posteriormente sean tomadas las primeras cinco de cada diccionario. Esto puede llegar a ser un problema puesto que es posible que varias palabras tengan la misma frecuencia de aparición, sin embargo son descartadas debido a que no se encuentran dentro de las cinco primeras posiciones de los diccionarios. Así mismo al hacer el ordenamiento se pierde el orden de aparición de las palabras, por lo que igual puede llegar a representar un problema sintáctico en la aplicación de los modelos de similitud.

3.1.4.6. Creación de nueva oración

Finalmente teniendo las palabras frecuentes de la derecha y la izquierda, se crea una sola oración con ellas conservando el lado al que correspondían de la palabra que se deseaba expandir, siendo así que se crea una expansión de información basándose en contenido web.

Finalmente esto se repite cada una de las palabras que se desean expandir.

Lo anterior se lleva a cabo para los corpus de frase-palabra y palabra-sentido, siendo que en este último se aplica a dos de las columnas de palabras, mientras que en el primero sólo se aplica a la columna correspondiente a la palabra.

3.1.5. Creación de los nuevos corpus

Teniendo las palabras ya expandidas se crean nuevos corpus de entrenamiento que deben de ser ajustados para que se les pueda aplicar los modelos de similitud semántica textual. Para lograr esto se diseñó un programa en Python en el que se acomodaran las frases expandidas de modo tal que pudieran ser comparadas, además de tener su escala de similitud brindada por el SemEval 2014.

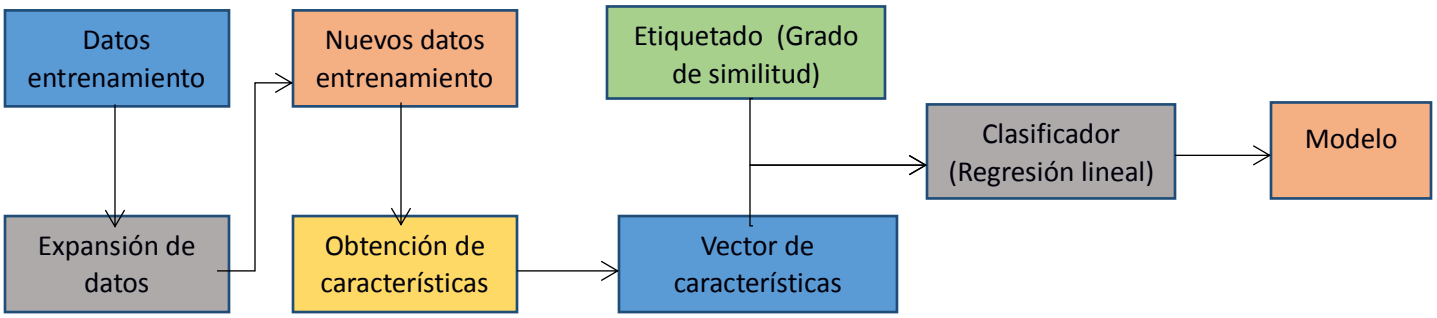
3.2. Modelo para detectar la similitud semántica.

Una vez que se han expandido los textos usando la propuesta desarrollada, se procede a extraer las características mencionadas anteriormente, primero sobre los datos de entrenamiento y luego sobre los datos de prueba.

Al aplicar las características sobre los datos de entrada, se obtienen datos numéricos, los cuales es necesario acomodar para que sean procesados posteriormente. Para ello se realizaron programas bajo Awk, con los que se generaron archivos ARFF que en si son del tipo de archivos con los que trabaja Weka por defecto.

Generados los archivos necesarios para crear el modelo (que en sí son vectores que contienen las características), se procedió a ingresarlos a Weka para así poder clasificarlos usando la técnica de regresión lineal (explicada anteriormente de forma teórica). En este proceso primero se ingresan los datos de entrenamiento (el cual se encuentra pre-etiquetado) para generar el modelo.

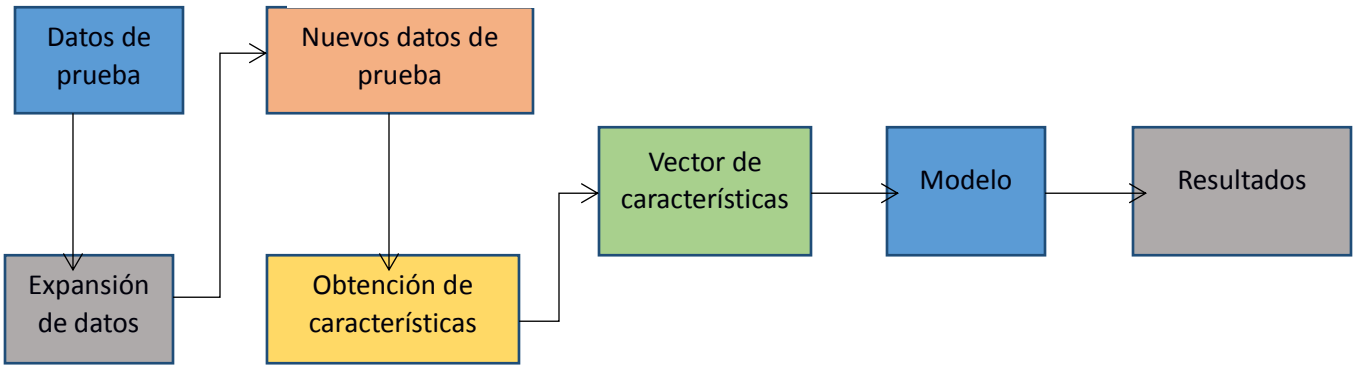
Figura 3.4 Proceso para la creación del modelo



Como se puede ver en la **Figura 3.4** para la creación del modelo primero se expande el conjunto de datos dados por el Semeval a modo de crear nuevos datos de entrenamiento, posteriormente se extraen las características de dichos datos para crear el vector de características, el cual es etiquetado para que finalmente se aplique la regresión lineal y se genere el modelo que sirve de entrenamiento.

Una vez que se obtuvo el modelo, posteriormente se repite un proceso similar al de la fase de entrenamiento, es decir, se expanden los datos de prueba para generan nuevos datos de prueba, de los cuales se extraen características y se genera un vector de las mismas, sin embargo estos no son etiquetados, puesto que se desea que sus valores sean predichos por el modelo. Esto se puede apreciar en la **Figura 3.5**.

Figura 3.5 Proceso de prueba del modelo



Una vez que se han aplicado han hecho las pruebas, se procede al análisis y comparación de los resultados obtenidos. Dicho análisis se hace en base al total de datos clasificados correctamente. En este trabajo se logra superar significativamente la clasificación respecto a los reportados por el SemEval 2014, esto se ve con más detalle en el capítulo siguiente.

Capítulo 4: Análisis de Resultados

En este capítulo se hace un análisis de los resultados arrojados usando el modelo de expansión de palabras y las características de similitud. Para ello se hace comparativa con los resultados mostrados en el Semeval 2014.

4.2. Información procesada

Durante la creación del modelo de similitud se dio como resultado un conjunto de datos que es importante mencionar ya que ayuda a entender la cantidad de información que fue usada para este trabajo. La descripción del conjunto de datos usados se encuentra en la **Tabla 4-1**

Tabla 4-1 Datos procesados

Información	Descripción
Corpus proporcionados por SemEval	Son dos corpus que originalmente brinda el SemEval, cada uno de ellos contiene 500 líneas compuestas de dos partes (oraciones y palabras) que deben de ser comparadas, además de una tercera parte que corresponde el etiquetado para la predicción.
Información extraída de 750000 sitios web para los tres corpus	Se expandieron 1500 palabras pertenecientes a los corpus brindados por el SemEval (500 del corpus de frase-palabra y 1000 del corpus de palabra-sentido). Por cada palabra se extrae información de la web de 500

	páginas por cada palabra, por lo que suponiendo que una página web no es visitada dos veces, se extrae información de 750000 sitios web diferentes.
Oraciones nuevas de diez palabras de longitud	Por cada una de las 1500 palabras a expandir se crearon oraciones de 10 palabras de longitud, por lo que se aumenta el número de palabras usadas para calcular la similitud de frase-palabra de 500 a 5000 y de 1000 a 10000 en el caso de palabra-sentido

4.3. Resultados de obtenidos de la clasificación

En la **Tabla 4-1** se muestran los diez mejores resultados brindados por los otros equipos en el SemEval 2014 y que se encuentran en su sitio web (http://alt.qcri.org/semeval2014/task3/index.php?id=results), así mismo se incluye el resultado obtenido por este trabajo.

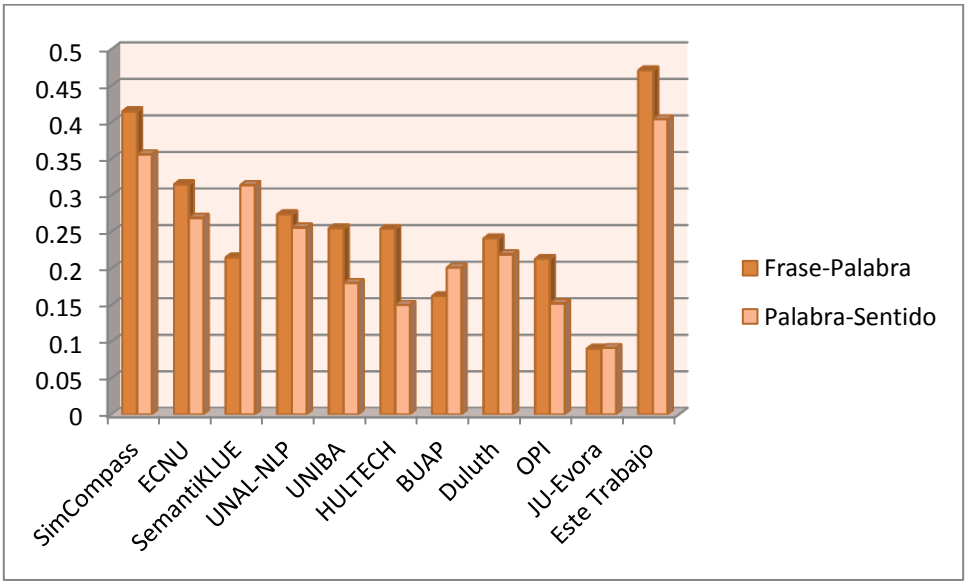
Tabla 4-2 Resultados Obtenidos

Equipo	Frase-Palabra	Palabra-Sentido
SimCompass	0.415	0.356
ECNU	0.315	0.269
SemantiKLUE	0.215	0.314
UNAL-NLP	0.274	0.256
UNIBA	0.255	0.18
HULTECH	0.254	0.15
BUAP	0.162	0.201
Duluth	0.241	0.219

OPI	0.213	0.152
JU-Evora	0.09	0.091
Este Trabajo	0.471	0.404

Como se puede apreciar los resultados obtenidos en este trabajo superan a los reportados por el SemEval 2014 como se ve en la **Figura 4.1**.

Figura 4.1 Comparación de los Resultados



Sin embargo a pesar de que los resultados arrojados por este trabajo son superiores, no se llega a obtener un indicador favorable, esto debido a que no se supera el 50% de clasificación correcta, por lo que queda claro que el método de regresión lineal no es del todo el indicado para esta clasificación.

Capítulo 5: Conclusiones y trabajo futuro

Con la elaboración de este trabajo y los resultados obtenidos, uno puede apreciar que se logró desarrollar un modelo que permite calcular la similitud entre textos de diferente tamaño de manera satisfactoria, además de que se superaron los indicadores reportados en el SemEval 20014, lo cual indica que el concepto de expandir el texto basándose en contenido web ayuda a mejorar el cálculo de la similitud basada en características.

Sin embargo a pesar de dicho logro, también se concluye que es necesario hacer mejoras al modelo, esto debido a que no se supera el 50% de clasificación correcta dada la técnica de clasificación (regresión lineal), por lo que en un futuro se pretende hacer pruebas usando máquinas de soporte vectorial o clasificación bayesiana.

De igual modo se pretende analizar con más a detalle si es necesario agregar o quitar algunas similitudes, puesto que en muchos casos cosas simples demuestran tener un mejor desempeño. Así mismo se espera que a futuro sea posible mejorar la propuesta de expansión de texto, esto debido a que, por ejemplo hubo algunos aspectos que no se consideraron tales como la ubicación de la palabra a expandir o la posible repetición de palabras en los lados de la palabra.

Finalmente, cabe mencionar que al desarrollar este trabajo se han adquirido más conocimientos acerca del procesamiento del lenguaje natural así como de la inteligencia artificial y se espera que posteriormente se logre mejorar y hacer uso real de este modelo, puesto que existen diferentes áreas en las que sería agradable aplicarlo.

Bibliografía.

1. Roth, M., Frank, A., (2013). *Automatically Identifying Implicit Arguments to Improve Argument Linking and Coherence Modeling*. Department of Computational Linguistics Heidelberg University, Alemania. Semeval 2013
2. Beltagy, A., et.al., (2013). *Montague Meets Markov: Deep Semantics with Probabilistic Logical Form*. The University of Texas at Austin. Semeval 2013
3. Shutova, E.,(2013). *Metaphor Identification as Interpretation. International Computer Science Institute and Institute for Cognitive and Brain Sciences*. University of California, Berkeley. Semeval 2013
4. Salehi, B., Cook,P., (2013). *Predicting the Compositionality of Multiword Expressions Using Translations in Multiple Languages.* Department of Computing and Information Systems, The University of Melbourne Victoria, Australia. Semeval 2013
5. Schwartz, H., et.al., (2013). *Choosing the RightWords: Characterizing and Reducing Error of the Word Count Approach.* University of Pennsylvania, Lymba Corporation. Semeval 2013
6. Horbach, A., (2013). *Using the text to evaluate short answers for reading comprehension exercises. Department of Computational Linguistics.* Saarland University, Saarbrücken, Alemania. Semeval 2013
7. Bird, S., et al., (2009). *Natural Language Processing whit Python.* USA: O'Reilly.
8. Di Pierro Massimo. (2011). *Web2Py Full Stack Web Framework* (4° ed.). Chicago, USA: Experts4Solutions
9. NLTK Project. (2015). *Natural Language Toolkit*. Recuperado el 4 de enero de 2015 de http://www.nltk.org/index.html
10. Princeron University. (2015). *WordNet A lexical database for English*. Recuperado el 4 de enero de 2015 de http://wordnet.princeton.edu
11. Crummy (2015). *Beautiful Soup.* Recuperado el 5 de enero de 2015 de http://www.crummy.com/software/BeautifulSoup/
12. Garcia Diego. (s.f.). *Manual de Weka.*

13. C.Leacok and M. Chodorow. (1998). *Combining local context and wordnet similarity for Word sense identification.* In Christiane Fellbaum, editor, MIT Press.
14. Lesk Michael. (1986). *Automatic sense desambiguation using machine readable dictionaries: How to tell a pine cone from an ice cream cone.* Proceedings of the 5th Annual International Conference on Systems Documentation.
15. Wu Z., Palmer M. (1994). *Verb semantics ans lexical selection.* James Pustejovsky.
16. Resnik, P. (1995). *Using Information content to evaluate semantic similarity in a taxonomy.* California, USA. Proceedings of the 14th International Joint Conference on Artifial Intelligence
17. Lin, D. (1998). *An information-theoretic definition of similarity.* California, USA. Proceeding of the Fifteenth International Conference on Machine Learning.
18. Jiang J., Conrath D. (1997). Semantic similarity based on corpus statics and lexical taxonomy. Proc of 10th International Conference on Research in Computacional Linguistics
19. Mihalcea R., et.al. (2006). *Corpus-based and knowledge-based measures of text semantic similarity.* Proceedings of the 21st National Conference on Artificial Intelligence
20. Turney, P. (2001). *Mining the web for synonyms: Pmi-ir versus lsa on toefl.* Proceedings of the 12th European Conference on Machine Learning.
21. Landauer T., et. al. (1998). *An Introduction to Latent Semantic Analysis*
22. aihorizon. (2015). *Machine Learning, Part 1: Types of learning problems.* Recuperado el 29 de marzo de 2015 de http://www.aihorizon.com/essays/generalai/machine_learning.htm
23. Colmenares, G. (s.f.). *Inteligencia Artificial.*
24. aihorizon. (2015). *Machine Learning, Part 2: Supervised and Unsupervised Learning.* Recuperado el 30 de marzo de 2015 de

http://www.aihorizon.com/essays/generalai/supervised_unsupervised_machine_learning.htm

25. Mendoza H, Vargas J, Lopez L, Bautista G. (2002). Métodos de Regresión. Universidad Nacional de Colombia, http://www.virtual.unal.edu.co/cursos/ciencias/2007315/. Licencia: Creative Commons BY-NC-ND.
26. Ñanculef, R. (2007). *Modelos Lineales para Clasificación. Parte b. Regresión Logística.* [Diapositivas de PowerPoint]. Recuperado de www.inf.utfsm.cl/~jnancu/tema2.beamer.pdf
27. Vidal Jesús A. (2002). *El lenguaje de programación AWK/GAWK. Una guía de usuario para AWK.* Madrid, España.
28. Hispanoteca. (2015). *Lemma und Lemmatiisierung.* Recuperado el 10 de enero de 2015 de http://www.hispanoteca.eu/Lexikon%20der%20Linguistik/l/LEMMA%20%20%20Lema.htm