UNIVERSIDAD TÉCNICA PARTICULAR DE LOJA

La Universidad Católica de Loja

ÁREA TÉCNICA

TÍTULO DE INGENIERO EN SISTEMAS INFORMÁTICOS Y COMPUTACIÓN

Esquema de clasificación de información universitaria basado en NERC: caso noticias UTPL

TRABAJO DE TITULACIÓN.

AUTORAS: Bermeo Cano, Mary Eugenia

Cueva Enriquez, María Alexandra

DIRECTORA: González Eras, Alexandra Cristina, Ing.

LOJA-ECUADOR

2015

APROBACIÓN DE LA DIRECTORA DEL TRABAJO DE TITULACIÓN

Ingeniera
Alexandra Cristina González Eras**.**
DIRECTORA DEL TRABAJO DE TITULACIÓN

De mi consideración:

El presente trabajo de titulación: Esquema de clasificación de información universitaria basado en NERC: caso noticias UTPL, realizado por Bermeo Cano Mary Eugenia y Cueva Enriquez María Alexandra, ha sido orientado y revisado durante su ejecución, por cuanto se aprueba la presentación del mismo.

Loja, mayo de 2015

f. ……
Ing. Alexandra Cristina González Eras
Cl.

DECLARACIÓN DE AUTORÍA Y CESIÓN DE DERECHOS

Bermeo Cano Mary Eugenia y Cueva Enriquez María Alexandra, declaramos ser autoras del presente trabajo de titulación: “Esquema de clasificación de información universitaria basado en NERC: caso noticias UTPL”, de la Titulación de Sistemas Informáticos y Computación, siendo la Ing. Alexandra Cristina González directora del presente trabajo; y eximimos expresamente a la Universidad Técnica Particular de Loja y a sus representantes legales de posible reclamos o acciones legales. Además certificamos que las ideas, conceptos, procedimientos y resultados vertidos en el presente trabajo investigativo, son de nuestra exclusiva responsabilidad.

Adicionalmente declaramos conocer y aceptar la disposición del Art. 88 del Estatuto Orgánico de la Universidad Técnica Particular de Loja que en su parte pertinente textualmente dice: "Forman parte del patrimonio de la Universidad la propiedad intelectual de investigaciones, trabajos científicos o técnicos y tesis de grado o trabajos de titulación que se realicen con el apoyo financiero, académico o institucional (operativo) de la Universidad".

f.....
Autora: Bermeo Cano Mary Eugenia
Cédula: 1104212145

f.....
Autora: Cueva Enriquez María Alexandra
Cédula: 1104047350

DEDICATORIA

Con infinito amor y gratitud dedico a mis padres, que con tanto sacrificio y con su ejemplo me supieron inculcar valores, el de valentía, honradez, trabajo, y decisión, a mis hermanas que me han apoyado e inspirado siempre, y a mi hija Josselin Chamba que es la razón de mi vida.

Mary Eugenia Bermeo Cano

Agradezco primeramente a Dios, a mi familia y a aquellas personas que aprecio y respeto, y que me ayudaron (directa o indirectamente) a concluir con esta importante etapa de mi vida, a todas ellas les doy gracias.

María Alexandra Cueva Enriquez

AGRADECIMIENTO

Muy sinceramente a la Universidad Técnica Particular de Loja, a la titulación de Ingeniería en Sistemas y Computación, a la Ing. Alexandra González quien ha sabido guiarnos en este proceso, ha mantenido paciencia y gracias a su sabiduría y experiencia hemos podido culminar con éxito nuestro trabajo de titulación.

A todos quienes nos han sabido apoyar de una u otra manera, amigos, compañeros de aula y familia, dios les pague por tanta bondad.

Mary y Ma. Alexandra

ÍNDICE DE CONTENIDOS

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

RESUMEN

La presente investigación propone el desarrollo de un esquema de clasificación de información universitaria. Para su desarrollo se aplican dos metodologías basadas en las técnicas de Reconocimiento y Clasificación de Entidades Nombradas (NERC), denominadas: Procesamiento de Lenguaje Natural (PLN) y Aprendizaje Automático (AA). NERC permiten aprovechar la riqueza del contexto en el cual se presentan las entidades nombradas, tales como: personas, organizaciones, locaciones, fechas, y títulos de persona. Como primera fase se tiene la creación de un corpus de doscientas noticias, etiquetado manualmente, el mismo que sirve para el análisis y creación de patrones. En segunda fase está la metodología de PLN, en donde se utiliza la herramienta GATE (General Architecture for Text Engineering), es una infraestructura open-source basada en Java, sirve para desarrollar y reutilizar componentes de software para resolver el problema de clasificación, y la última fase es la metodología de Aprendizaje Automático, en donde se aplica el algoritmo de clasificación SVM o Maquinas de Vectores de Soporte, para lo cual se presentan los resultados mediante un demo.

Palabras clave: NER, NERC, PLN, SVM, Aprendizaje Automático, GATE, LUCENE, JAPE y ANNIE.

ABSTRACT

This research proposes the development of a classification scheme of college information. Natural Language Processing (NLP) and Machine Learning (AA): two methodologies for development based on the recognition and classification techniques Named Entity (NERC), called apply. NERC allow harness the wealth of the context in which the named entities such as are presented: people, organizations, locations, dates, and titles of person. As first phase of the creation of a corpus of two hundred news, labeling manually, the same as used for the analysis and pattern making. In second phase is PLN methodology, where the GATE (General Architecture for Text Engineering) tool is used, is an open-source Java-based infrastructure serves to develop and reuse software components to solve the problem of classification, and The last phase is the methodology of machine learning, where the SVM algorithm and Support Vector Machines classification, for which the results are presented through a demo applies.

Keywords: NER, NERC, PLN, SVM, Machine Learning, GATE, LUCENE, JAPE, ANNIE.

INTRODUCCIÓN

La gran cantidad de noticias que se encuentra en la Web en formato digital, hacen complicado el análisis del contexto. Las técnicas NERC son una opción para solucionar este problema y facilitan la clasificación del enorme espacio de medios textuales producidos cada día por las organizaciones, gobiernos e individuos, un ejemplo son las noticias universitarias.

Las técnicas de Reconocimiento y Clasificación de Entidades Nombradas (NERC), permiten a través del reconocimiento de entidades nombradas clasificar documentos. La implementación de estos sistemas ha cambiado la forma de indexar la información, haciéndola accesible. A través de la coincidencia de palabras en los textos y los esquemas manuales de clasificación de las instituciones, ofrecen la posibilidad de relacionar las entidades como nombres de personas, organizaciones y palabras claves asociados dentro del contexto de la noticia.

El Reconocimiento y Clasificación de Entidades Nombradas (NERC) constituye la tarea base de todo sistema de extracción de información, a la vez que juega un papel muy importante en la clasificación automática de noticias, la categorización de textos, la construcción automática de resúmenes, etc. El NERC puede definirse como la tarea de identificar y clasificar en un documento textual expresiones que identifican instancias de conceptos relevantes para algún dominio de aplicación.

Para manipular de forma adecuada las fuentes de información textual, es necesaria la aplicación de un conjunto de técnicas para el Procesamiento del Lenguaje Natural. Éstas incluyen un conjunto de tareas que van desde las más básicas, como la segmentación de los textos por palabras y el análisis morfosintáctico, hasta las de más alto nivel, como la extracción automática, la clasificación automática de noticias y el desarrollo de interfaces de usuario.

Las noticias universitarias generadas en la UTPL, se encuentran por fecha de edición, actualmente no permite realizar una búsqueda adecuada. Para solucionar este inconveniente se propone un esquema de clasificación de la información por categorías para lo cual se trabaja con dos soluciones, la primera va orientada a la metodología PLN en donde el proceso en su mayoría es manual y la segunda va orientada a Aprendizaje Automático en donde la mayoría del proceso es automático.

El objetivo en este trabajo es investigar la técnica NERC, basada en Procesamiento de Lenguaje Natural (PLN) y Aprendizaje Automático (AA); para luego proponer y desarrollar un esquema de clasificación para las noticias de la UTPL, que contemplen cuatro tipos de

categorías definidas como: académica, investigativa, socio/cultural y emprendimiento. Y finalmente realizar la validación y evaluación del esquema propuesto.

En esta investigación se presenta un estado del arte de las técnicas NERC, además de conocer en teoría lo que es Procesamiento de Lenguaje Natural y Aprendizaje Automático, seguidamente se plantea el esquema de clasificación propuesto, donde se explica las tres fases que son: etiquetación manual del corpus de entrenamiento, el proceso de la metodología PLN y el proceso para la metodología de Aprendizaje Automático. Finalizando con el análisis de resultados, conclusiones y recomendaciones.

PERSPECTIVA GENERAL

Objetivo General

- Determinar un esquema de clasificación de las noticias publicadas por la Universidad Técnica Particular de Loja, utilizando las técnicas de Clasificación y Reconocimiento de Nombres de Entidades NERC.

Objetivos Específicos

- Investigar las técnicas NERC, basadas en Aprendizaje Automático y Procesamiento de Lenguaje Natural.
- Desarrollar un esquema de clasificación para las noticias de la UTPL que contemple categorías.
- Realizar la validación, evaluación, y comparación de la clasificación de noticias mediante las dos técnicas PLN y AA.

HIPÓTESIS

- ¿Las técnicas NERC pueden ser aplicadas para la clasificación de documentos no estructurados en el contexto universitario?

GLOSARIO

AA: (Aprendizaje Automático), por lo general se lo conoce por su terminología en inglés Machine Learning, es desarrollar técnicas que permitan a las computadoras aprender, es decir, se considera como un proceso de inducción del conocimiento.

ANNIE: (Nearly – New Information Extractions System), es un componente de GATE que contiene recursos y herramientas orientadas a la Extracción de Información.

API: (Application Programming Interface), es el conjunto de subrutinas, funciones y procedimientos (o métodos, en la programación orientada a objetos) que ofrece cierta biblioteca para ser utilizado por otro software como una capa de abstracción. Son usadas generalmente en las bibliotecas.

Corpus: es un conjunto amplio y estructurado de ejemplos reales de uso de la lengua. Estos ejemplos pueden ser textos (lo más común) o muestras orales (generalmente transcritas). Un corpus lingüístico es un conjunto de textos relativamente grande, creado independientemente de sus posibles fines de uso.

Dataset: es una representación de datos residente en memoria que proporciona una modelo de programación relacional coherente independientemente del origen de datos que contiene.

Demo: es un prototipo de un determinado programa informático con el fin de mostrar la idea de funcionamiento y demostrar sus funcionalidades. Se utilizan para que los usuarios o potenciales clientes puedan probar el software antes de comenzar a utilizarlo en un ambiente real.

Función de Kernel: Son funciones matemáticas que se emplean en las Máquinas de Soporte Vectorial. Estas funciones son las que le permiten convertir lo que sería un problema de clasificación no lineal en el espacio dimensional original, a un sencillo problema de clasificación lineal en un espacio dimensional mayor.

GATE: (General Architecture for Text Engineering), es una infraestructura open-source basada en Java que sirve para desarrollar y reutilizar componentes de software para resolver diversos problemas en el dominio del Procesamiento de Lenguaje Natural.

Gazzeter: Se denomina un gazzetter a una lista de palabras pertenecientes a un tipo de categoría, que sirven posteriormente para crear las reglas (JAPE).

GNU: (General Public License o Licencia Pública General), es la licencia más ampliamente usada en el mundo del software y garantiza a los usuarios finales (personas, organizaciones, compañías) la libertad de usar, estudiar, compartir (copiar) y modificar el software.

JAPE: (Java Annotation Patterns Engine), permite reconocer anotaciones utilizando gramáticas que están constituidas por patrones y reglas.

LibSVM: es un archivo JAR, y tiene que estar en la ruta de clase construir el clasificador SVM. Esta permite que el usuario pueda realizar estadísticas de la obtención de las clases clasificadas.

Lucene: Es una herramienta o motor de búsqueda de alto rendimiento escrito en Java y es Open Source. Es escalable y logra su alto rendimiento mediante el uso de índices.

MUC: (Message Understanding Conference), las conferencias MUC ayudan a fomentar el desarrollo de nuevos y mejores métodos de extracción de información, mediante sus estándares propuestos.

NE: (Named Entities), es una palabra que no varía en el transcurso del tiempo como por ejemplo: nombres de las personas, las organizaciones, las locaciones, las expresiones de los tiempos, cantidades, valores monetarios, porcentajes, etc.

NER: (Named Entity Recognition), es una tarea de lingüística computacional que busca reconocer cada palabra de un documento para saber si es o no un nombre de entidad.

NERC: (Named Entity Recognition and Classification), es una tarea lingüística computacional en la cual se busca clasificar cada palabra de un documento en una determinada categoría de una lista de categorías definida, como por ejemplo pueden ser las siguientes categorías: persona, lugar, organización o fecha

Open Source: es una expresión que se utiliza cuando un programa informático permiten el acceso a su código de programación, lo que facilita modificaciones por parte de otros programadores ajenos a los creadores originales del software en cuestión.

PLN: (Procesamiento de Lenguaje Natural), es el uso de computadoras para entender lenguajes (naturales) humanos tales como inglés, francés o japonés

Precisión: se define como la proporción de material recuperado realmente relevante, del total de los documentos recuperados.

Recall: busca la proporción del material relevante recuperado del total de los documentos que son relevantes en la muestra analizada sin importar si estos han sido recuperados por el sistema.

SVM: (Support Vector Machine o máquinas de vectores de soporte), es un sistema de aprendizaje basado en el uso de un espacio de hipótesis de funciones lineales en un espacio de mayor dimensión inducido por un Kernel.

TreeTagger: es un lenguaje independiente part-of-speech, que es compatible con varios idiomas diferentes a través de los archivos de parámetros, incluyendo Inglés, francés, alemán, español, italiano y búlgaro.

WEKA: (Waikato Environment for Knowledge Analysis), en español entorno para análisis del conocimiento de la Universidad de Waikato, es una plataforma de software para el aprendizaje automático y la minería de datos escrito en Java y desarrollado en la Universidad de Waikato. Weka es software libre distribuido bajo la licencia GNU-GPL.

CAPÍTULO I

MARCO REFERENCIAL

En este capítulo se realiza un estudio de las técnicas de Reconocimiento y Clasificación de Entidades Nombradas (NERC), partiendo de la definición de Extracción de Información, en donde se expone la arquitectura general de un sistema de Extracción de Información (EI), Hobbs (1993), presenta la diferencia entre NER y NERC (Tabla 1). Se define el tema de Procesamiento de Lenguaje Natural (PLN), donde se presenta los niveles de análisis y los problemas que existen relacionados con NERC. También se define Aprendizaje Automático (AA); definiendo los cinco tipos que existen; se mencionan también algunos ejemplos de algoritmos, centrándose en el Algoritmo SVM, que es aplicado en la investigación.

1.1. Extracción de Información.

Pazienza (1997), menciona que las tecnologías de Extracción de Información, son una disciplina que forma parte del Procesamiento de Lenguaje Natural. La EI tiene como objetivo extraer aquellos hechos relevantes que se encuentran de forma explícita en los documentos, es un proceso posterior a la Recuperación de la Información (RI).

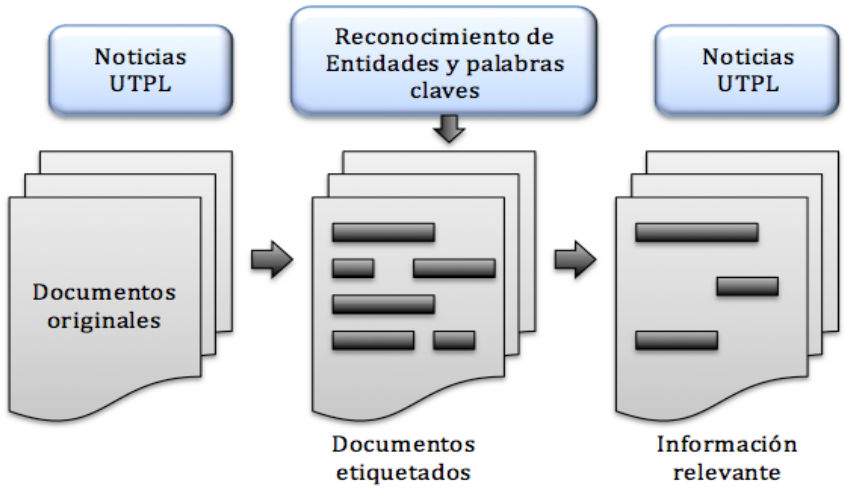

Figura 1. Ejemplo de extracción de información.
Fuente: Las autoras.

Con el objetivo de medir los avances de la EI, se han promovido en Estados Unidos varias conferencias entre ellas Message Understanding Conference (MUC)[1], TIPSTER[2] (programa de investigación sobre recuperación y extracción de información del gobierno de EE.UU) y Automatic Content Extraction (ACE)[3] cuya función principal es promover el desarrollo de nuevos y mejores métodos de EI, celebradas año tras año y se trata un tópico especifico, investigadores ponen a prueba sus métodos sobre un corpus etiquetado. En las conferencias MUC, se han definido cinco tareas que se ejecutan en la EI; así:

[1] http://www.itl.nist.gov/iaui/894.02/related_projects/muc/proceedings/muc_7_toc.html
[2] http://www-nlpir.nist.gov/related_projects/tipster/
[3] http://www.itl.nist.gov/iad/mig//tests/ace/

- **Reconocimiento de Entidades Nombradas (NE):** Se trata de reconocer nombres personas, organizaciones, lugares, fechas, tiempo, moneda etc.
- **Plantilla de Elementos (TE):** Tarea que añade información descriptiva a los resultados de NE, utiliza CO.
- **Tareas de Correferencia (CO):** Identifica expresiones de texto que hagan referencia a un mismo objeto, este proceso incluye las tareas de NE y TE.
- **Plantilla de Relación (TR):** Descubre las relaciones entre entidades, trabaja con número de partida de posibles relaciones entre los distintos elementos.
- **Plantilla de Escenario (ST):** Construye un escenario de eventos específicos donde los resultados de TE y TR coinciden y obtiene información relacionada a estos elementos.

Las conferencias MUC son ampliamente reconocidas dentro de EI, especialmente durante el lapso de 1990 y 1998 donde la competición entre diferentes grupos de investigación aportaron con metodologías, evaluación, diseños en definitiva experiencias que más tarde Hobbs (1993) las reunió y propuso una arquitectura general para EI, que a decir de su autor es “una cascada de módulos que en cada paso agregan estructura al documento, y algunas veces, filtran información relevante por medio de aplicar reglas” la cual consta de 8 procesos, secuenciales que se detallan en la Figura 2.

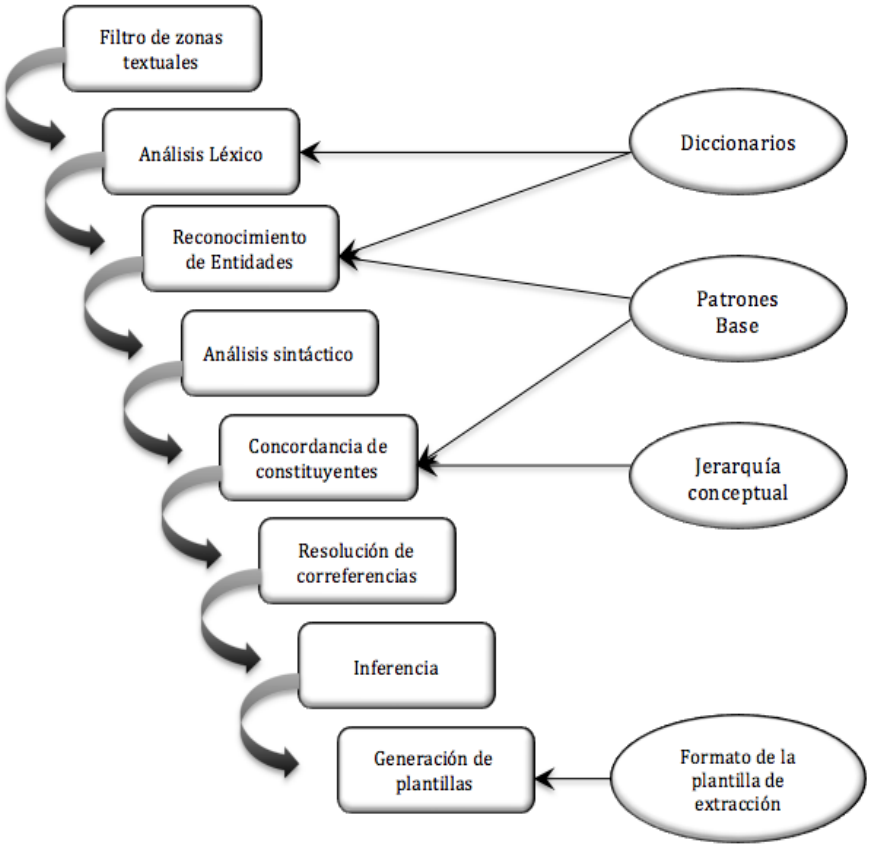

Figura 2. Arquitectura General de un sistema de EI.
Fuente: Hobbs (1993).

A partir del esquema de Hobbs (1993), se han propuesto varias metodologías alternativas, en donde distinguen 4 módulos que varían del idioma en que estén trabajando, Llidó (2003) y Torres (2012), se destacan las siguientes: Segmentador, Procesamiento morfológico y léxico, Análisis sintáctico y Análisis de dominio.

1.2. Reconocimiento y Clasificación de Entidades Nombradas.

Para tener una mejor compresión de lo que es el Reconocimiento y Clasificación de Entidades Nombradas (NERC), debemos iniciar dando una breve explicación de su antecesor (NER), y luego una definición de NERC.

1.2.1. Reconocimiento de Entidades Nombradas (NER).

Sánchez (2009), menciona que el Reconocimiento de Entidades Nombradas (NER), es una tarea de lingüística computacional que busca reconocer cada palabra de un documento para saber si es o no un nombre de entidad.

1.2.2. Reconocimiento y Clasificación de Entidades Nombradas (NERC).

Sánchez (2009), indica que el Reconocimiento y Clasificación de Entidades Nombradas es una tarea de lingüística computacional que busca reconocer y clasificar cada palabra de un documento en una de las categorías siguientes: persona, lugar, organización, locación, titulo de persona, fecha etc.

A continuación se presenta un ejemplo con la finalidad de demostrar la diferencia entre NER y NERC, se parte con la frase “Luis Ramón visitó la ciudad de Quito”, en donde primeramente mediante la técnica NER, se identifica las palabras que pertenezcas a una EN, en este caso se encontró: Luis, Ramón y Quito, las otras palabras de la frase no son una Entidad Nombrada. El segundo paso es la clasificación de estas EN encontradas en la frase; para lo cual mediante NERC, se las clasifica Luis y Ramón como PER (Persona) y Quito como LUG (Lugar).

Tabla 1. Diferencia entre NER y NERC.

Tema	Frase	Detalle
NER	Luis Ramón visitó la ciudad de Quito	Previamente se define el siguiente esquema: ESX[4] donde cada letra está categorizada de la siguiente forma: **E:** Indica que la palabra es el inicio de la palabra de una Entidad Nombrada[5] (EN). **S:** Indica su pertenencia a la entidad nombrada, pero que no se encuentra al inicio de esta. **X:** Indica que la palabra no forma parte de una entidad nombrada. Aplicando a la frase antes mencionada tenemos lo siguiente: Luis Ramón_S visitó_X la_X ciudad_X de_X Quito_E
NERC	Luis Ramón visitó la ciudad de Quito	Una vez reconocido en el proceso NER se procede a la clasificación, las categorías que se tendrán son: PERSONA, ORGANIZACIÓN, LUGAR, donde abreviado quedará PER, ORG, LUG respectivamente; entonces aplicando a la frase da el siguiente resultado: Luis _E_PER Ramón_S_PER visió _ X _ X la_X_X ciudad_X_X de_ X _ X Quito_S_LUG

Fuente: Sánchez, C. (2008). Clasificación de Entidades Nombradas. Puebla: INAOE.

1.2.3. Tipos de problemas en una entidad nombrada.

Existen cuatro tipos de problemas que se presentan al momento de reconocer y clasificar una Entidad Nombrada, a continuación se detallan en Tabla 2:

Tabla 2. Tipos de problemas en una entidad nombrada.

Ambigüedad	**Ambigüedad de nombres de entidades**	Se refiere a cuando aún escritas las palabras de la misma forma, deben ser clasificadas como NE de diferente clase. Por ejemplo la palabra Loja se la puede clasificar dentro del grupo de Persona o del grupo Locación.
	Ambigüedad con palabras comunes	Se refiere a que pueden existir palabras que a la vez puede ser un NE o una palabra común, por ejemplo Puertas, donde lo clasificaría como Persona (Apellido) y puede también clasificarlo como una puerta (de casa).

[4] *Siglas que se designa a una palabra para identificar si es o no una Entidad Nombrada.*
[5] *Es un una palabra o secuencias de palabras que se identifican como nombre de persona, organización, lugar, fecha, tiempo, porcentaje o cantidad.*

Correferencia	**Correferencia con Foco**	Es aplicada a textos cuyo foco se centra en una única persona, organización o locación. Esta resolución es más aplicable para el aprendizaje automático.
	Correferencia Completa	A diferencia de la de la correferencia con foco, esta establece relaciones en de entidades en todo el texto obteniendo u mayor número de entidades y corre el riesgo de aumentar la ambigüedad. (Garcia y Gamayo, 2011)
Variación de Nombre de Entidades	Este problema surge cuando un Named Entity tiene diferentes formas de llamarse, pero a la final apunta al mismo nombre, por ejemplo tenemos el nombre "Luis Ramón", que también puede aparecer en el contexto como "Ramón Luis" o "Ramón N"; como se observa es la misma persona	
Desambiguación del Sentido de las Palabras	La ambigüedad en el proceso lingüístico se presenta cuando pueden admitirse distintas interpretaciones a partir de la representación o cuando existe confusión al tener diversas estructuras y no tener los elementos necesarios para eliminar las incorrectas (Gelbukh y Galicia, 2007), es decir, cuando en el lenguaje de destino se le pueden asignar dos o más expresiones distintas.	

Fuente**:** Sánchez, C. (2008). Clasificación de Entidades Nombradas. Puebla: INAOE.

Como se puede observar en el cuadro existen cuatro tipos de problemas, que surgen al momento de darse el Reconocimiento y Clasificación de una Entidad Nombrada.

1.3. Metodologías basadas en las técnicas NERC.

El Procesamiento de Lenguaje Natural abreviado PLN y el Aprendizaje Automático que posteriormente se lo denominará AA, son dos metodologías diferente basadas en NERC pero que pertenecen a la inteligencia artificial y lingüística.

1.3.1. Procesamiento de Lenguaje Natural (PLN).

Covington, (2002), el Procesamiento de Lenguaje Natural, es el uso de computadoras para entender lenguajes (naturales) humanos tales como inglés, francés o japonés. Por entender no se quiere decir que el computador tenga pensamientos, sentimientos y conocimientos humanizados, sino que el computador pueda reconocer y usar información expresada en lenguaje humano.

Manaris y Slator (1996), definen a un sistema de PLN como aquel que encapsula un modelo del lenguaje natural en algoritmos apropiados y eficientes. En donde las técnicas de

modelado están ampliamente relacionadas con eventos en muchos otros campos, incluyendo:

- Ciencia de la computación, la cual provee métodos para representar modelos, diseñar e implementar algoritmos para herramientas de software.
- Lingüística, la cual contribuye con nuevos modelos lingüísticos y procesos.
- Matemática, la cual identifica modelos formales y métodos.
- Neurociencia, la cual explora los mecanismos mentales y otro tipo de actividades físicas.

Entre estos campos, la lingüística ha aportado el conocimiento lingüístico de las lenguas naturales. Este conocimiento dentro de un sistema de PLN puede ser dividido en niveles definidos en términos de la característica declarativa (qué) y procedural (cómo) (Manaris y Slator, 1996).

1.3.1.1. Niveles de análisis de PLN.

Covington, (1994), indica que el Lenguaje Natural se divide en niveles de análisis, los cuales a continuación se describe cada uno.

1. Nivel morfológico

Hernández,. (2007), aclara que el análisis morfológico se refiere a la revisión y detección de la correcta estructura de cada palabra de la instrucción. Además determina la forma, clase o categoría gramatical de cada palabra de una oración.

2. Nivel sintáctico

Sosa y Hernández, (1997), el nivel sintáctico corresponde a la relación que existe entre los diferentes símbolos y signos del lenguaje; así como también la forma en que las palabras se relacionan entre sí. En este nivel se analizan todas las combinaciones gramaticales referentes al uso del artículo y del sustantivo.

3. Nivel semántico

Vallez y Pedraza, (2007), el nivel semántico es aquel que estudia el procesamiento y detección del significativo que cada elemento tiene por sí mismo dentro de la oración. En este nivel puede producirse ambigüedad debido a que una palabra puede tener uno o varios significados; por ejemplo la palabra banco puede ser considerada como asiento o como entidad bancaria.

En este nivel también se debe tomar en cuenta la variación léxica, la que se refiere a la posibilidad de utilizar distintos términos que muestran un mismo significado, denominado *sinonimia;* por ejemplo la palabra coche, vehículo y automóvil pertenecen a un igual significado.

4. Nivel pragmático

Contreras (2011), el nivel pragmático se interpreta la estructura de la oración para determinar el verdadero significado dentro del contexto específico. En algunas ocasiones el significado de una palabra debe de tomárselo en sentido figurado; por ejemplo en la frase "Él se moría de risa", existe un problema de interpretación de significado.

Para una mejor comprensión de los niveles de PLN, a continuación se muestra las principales características de cada uno de sus niveles, en donde se muestra el tipo de acción y como se da la acción. Ver Tabla 3.

Tabla 3. Niveles de conocimiento en el PLN.

Nivel	**Características**	
	Qué acción	**Cómo es la acción**
Morfológico (Léxico)	Unidades de palabras, Palabras	Formar palabras, Derivar unidades de significado
Sintáctico	Roles estructurales de palabras	Formar oraciones
Semántico	Significado independiente del contexto	Derivar Significado de oraciones
Pragmático	Significado dependiente del contexto	Derivar significado de oraciones relativo al discurso próximo

Fuente: Muñoz, V. (2008). Herramientas para la Extracción de Información. México.

1.3.1.2. Fases del proceso de descubrimiento de conocimiento.

En proceso de extracción de información tiene como objetivo, el descubrimiento de conocimiento. Para lo cual se debe ejecutar las siguientes fases que se describen a continuación. Ver Figura 3.

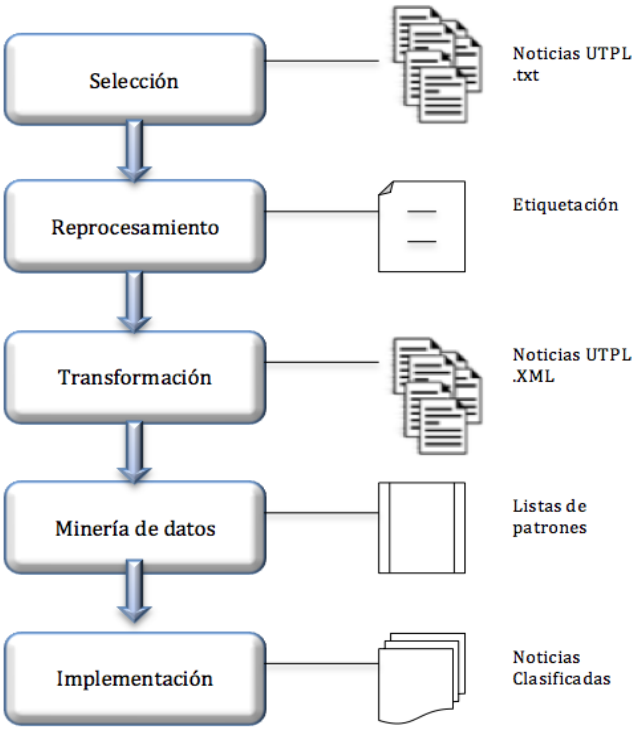

Figura 3. Proceso de Extracción de Información.
Fuente: Muñoz, V. 2008

Como se puede observar en la Figura 3, las fases para realizar descubrimiento de conocimiento son: selección, pre-procesamiento, transformación, minería de datos e implementación. Todas estas fases tienen como objetivo descubrir nuevo conocimiento.

1.3.1.3. Problemas relacionados con sistemas PLN.

Manaris y Slator, (1996), los sistemas de PLN deben atacar una variedad de problemas relacionados, entre ellos se mencionan a continuación:

- ***Inexactitud***, incluyendo errores ortográficos, signos de puntuación incorrectos, palabras transpuestas, y oraciones gramaticales. Un ejemplo puede ser una noticia con faltas de ortografía.

Figura 4. Noticia con falta de ortografía
Fuente: sitio Web de la UTPL (noticias)

- ***Incompletitud***, incluyendo construcciones elípticas, anáforas, etc. Por ejemplo una noticia en donde no exista ni una entidad nombrada; es decir no hay nombres de persona, ni organizaciones etc.

- ***Imprecisión***, incluyendo el uso de términos relativos sin un punto específico de referencia y el uso de términos cualitativos. Un ejemplo puede ser una noticia con párrafos sin puntos.

- ***Ambigüedad,*** debido a que pueden surgir múltiples interpretaciones en cualquier nivel del conocimiento lingüístico la ambigüedad puede ser resuelta usando el conocimiento de un nivel más alto, en la Figura 5, se muestra un ejemplo.

Figura 5. Ambigüedad.
Fuente: Muñoz, V. (2008). Herramientas para la Extracción de Información. México.

1.4. Herramientas.

Existen varias herramientas para el proceso de PLN, se describen en la siguiente Tabla.4:

Tabla.4: Descripción de Herramientas para PLN.

Herramientas	Características generales	Atributos de extracción
Gate (General Architecture for Text Engineering).[6]	Realiza análisis sintáctico, morfológico, etiquetado; además posee herramientas de recuperación de información, extracción de información para diferentes idiomas, etc.	Palabras etiquetadas según su naturaleza, ordenadas según su aparición en la frase.
Apache OpenNLP [7]	Realiza análisis sintáctico y morfológico, en donde realiza: segmentación de la oración, etiquetado de texto, extracción del nombre de las entidades, fragmentación, análisis y resolución de correferencia	Cualquier palabra que se encuentre dentro del idioma en el cual se encuentre configuradas las etiquetas de comparación

[6] GATE. General Architecture for Text Engineering. Disponible en: https://gate.ac.uk/
[7] Apache OpenNLP. Disponible: https://opennlp.apache.org/

LingPipe [8]	Permite realizar la tokenización de los textos y realizar la extracción de oraciones. Ayuda a verificar la ortografía de las consultas realizadas.	Permite realizar la búsqueda de nombres de personas, organizaciones y ubicaciones.
Lucene[9]	Es un motor de búsqueda de alto rendimiento escrito en Java y es Open Source. Es escalable y logra su alto rendimiento mediante el uso de índices. Tiene algoritmos de búsqueda eficientes. Estos algoritmos permiten rankear los resultados.	Permite realizar consultas de frases; consultas comodín; consultas de proximidad; consultas de rango y más; búsquedas por campos; ordenar por cualquier campo; y búsquedas a múltiples índices uniéndolos.

Fuente: Gate (General Architecture for Text Engineering) [6], Apache OpenNLP [7], LingPipe [8], Lucene[9].
Elaboración: Las Autoras

Como se puede observar existen varias herramientas que permiten realizar el Procesamiento de Lenguaje Natural, se describen las que se van utilizar en esta investigación con sus respectivas características.

1.4.1. Aprendizaje automático (AA).

Mitchell (1997), define al Aprendizaje Automático como una rama de la Inteligencia Artificial, se trata de la construcción y el estudio de los sistemas que se pueden aprender a partir de los datos.

Mitchell (1997), señala que AA es un programa de ordenador, APRENDE a partir de una experiencia **E** a realizar una tarea **T** (de acuerdo con una medida de rendimiento **P**), si su rendimiento al realizar **T**, medido con **P**, mejora gracias a la experiencia **E**.

Di Decod (2012), el objetivo del Aprendizaje Automático es desarrollar técnicas que permitan a las computadoras aprender, es decir, se considera como un proceso de inducción del conocimiento. El aprendizaje automático se centra en el estudio de la complejidad computacional de los problemas.

Kotsiantis (2007), el Aprendizaje Automático se puede aplicar en: motores de búsqueda, diagnósticos médicos, detección de fraude en el uso de tarjetas de crédito, análisis del mercado de valores, clasificación de secuencias de ADN (Ácido Desoxirribonucleico), reconocimiento del habla y del lenguaje escrito, juegos y robótica etc.

[8] LingPipe: Disponible en: http://alias-i.com/lingpipe/
[99] Lucene. Disponible en: http://lucene.apache.org/core/

1.4.1.1. Tipos de aprendizaje automático.

Ceccaroni (2009), un sistema es realmente inteligente si es capaz de observar su entorno y aprender de él. El Aprendizaje Automático busca aumentar las capacidades de los programas de inteligencia artificial; su límite está en el conocimiento que se les ha introducido; no resuelven problemas más allá de esos límites. Dentro de los tipos de aprendizaje se mencionan a continuación:

Aprendizaje inductivo: Se crea modelos de conceptos a partir de generalizar ejemplos simples. Se busca patrones comunes que expliquen los ejemplos. Su objetivo es descubrir leyes generales o conceptos a partir de un número limitado de ejemplos (búsqueda de patrones comunes). Su funcionamiento reside en la observación de similaridades entre ejemplos. Sus métodos se basan en el razonamiento inductivo.

Aprendizaje analítico o deductivo: Aplica la deducción para obtener descripciones generales a partir de un ejemplo de concepto y su explicación.

Aprendizaje genético: Aplica algoritmos inspirados en la teoría de la evolución para encontrar descripciones generales a conjuntos de ejemplos.

Aprendizaje conexionista: Busca descripciones generales mediante el uso de la capacidad de adaptación de redes de neuronas artificiales.

Existen tipos de aprendizaje automático según el algoritmo:

Tabla 5. Tipos de Aprendizaje de un algoritmo.

Tipo de aprendizaje	Definición
Aprendizaje supervisado	Produce una función que establece una correspondencia entre las entradas y las salidas deseadas del sistema.
Aprendizaje no supervisado	Todo el proceso se lleva a cabo sobre un conjunto de ejemplos formado por entradas al sistema. No existe información de las categorías de esos ejemplos.
Aprendizaje por refuerzo	El algoritmo aprende observando el mundo que le rodea. Su información de entrada es la retroalimentación que obtiene del exterior en función de sus acciones.
Transducción	Similar al aprendizaje supervisado, pero no construye de forma explícita una función. Trata de predecir las categorías de los futuros ejemplos basándose en los ejemplos de entrada, sus respectivas categorías y ejemplos nuevos.
Aprendizaje multi-tarea	Métodos de aprendizaje que usan conocimiento previamente aprendido por el sistema con el fin de enfrentarse a problemas similares a los vistos.

Fuente**:** Dovale, M. (2009). Inteligencia Artificial y Computacional.

1.4.1.2. Etapas para el diseño de sistemas de aprendizaje automático.

Zafra (2009), el desarrollo de un sistema de aprendizaje tiene que pasar por una serie de etapas que se deben especificar con cuidado si se quiere que el sistema funcione de manera correcta. La clave de los sistemas de aprendizaje es que a partir de una experiencia limitada, el sistema sea capaz de aprender de ella y realizar la misma tarea de forma más eficiente o ser capaz de realizar nuevas tareas. Todo sistema deberá basarse en los siguientes cinco pasos.

1. Elegir la experiencia que usaremos para entrenar el sistema de aprendizaje.

Se trata de un factor clave en el proceso de aprendizaje inductivo, ya que a partir de dicha información se va a realizar el proceso de generalización.

Si la información de partida no es consistente, está incompleta o contiene ruido, será muy normal que no obtengamos unos resultados aceptables.

De este modo, habrá que garantizar que la información sea representativa del problema, siendo conscientes de que normalmente los datos que se considera en el entrenamiento pueden tener un número excesivo de atributos (alta dimencionalidad), ruido en las clases de los ejemplos (ejemplos mal etiquetados), o en los atributos (atributos con ruido en sus valores), atributos irrelevantes, redundantes o con valores perdidos para determinados ejemplos.

2. Definir el tipo de función objetivo que pretendemos aprender.

Se debe dejar bien definido qué se quiere aprender exactamente, se debe aprender solamente lo necesario, cuanto más pretenciosos y ambiguo sea el objetivo, más complicado será el proceso.

La función objetivo que se quiere aprender se diseña para alcanzar la finalidad especificada y el proceso de aprendizaje aproximará este objetivo mediante una función del mismo tipo que denominaremos hipótesis. Existen diferentes tipos de funciones dependiendo de la clase de problema que se esté tratando. Clasificación, regresión, agrupación o cualquier otro problema.

3. Determinar la representación que utilizaremos para la función objetivo.

Al fijar la representación, se está definiendo un espacio de búsqueda para la hipótesis considerada con la que se pretende aproximar la función objetivo. A este espacio se le

denomina espacio de hipótesis. Es importante que el espacio de hipótesis contenga a la función objetivo. Si no así, muestra hipótesis está condenada a equivocarse en algunos ejemplos.

Cuanto más complejo sea el espacio de hipótesis más difícil será aprender la hipótesis y más ejemplos de entrenamiento serán necesarios, pero más capacidad tendrá para aproximar la función objetivo. En muchas ocasiones, cuando no se tiene muy claro qué tipo de representación es más adecuada, se prueban varias y se evalúa la calidad de las soluciones alcanzadas.

4. Seleccionar el algoritmo y aplicarlo para aprender la función objetivo.

Para cada tipo de función objetivo, espacio de hipótesis o tipo de conocimiento se tiene distintos algoritmos que pueden aplicarse. Es posible utilizar varios algoritmos distintos o diferentes implementaciones y comparar posteriormente su rendimiento.

El funcionamiento normal suele seleccionar el algoritmo o la implementación en función de la plataforma o de otras herramientas con las que deba utilizarse. Es habitual ejecutar el algoritmo de aprendizaje automático con distintos parámetros para evaluar y comparar su rendimiento.

5. Evaluar el sistema.

Es necesario evaluar el sistema para determinar si funciona bien en la práctica y generaliza en situaciones no vistas durante el entrenamiento. Un sistema que solamente funcione con ejemplos utilizados como experiencia en la fase d entrenamiento dl proceso de aprendizaje y que no sea capaz de abordar nuevos ejemplos para enfrentarse a nuevas situaciones, no es un sistema válido. La evaluación se puede realizar:

- Durante el aprendizaje, para seleccionar una hipótesis adecuada.
- Al final del aprendizaje, para indicar el nivel de confianza de la hipótesis elegida.

1.4.1.3. Ejemplos de algoritmos para aprendizaje automático.

A continuación se exponen cuatro algoritmos de clasificación con aprendizaje automático, donde su objetivo principal es la clasificación. Ver Tabla 6.

Tabla 6. Descripción de algoritmos de aprendizaje automático.

Algoritmo	Función	Ventajas	Desventajas
Árboles de decisión.	Realizan sus acciones en función de una o varias variables para la determinación de una acción correcta. Utiliza nodos como ramas, realiza un test para tomar la mejor decisión.	Permite comparar los posibles resultados de una decisión. Toma la mejor opción sobre la base de datos existente.	Que cuando se tiene una gran cantidad de variables suele tener dificultades para elegir la mejor decisión. (Hernández, P., 2004)
Naive Bayes.	Tiene sus fundamentos en el Teorema de Bayes, es uno de los algoritmos de aprendizaje práctico más utilizados por su sencillez.	Requiere una pequeña cantidad de datos de entrenamiento para la clasificación.	No es apto para el manejo de variables aleatorias continuas. (Lewis, D., 1998)
Máquinas de vectores de soporte (SMV).	Es un sistema de aprendizaje basado en el uso de un espacio de hipótesis de funciones lineales en un espacio de mayor dimensión.	Las máquinas van aprendiendo. Usa la función Kernel. La estimación de los parámetros se realiza a través de la optimización de una función de costo.	Se da el caso de "Overtraining" o sobre entrenamiento, donde no se pueden clasificar bien ejemplos nunca antes vistos. (Tomás, David et al, 2005).
k - vecinos más próximos.	Los criterios de vecindad son una de las aproximaciones más conocidas en las técnicas de clasificación supervisada, exigen la definición de una medida de distancia entre los elementos del espacio de representación.	La clasificación de un nuevo punto del espacio de representación, se calcula en función de las clases, conocidas de los puntos más próximos a él.	Su lenta respuesta (método global). Se quiere siempre tener un método local en el que solo los vecinos más cercanos son considerados. (Moreiro González, 2002).

Fuente: Hernández, P., 2004; Lewis, D., 1998; Tomás, David et al, 2005; Moreiro González, 2002.
Elaboración: Las Autoras

1.4.1.4. Máquinas de vectores de soporte (SVM).

Martín (2002), las máquinas de vectores de soporte (Support Vector Machine SVM, por sus siglas en inglés). Es un nuevo sistema de aprendizaje el cual ha tenido un desarrollo muy significativo en los últimos años tanto en la generación de nuevos algoritmos como en las estrategias para su implementación.

Resendiz (2006), SVM es un sistema de aprendizaje basado en el uso de un espacio de hipótesis de funciones lineales en un espacio de mayor dimensión inducido por un Kernel, en el cual las hipótesis son entrenadas por un algoritmo tomado de la teoría de optimización el cual utiliza elementos de la teoría de generalización. SVM es un sistema para entrenar máquinas de aprendizaje lineal eficientemente tanto que para clasificación como para

regresión se han encontrado muchas aplicaciones como clasificación de imágenes, reconocimiento de caracteres, detección de proteínas, clasificación de patrones, identificación de funciones, etc.

- **Algoritmo SVM linealmente separables.**

En el caso de ser linealmente separable, las MVS conforman hiperplanos que separan los datos de entrada en dos subgrupos que poseen una etiqueta propia. En medio de todos los posibles planos de separación de las dos clases etiquetadas como {-1, +1} , existe sólo un hiperplano de separación óptimo, de forma que la distancia entre el hiperplano óptimo y el valor de entrada más cercano sea máxima (maximización del margen) con la intención de forzar la generalización de la máquina que se esté construyendo.

Aquellos puntos o ejemplos sobre los cuales se apoya el margen máximo son los denominados vectores de soporte. Un ejemplo de este caso se observar en la Figura 6.

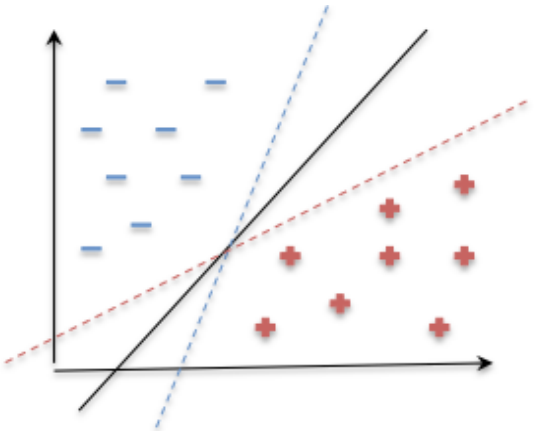

Figura 6. Ejemplo datos linealmente separados.
Fuente: Resendiz (2006)

- **Algoritmo SVM no linealmente separable.**

Para el caso no lineal existen dos casos que vale la pena mencionar:

a) Algoritmo SVM con margen máximo en el espacio de características.

Es de una mayor dimensionalidad y se obtiene a través de una trasformación a las variables del espacio de entrada mediante el uso de una función kernel.

b) SVM con margen blando.

El caso de margen blando, es utilizado cuando no es posible encontrar una trasformación de los datos que permita separarlos linealmente, bien sea en el espacio de entrada o en el espacio de características.

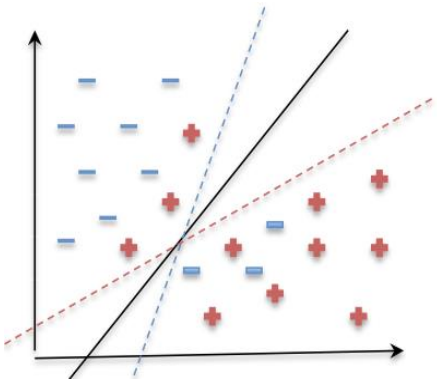

Figura 7. Ejemplo datos linealmente no separados.
Fuente: Resendiz (2006)

SVM con margen máximo en el espacio de características.

Hay casos donde los datos no pueden ser separados linealmente a través de un hiperplano óptimo en el espacio d entrada. En muchas situaciones, los datos, a través de una transformación no lineal del espacio de características y se pueden aplicar los mismos razonamientos que para las SVM lineal con margen máximo.

La transformación de los datos de un espacio inicial a otro de mayor dimensión se logra mediante el uso de la función kernel.

Gunn (1997), una función núcleo o kernel es un producto interno en el espacio de características, que tiene su equivalente en el espacio de entrada.

$$K(x, x') = < \Phi(x), \Phi(x') >$$

Donde K, es una función simétrica positiva definitiva que cumple las condiciones del teorema de Mercer.

De manera gráfica se puede observar en la Figura 8, como la función Kernel permite realizar la separación y el traslado de los datos al espacio de características.

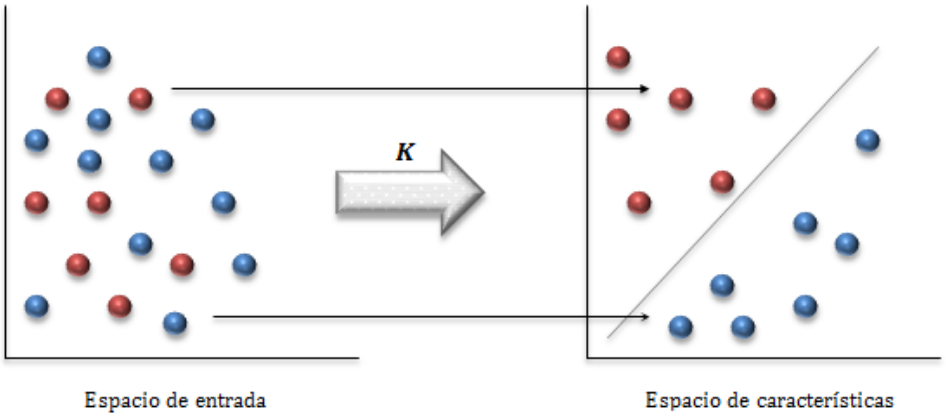

Figura 8. Ejemplo función Kernel.
Fuente: Benavidez, et al., (2006)

Entre los Kernel más comunes, se encuentran: la función lineal, polinomial, RDF (Radial Basic Function), entre otros.

SVM con margen blando.

Este tipo particular de las SVM trata aquellos casos donde existe datos d entrada erróneos, ruido o alto solapamiento de las clases en los datos de entrenamiento, donde se puede ver afectado el hiperplano clasificador, por esta razón se cambia un poco la perspectiva y se busca el mejor hiperplano clasificador que pueda tolerar el ruido en los datos de entrenamiento. Ver ejemplo Figura 9.

Benavidez, et al., (2006), esto se logra relajando las restricciones presentadas en el caso lineal, introduciendo variables de holgura no-negativas $\xi_i \geq 0$.

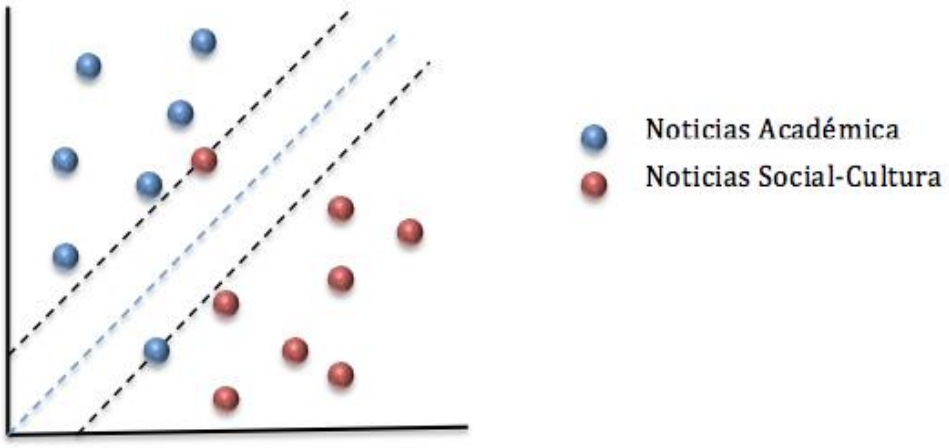

Figura 9. MVS con margen blanco.
Fuente: Modificación de Benavidez, et al., (2006)

De manera que, matemáticamente el problema queda definido como:

Minimizar $\frac{1}{2}||w||+C\sum_{i}^{j} = 1\ \xi_i$

Sujeto a: $y_i(<W_iX_i>+b) \geq 1-\xi_i$

$\xi_i \geq 0,\ i = 1, \ldots, l$

Benavidez, et al (2006), en la función de optimización que se debe solucionar para el modelo de SVM con margen blando se incluye un término de regularización que depende de la magnitud de las mismas y del margen. Además, este término incluye una constante C, que determina la holgura del margen blando. El valor de este parámetro C, el cual debe ser estimado a priori, depende del evaluador. La elección de un valor para este parámetro y el tipo de función kernel influyen en el desempeño de las SVM.

Siguiendo el mismo procedimiento utilizado en el caso linealmente separable, la resolución de este problema en 2.3, viene dada por la búsqueda de los multiplicadores de Lagrange, para esto se construye un Lagrangiano y se resuelve en el problema dual. En física, un lagrangiano es una función escalar a partir de la cual se puede obtener la evolución temporal, las leyes de conservación y otras propiedades importantes de un sistema dinámico.

$$Maximizar \quad \sum_{i=1}^{N}\alpha_i - \frac{1}{2}\sum_{i,j=1}^{N} y_i y_j \alpha_i \alpha_j K(x_i, x_j)$$

$$sujeto\ a: \quad \sum_{i=1}^{N} y_i \alpha_i = 0$$

$$0 \leq \alpha_i \leq C \qquad 1 \leq i \leq N$$

Betancourt, (2005), la función a maximizar es la misma que para el caso de margen máximo, a diferencia de la restricción 0 ≤ α i ≤ C. Esto implica que los datos o patrones que cumplen la condición de tener valores α i ≥ C tienen el mismo comportamiento en la SVM con margen máximo. Es decir, que las SVM con margen máximo se pueden obtener con C = ∞. Esto significa que mientras más grande es el valor de C, más estricta es la SVM al momento de permitir errores, penalizándolos con mayor rigurosidad.

Clasificación con SVM Multiclase

Zubiaga, (2008), menciona que debido a la naturaleza dicotómica de SVM, surgió la necesidad de implementar nuevos métodos que pudieran resolver problemas multiclase. Con este objetivo, se han propuesto diferentes aproximaciones. Por una parte, como

aproximación directa, Weston y Watkins (1998) proponen una modificación de la función de optimización que tiene en cuenta todas las clases:

$$\text{mín}\,\frac{1}{2}\sum_{m=1}^{n}||w_m||^2 + C\sum_{i=1}^{l}\sum_{m\neq y_i}\xi_i^m$$

Sujeto a:

$$w_{y_i}\cdot x_i + b_{y_i} \geq w_m \cdot x_i + b_m + 2 - \xi_i^m, \xi_i^m \geq 0$$

Hsu y Lin (2001), mencionan que diversas técnicas para la aproximación a SVM multiclase de k clases se han basado en la combinación de clasificadores binarios.

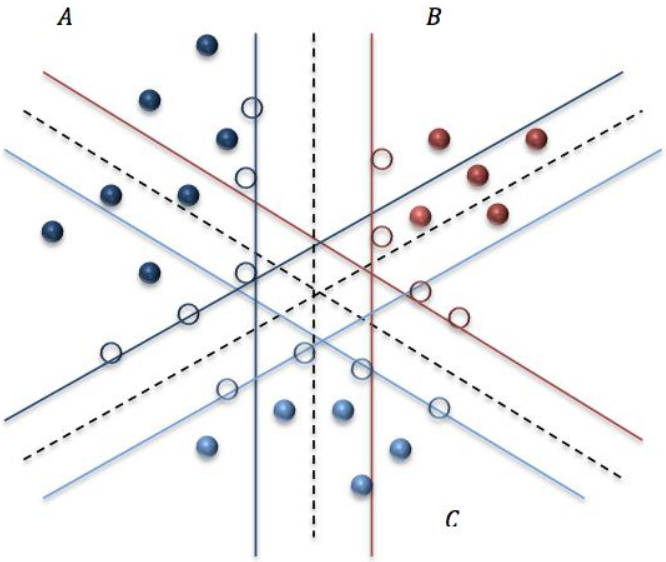

Figura 10. Clasificación MVS multiclase.
Fuente: Modificación de Hsu y Lin (2001)

Uno contra el resto, construye *k* clasificadores que definen otros tantos hiperplanos que separan la clase *i* de los *k-1* restantes. Por ejemplo, para un problema de 4 clases, se crean los clasificadores *1 vs 2-3-4, 2* vs 1-3-4, 3 vs 1-2-4 y 4 vs 1-2-3. Al recibir nuevos documentos, estos son sometidos a los *k* clasificadores, escogiendo como resulta aquella que maximiza el margen:

$$\hat{C}_i = \arg\max_{i=1,\ldots,k}(w_i x + b_i)$$

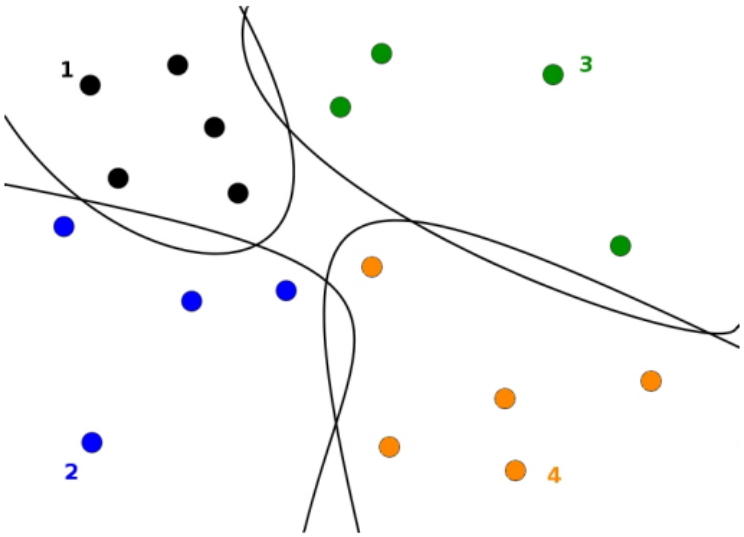

Figura 11. Clasificación una contra el resto.
Fuente: Hsu y Lin (2001)

Uno contra uno, construye k(k-1/2) clasificadores, uno para cada par de clases posible, enfrentando así a todas las clases una a una. Por ejemplo, un problema con 4 clases generaría los siguientes clasificadores: 1 vs 2, 1vs 3, 1 vs 4, 2 vs3, 2 vs 4 y 3vs 4.

Una vez realizado esto, se somete a cada documento de la colección de test a todos estos clasificadores, donde se añade un voto a la clase ganadora para cada caso. Finalmente, aquella que más votos obtenga será la clase propuesta por el sistema.

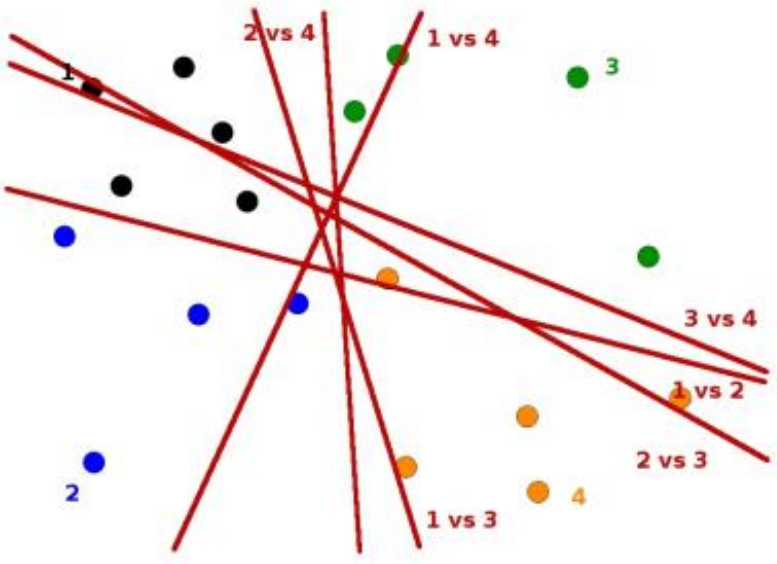

Figura 12. Clasificación uno contra uno.
Fuente: Modificación de Hsu y Lin (2001)

Características de los support vector machines

Colmenares (2009), esta herramienta presenta sus ventajas en comparación con otros algoritmos; sin embargo, también posee algunas desventajas:

Tabla 7. Ventajas y desventajas del algoritmo SVM.

SVM (Support vector machines)	
Ventajas	**Desventajas**
Combina aspectos geométricos fáciles de visualizar y elementos teóricos de importante fundamento, como es Dualidad, condiciones de KKT (Karush-Kuhn-Tucker), funciones Kernel y el Riesgo Estructural.	Trabaja con datos numéricos, y por lo tanto necesita una codificación especial para emplear datos originalmente nominales.
Permite la construcción de separadores no lineales mediante funciones de Kernel gracias a la formulación Dual	Para obtener la clasificación no lineal se necesita proyectar o "mapear" los objetos en un espacio de mayor dimensión. La representación de los productos punto mediante la función Kernel requiere de múltiples cálculos. Esta matriz puede tornarse singular al tener muchos atributos poco relevantes o con valores muy pequeños, por lo que la convergencia de este problema se ve fuertemente afectada.
La función discriminante depende sólo de una cantidad reducida de objetos (los Support Vectors) que están más cerca de la superficie clasificadora, a diferencia de método estadísticos como el Análisis Discriminante, en el cual valores extremos determinan el hiperplano construido. De acuerdo a esto, la velocidad y exactitud del modelo depende sólo de un número reducido de los objetos originales	
Mediante las funciones Kernel se puede obtener el valor de producto punto entre las proyecciones de los objetos. De esta forma, no es necesario realizar explícitamente la proyección de estos.	
Tiene un desempeño computacional más eficiente que otros algoritmos d Inteligencia Artificial (Redes Neuronales por ejemplo).	
Minimiza el problema de sobreajuste o en su terminología en inglés overfitting, debido a que se preocupa de no obtener modelos muy ajustados a los datos, mediante la maximización del margen entre los hiperplanos paralelos al hiperplano separador. Para esto, minimiza la norma del vector normal al hiperplano separador.	

Fuente: Colmenares 2009
Elaboración: Las Autoras

CAPÍTULO II

MATERIALES Y MÉTODOS

La presente investigación es de tipo investigativa y experimental, se encierra en el análisis de la técnica NERC y la aplicación de dos metodologías que son: Procesamiento de Lenguaje Natural y Aprendizaje Automático enmarcando estas técnicas en un esquema de clasificación que se propone y se lo desarrolla a lo largo de la investigación.

En este segundo capítulo, se presenta el esquema de clasificación de noticias UTPL propuesto por el grupo de investigación, teniendo en cuenta que el dominio con el que se trabaja son las noticias publicadas por la Universidad Técnica Particular de Loja. Se describe las herramientas seleccionadas para trabajar, tanto como para la técnica de Procesamiento de Lenguaje Natural como para la técnica de Aprendizaje Automático.

2.1. Esquema de clasificación de noticias.

El objetivo general en la presente investigación es determinar un esquema de clasificación de las noticias publicadas por la UTPL utilizando las técnicas de Clasificación y Reconocimiento de Nombres de Entidades (NERC), a continuación se expone mediante un gráfico el esquema de clasificación planteado:

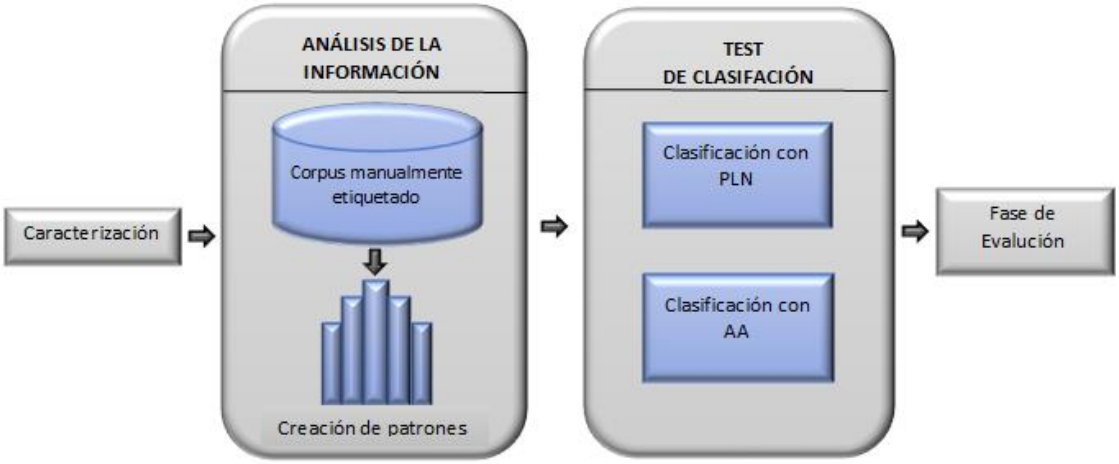

Figura 13. Esquema de clasificación.
Elaboración: Las Autoras

Como muestra la figura 13, el esquema de clasificación planteado, tiene como característica principal la aplicación de los sistemas NERC mediante la aplicación de dos metodologías que son: Procesamiento de Lenguaje Natural y Aprendizaje Automático. Como se puede observar primero se tiene la caracterización, en donde se describe el tipo de información con la que se va a trabajar. Luego se desarrolla la primera fase que es la etiquetación manual, en donde como resultado se obtiene las reglas o patrones, que servirán para trabajar en la fase dos y tres que son PLN y AA respectivamente y finalmente la fase cuatro que se trata de la evaluación de resultados.

2.1.1. Caracterización.

En esta fase del proyecto, se especifica que el dominio con el que se trabaja en todo el proceso de Extracción de Información son las noticias que se encuentra en el sitio web de Universidad Técnica Particular de Loja http://www.utpl.edu.ec/comunicacion/ Para lo cual se tomó una muestra de 400 noticias emitidas en los años 2011 hasta 2014, información suficiente para utilizarla en la fase de entrenamiento y evaluación. Estas noticias tienen un estándar de publicación, cada noticia tiene: título, fecha de publicación y el contenido. A continuación se muestra como son presentadas las noticias en la actualidad.

Figura 14. Ejemplo de noticia, de repositorio UTPL.
Fuente: sitio Web de la UTPL

2.1.2. Etiquetado manual.

Liu, Zhang, Wei y Zhou (2011), indican que en otros sistemas desarrollados normalmente se parte de un corpus anotado para el proceso de definir patrones; sin embargo la presente investigación muestra información nueva y es por este factor que se vio la necesidad de crear un corpus anotado de forma manual).

A continuación se presenta un diagrama, en donde se muestra la metodología a seguir para el proceso de etiquetado manual.

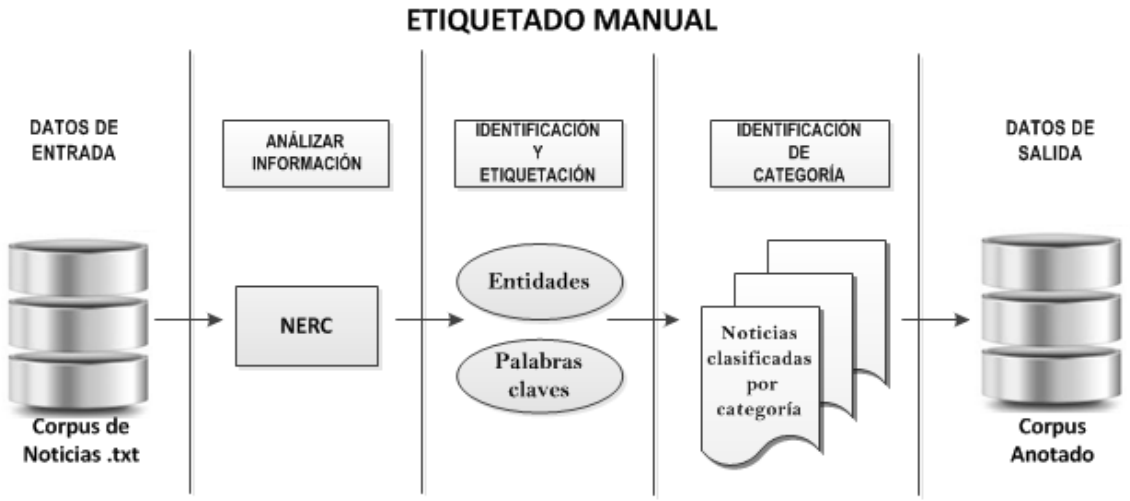

Figura 15. Metodología para proceso de etiquetado manual.
Elaboración: Las Autoras

Como se puede observar en la Figura 15, el diagrama muestra la metodología que se siguió para el etiquetado manual. En donde primero se tiene como dato de entrada un corpus de noticias en extensión .txt, para luego proceder a analizar la información mediante las técnicas NERC. En la herramienta GATE se realiza la identificación y etiquetación de las entidades y las palabras claves, que permitirán definir una mejor clasificación por categorías de las noticias. Finalmente se tiene como resultado nuestro corpus anotado.

2.1.2.1. Consideraciones para el proceso de etiquetación manual.

Martínez, (2008) el etiquetado manual consiste en identificar y etiquetar manualmente los nombres de entidades tanto como: persona, organización, locación, fecha y título de la persona; y también etiquetar las palabras claves que cumplan las características de un tipo de categoría que puede ser: académica, investigativa, social/cultural o de emprendimiento. Para este proceso se ha adoptado las reglas MUC Message Understanding Conference de acuerdo a nuestras necesidades.

Para la definición de las categorías a clasificarse, se consideró en base a las actividades que realiza la Universidad Técnica Particular de Loja durante todo el año y cada categoría tiene sus propias características y criterios que se tomaron en cuenta.

A continuación se describe los criterios que se tomaron en cuenta para etiquetar los nombres de entidad persona, organización, locación y las palabras claves.

Tabla 8. Características y criterios para etiquetar un nombre de entidad.

Entidad	Características
Nombre de persona	Para identificar los nombres de persona se toma en cuenta lo siguiente: • Se debe tener por lo menos dos palabras seguidas con la primera letra que comience con mayúscula ejemplo: Andrea Cueva. • Se dice que es un nombre cuando va seguida de un título de persona ejemplo: (Sr., Dr., Lic., etc.).
Organización	Al identificar las organizaciones se toma en cuenta lo siguiente: • Cuando se tiene siglas se dice que es una organización ejemplo (UTPL, ONU, SENPLADES, etc.). • Se debe tener por lo menos de tres a cuatro palabras seguidas con la primera letra que comience con mayúscula ejemplo: Ministerio de Salud Pública del Ecuador.
Locación	Para identificar una locación se toma en cuenta lo siguiente: • Se refiere a nombres de ciudades por lo que se identifica a una palabra que comienza con mayúscula. • Son nombres de salones, centro de convenciones, instalaciones y se los identifica por esos sustantivos ejemplo (en el salón de la Casa de la cultura, se llevara a cabo en las instalaciones del Teatro Universitario, etc.).
Palabras claves	• Para la identificación de palabras claves de las noticias nos basamos en términos que identifican una categoría ver tabla 9.

Elaboración: Las Autoras

Así mismo a continuación se presenta las características que se consideraron para definir una noticia a la categoría que corresponda.

Tabla 9. Características y criterios para considerar el tipo de categoría de una noticia.

Categoría	Características
Académica	Contempla temas académicos sobre la UTPL: • Jornadas de reflexión académica. • Pruebas de admisión universitaria. • Cursos, Seminarios, congresos que se llevan a cabo durante el año referente a las diferentes actividades de las titulaciones de la UTPL.
Emprendimiento	Contiene temas de proyectos que se ejecutan en la UTPL: • Proyecto de innovación propuesto por estudiantes. • Productos de las empresas de PRENDHO. • Convenios de ayuda que realiza la UTPL con municipios, empresas.
Investigación	Abarca temas de investigación que se llevan a cabo en la UTPL: • Publicación de investigaciones en los diferentes departamentos de la UTPL. • Premios recibidos por proyectos de investigación realizados en la UTPL.
Social/Cultural	Incluye temas sobre actos sociales y culturales que se lleva en la UTPL: • Eventos Deportivos. • Elección de Reinas. • Aniversario de Fundación. • Misión Ecuador. • Familia Idente. • Fechas Festivas (Navidad, Día del Maestro, etc.).

Elaboración: Las Autoras

2.1.3. Metodología para la técnica de procesamiento de lenguaje natural.

A continuación se presenta mediante un esquema, la metodología que se siguió para la clasificación de noticias de la UTPL mediante la técnica de PLN.

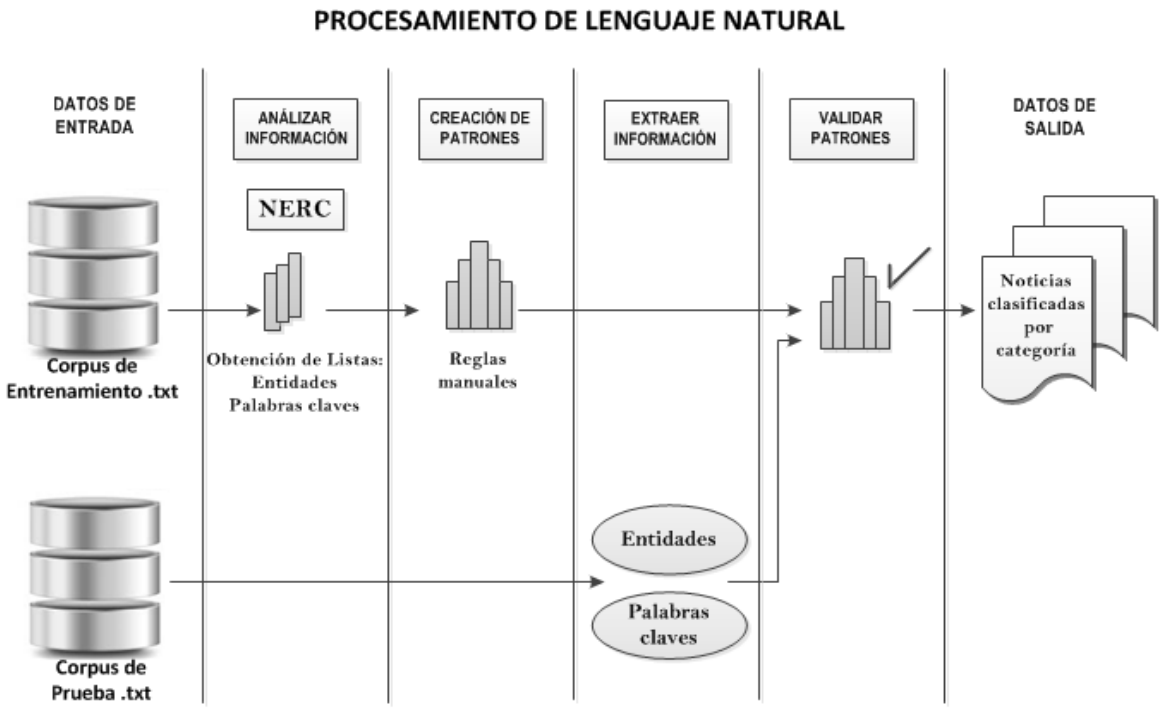

Figura 16. Metodología para proceso de etiquetado manual.
Elaboración: Las Autoras

Como se observa en la Figura 16, el proceso de clasificación de noticias por categoría, mediante la metodología de PLN empieza con la definición de los datos de entrada; para lo cual se tendrá un corpus de doscientas noticias en extensión .txt. El siguiente paso es la obtener listas tanto de entidades como de todas las palabras claves para realizar las reglas; este subproceso es muy importante en el PLN, su solución se da mediante la creación de patrones, en donde se definió las reglas correspondientes para cada caso particular que se encontró, estos patrones ayudarán a definir a la categoría que corresponda cada noticia. El siguiente paso es la validación de los patrones elaborados, una vez realizado este subproceso se tendrá como dato de salida todas las noticias del corpus de entrenamiento clasificadas.

Para finalizar el proceso de PLN y para comprobar el resultado se utiliza el corpus de prueba, para lo cual primero se realiza el subproceso de extracción de la información, en donde se extraen los nombres de entidades y palabras claves para posteriormente, proceder

a validar los patrones que se realizaron en la etapa anterior, y finalmente obtener como datos de salida todas noticias del corpus de prueba, clasificadas por categoría.

La primera herramienta que se utiliza en PLN es Lucene[10], la misma que permite obtener un listado de nombres de entidades, indicando el peso y frecuencia de cada entidad. De Lucene se obtiene un listado de las palabras que más se repiten en las noticias; pero depurando las palabras que no servirían para determinar una categoría, un ejemplo es la entidad organización "UTPL", la misma que se repite en casi o todas las noticias, por lo cual no se la debe considerar para constar en un patrón. Luego se modela mapas conceptuales de todos los posibles patrones que existan para cada categoría de clasificación de las noticias, que puede ser: académica, investigativa, social/cultural y emprendimiento.

La segunda herramienta utilizada es GATE[11], en esta herramienta se ingresa la información que se va a clasificar. Gate permite ingresar las listas o gazzeters obtenidas de la herramienta Lucene y crear los patrones a través de Japes y subirlos al módulo ANNIE; para posteriormente ejecutar y validar estos patrones. Finalmente se obtiene las noticias clasificadas por categorías. Así mismo en GATE se realiza las pruebas, subiendo un nuevo corpus, en donde la herramienta realiza la extracción de entidades y palabras claves, para posteriormente realizar la validación de patrones y proceder a clasificar la información.

2.1.3.1. Selección de herramientas para la técnica de PLN.

En PLN, se utilizaron las herramientas Lucene y GATE, a continuación se describen:

Herramienta Lucene

Fernández, (2009), Lucene es un motor de búsqueda de alto rendimiento escrito en Java y es Open Source. Es escalable y logra su alto rendimiento mediante el uso de índices. Tiene algoritmos de búsqueda eficientes. Estos algoritmos permiten rankear los resultados (haciendo posible obtener primero los mejores resultados); hacer phrase queries (consultas de frases), wildcard queries (consultas comodin), proximity queries (consultas de proximidad), range queries (consultas de rango) y más; búsquedas por campos; ordenar por cualquier campo; y búsquedas a múltiples índices uniéndolos.

La herramienta entrega un documento en Excel, en donde se puede observar todas las entidades nombradas encontradas, cada una con sus respectivos pesos y frecuencias. Este

[10] Lucene. Disponible en: http://lucene.apache.org/core/
[11] GATE, Disponible en: http://www.gate.ac.uk/

resultado sirve como pauta para realizar las reglas en base a los patrones identificados en las noticias; para posteriormente ingresar estas reglas en la herramienta GATE.

Vista de Lucene

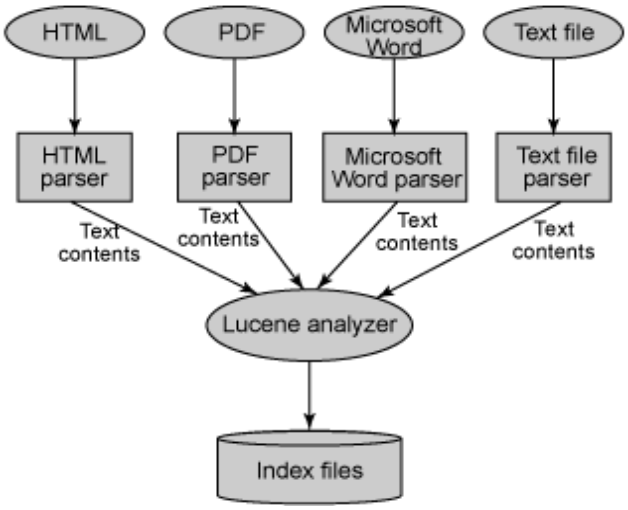

Figura 17. Arquitectura de Lucene.
Fuente: Fernández, (2009)

Como se puede ver en la Figura 17, Lucene trabaja con diferentes tipos de documentos con diferentes extensiones como por ejemplo: HTML, PDF, Microsoft Word y Text file. Estos documentos son analizados en la herramienta y se obtiene un documento resumen de las palabras que más se repiten en las noticias.

Herramienta Gate (General Architecture for Text Engineering)

GATE es una infraestructura open-source (código abierto), basada en Java que sirve para desarrollar y reutilizar componentes de software para resolver diversos problemas en el dominio del Procesamiento de Lenguaje Natural. GATE también permite construir y anotar o etiquetar corpus y evaluar las aplicaciones generadas, conectándose a una base de datos de textos.

Tabla 10: Módulos de GATE.

Nombre de módulo	Definición
Language Resources Recursos de Lenguaje	Que representa entidades como documentos, corpus u ontologías.
Processing Resources Recursos de Procesamiento	Que representa entidades que son en su mayoría algoritmos, como parsers, generadores, etc.
Visual Resources Recursos Visuales	Que representa la visualización y edición de los componentes de la GUI.

Fuente: General Architecture for Text Engineering[11]
Elaboración: Las Autoras

Capa de recursos de procesamiento

En GATE los recursos de procesamiento son las herramientas de reprocesamiento. A continuación se describe a ANNIE Y JAPE, que son los utilizados en esta investigación.

Muños, (2008), *JAPE (Expresiones Regulares)* es uno de los recursos más importantes de GATE. JAPE (Java Annotation Patterns Engine) es un transductor de estados finitos para las anotaciones de los documentos. JAPE permite reconocer expresiones regulares para las anotaciones de los documentos. Este módulo recibe como insumo una gramática JAPE y un documento o un corpus.

ANNIE (a Nearly-New Information Extraction System): es un componente de GATE que contiene recursos y herramientas orientadas a la Extracción de Información. Incorpora, por ejemplo, un amplio abanico de recursos para los distintos niveles de análisis del lenguaje. Los componentes de ANNIE se organizan secuencialmente, tal y como se observa en la Figura 18. (GATE, 2014)

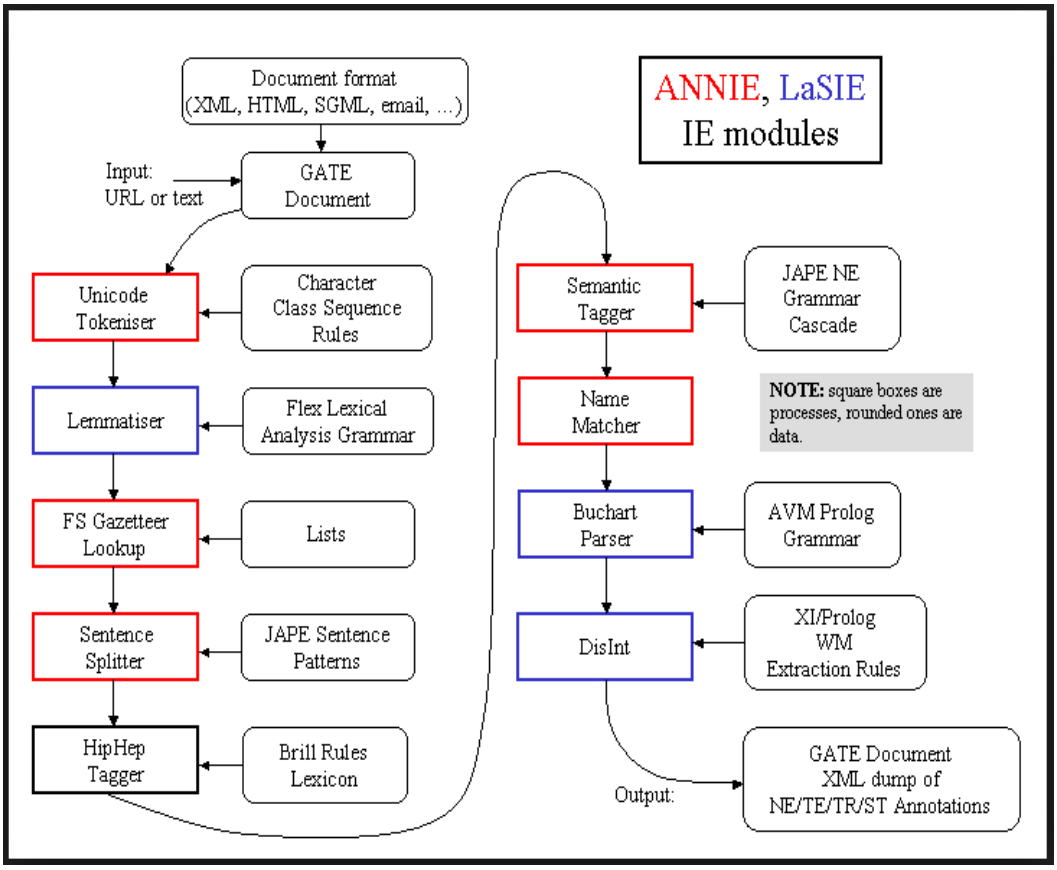

Figura 18. Componentes de ANNIE.
Fuente: Disponible en: https://gate.ac.uk/

A continuación se presenta la descripción de los componentes más importantes del sistema.

Tabla.11: Componentes de ANNIE.

Nombre del recurso	Descripción
Tokeniser (Tokenizador)	Un tokenizador descompone un texto en tokens simples, tales como números, signos de puntuación y palabras de diferentes tipos. Es una de las tareas preliminares cuando se quiere procesar un texto. Como tipos de tokens ANNIE distingue de forma general entre Token y SpaceToken (espacio entre palabras).
Gazetter	Consiste en un conjunto de listas (ficheros de textos plano) en las que se representa conjuntos de nombres, tales como nombres de ciudades, organizaciones, días de la semana, etc., y que se utilizan para reconocer en el texto dichas entidades. Estas listas pueden ser editadas e incluso se pueden crear otras nuevas.
Sentence Splitter	Consiste en un conjunto de transductores de estados finitos que permiten segmentar el texto en oraciones. Utiliza la lista de abreviaturas del *gazzetter* para distinguir los "." que indican el final de una abreviación de aquellos que delimitan las oraciones. Cada oración resultante es anotada con el tipo "Sentence", mientras que a cada delimitador se asocia una etiqueta "Split". Es independiente del dominio y de la aplicación.
Part of Speech Tagger	Es una versión modificada del etiquetador *Brill*, que realiza la anotación de cada palabra o símbolo del texto con su categoría morfológica (*POS Tagging*). Para ello, utiliza un léxico por defecto y un conjunto de reglas extraídas del entrenamiento sobre un corpus extenso de noticias del Wall Street Journal. Existe la posibilidad de modificar tanto el léxico como las reglas.
Semantic Tagger	El etiquetador semántico está basado en el lenguaje JAPE, y contiene un conjunto de reglas que actúan sobre las etiquetas asignadas en las etapas anteriores para anotar las entidades
Orthographic Coreference (OrthoMatcher)	Añade relaciones de identidad entre las entidades anotadas por el etiquetador semántico, con el objetivo de detectar posibles referencias anafóricas. Para ello, utiliza una tabla de cadenas que representan la misma entidad, como por ejemplo, "IBM" y "Big Blue" o "Coca Cola" y "Coke"; y una tabla de cadenas que, erróneamente, se podrían interpretar como representantes de la misma entidad, pero que corresponden a entidades distintas, como por ejemplo, "BT Wireless" y "BT Cellnet".
Pronominal Coreference	Realiza la resolución de referencias pronominales; es decir, identifica a qué entidad de las mencionadas anteriormente en el texto se refiere un determinado pronombre. De nuevo, precisa de las anotaciones realizadas por los otros módulos.

Fuente: General Architecture for Text Engineering[11]
Elaboración: Las Autoras

2.1.4. Metodología para la técnica de Aprendizaje Automático.

A continuación se presenta mediante un esquema, la metodología que se siguió para la clasificación de noticias de la UTPL mediante la técnica de Aprendizaje Automático.

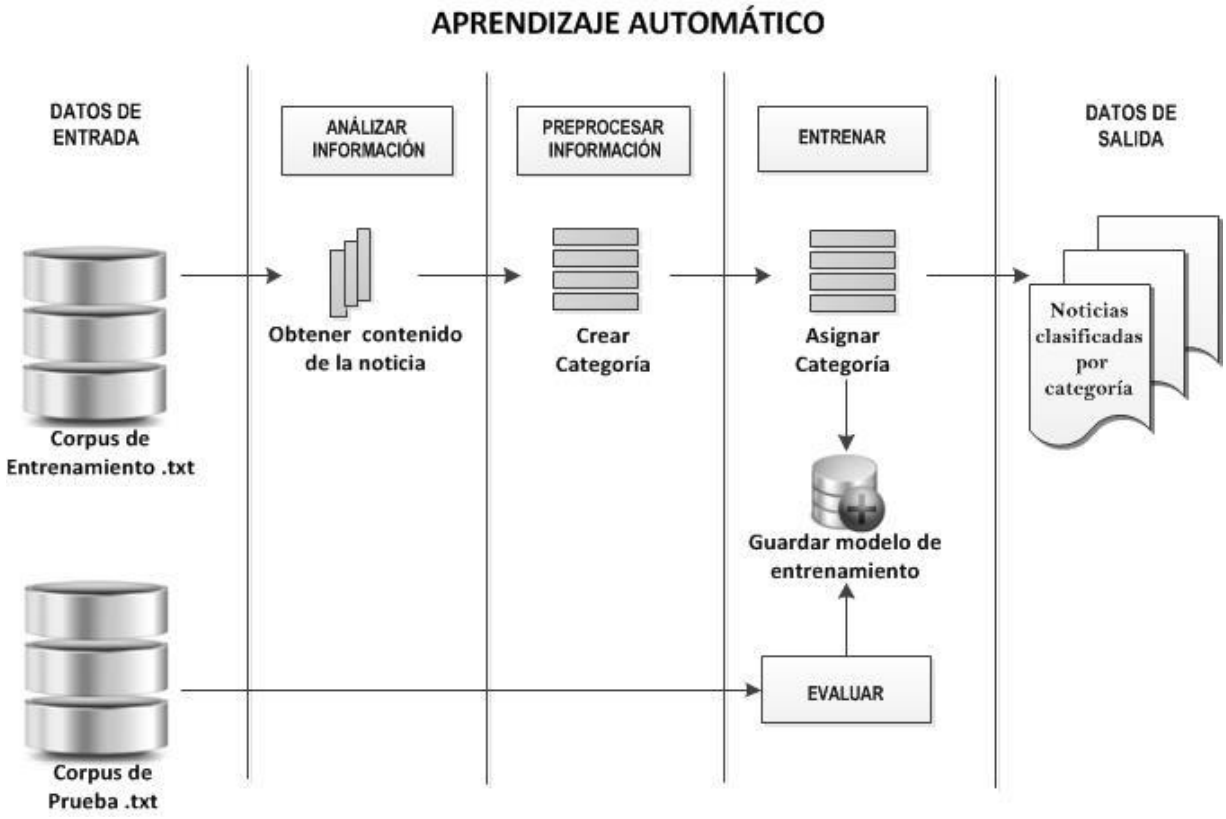

Figura 19. Metodología para la técnica de Aprendizaje Automático.
Elaboración: Las Autoras

Como se observa en la Figura 19, el proceso de clasificación de noticias por categoría, mediante la técnica de AA empieza con la definición de los datos de entrada; donde se tiene un corpus de doscientas noticias en extensión .txt. El siguiente paso es el análisis de la información, donde se llama un Api de Weka, el mismo que utiliza un algoritmo de clasificación SVM, aquí se analiza el contenido; para posteriormente realizar el preprocesamiento y crear las categorías. Mediante un entrenamiento con el corpus que se está trabajando se asigna cada noticia a la categoría que corresponde, obteniendo el corpus clasificado mediante categorías, este modelo de entrenamiento se procede a guardar.

Para finalizar el proceso de AA y para comprobar el resultado se utiliza el corpus de prueba, para lo cual primero se realiza el subproceso de evaluación de la información, en donde es analizado de acuerdo al modelo que se guardó en el proceso anterior y finalmente obtener como datos de salida todas noticias del corpus de prueba, clasificadas por categoría. Cada vez que se evalué un nuevo corpus de prueba se va a guardar un nuevo modelo de entrenamiento.

2.1.4.1. Selección de herramientas para la técnica de AA.

Para trabajar con la técnica de AA, primero se eligió el algoritmo denominado SVM (Support Vector Machines) o Máquinas de Vectores de Soporte, el mismo que es un algoritmo de aprendizaje automático supervisado, este algoritmo permite resolver problemas de clasificación. Otros trabajos utilizan este algoritmo, como por ejemplo en el proyecto de clasificación de preguntas en sistemas de búsquedas de respuestas, en donde hace uso de tres técnicas de aprendizaje: SVM siendo un método de aprendizaje lineal, a través del uso de kernels y reglas no lineales.

Algoritmo SVM

Se eligió trabajar con SVM por que ha sido probado en diversidad de problemas de PLN. Gran parte de problemas de PLN se presenta como un problema de clasificación multi-clase. En donde SVM descompone en un número de tareas de clasificación binario, y lo clasifica y entrena a cada una de las clases.

El SVM es un algoritmo cuyo objetivo es partir de un producto escalar de vectores multidimensionales, en el cual se traza un hiperplano, que separe los vectores de una clase del resto. A este producto escalar se lo conoce como Kernel. El SVM se basa en patrones que ayudan a maximizar la distancia de las clases, para lo cual se encarga de extraer elementos característicos de los documentos.

SVM permite obtener la clasificación de datos linealmente separable y linealmente no separable. EL SVM linealmente no separable, para su solución adopta las técnicas de Margen Suave o de Kernel.

En el proceso de clasificación de las noticias UTPL, se trabajó con el API de WEKA que ya tiene incorporado el algoritmo de SVM y resuelve el problema de la clasificación a través de la biblioteca LivSVM.

Candel, (2011), como se mencionó anteriormente uno de los problemas de PLN, es la clasificación en más de dos categorías. El SVM para solucionarlo utiliza dos métodos, uno contra el resto (one against the rest o oar-SVM) y uno contra uno (one against one o oao-SVM), el primero de ellos consiste en implementar un número *k* de SVMs, en donde la *r-ésima* SVM está encargada de separar a los elementos de la clase *r* del resto de los ejemplos. Por parte el método uno contra uno crea *k (k* 1)/2 SVMs, una para cada par de clases.

En este trabajo interesa clasificar un corpus de 200 noticias, en cuatro categorías mediante SVM, a continuación se indica una representación gráfica de la solución a este problema multiclase.

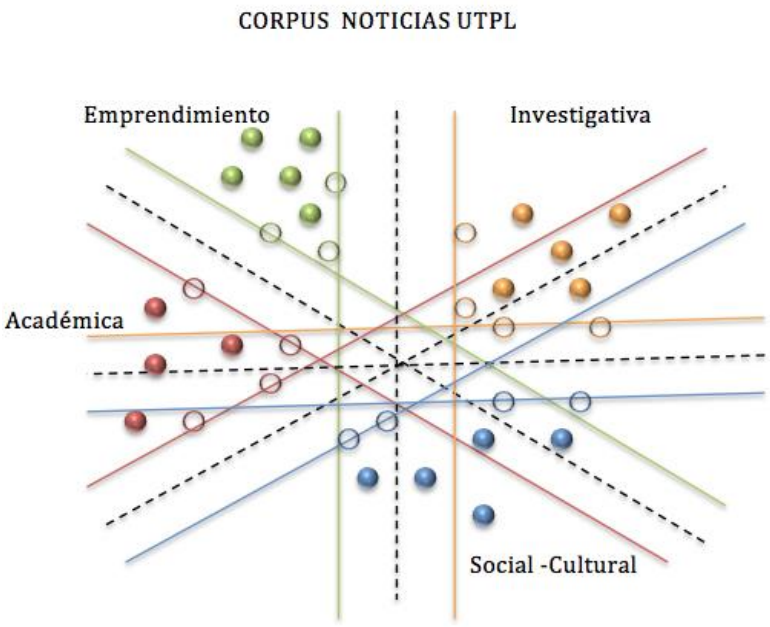

Figura 20. Clasificación multiclase: Caso Noticias UTPL.
Elaboración: Las Autoras

Se puede observar en la Figura 20, cómo se puede separar nuestro problema de cuatro clases a través del algoritmo SVM multiclase utilizando SVM binaria, donde los colores de los márgenes indican a qué clase está asociado el hiperplano. EL SVM multiclase, soluciona el problema planteado.

Weka

También se trabajó con la herramienta Weka, que tiene ya implementado el algoritmo SVM; donde dichas bibliotecas permiten, entre otras tareas, definir los parámetros del algoritmo. Dado un corpus de entrenamiento se etiqueta las clases y entrena una SVM para construir un modelo que prediga la clase de un nuevo corpus.

La herramienta Weka desarrollada en la Universidad de Waikato (Nueva Zelanda) bajo la plataforma Java, se eligió esta herramienta por que tiene la licencia GNU (General Public License) y además cuenta con una colección de algoritmos de aprendizaje que se pueden aplicar directamente a un conjunto de datos o llamados desde su propio código Java. También es muy adecuado para el desarrollo de nuevos esquemas de aprendizaje

automático.[12]

Weka no trae implementado de forma originaria SVM, pero su arquitectura permite agregarlo; en este caso se utilizó LibSVM, un software muy conocido que implementa el algoritmo SVM. Para para poder implementarlo en nuestra aplicación fue necesario trabajar con la biblioteca LibSVM .jar en la carpeta de Weka.

Liu, Zhang, Wei y Zhou, (2011), la biblioteca LibSVM es un archivo JAR, y tiene que estar en la ruta de clase construir el clasificador SVM. Esta permite que el usuario pueda realizar estadísticas de la obtención de las clases clasificadas.

[12] Weka. Disponible en: http://weka.wikispaces.com/LibSVM

CAPÍTULO III

DESARROLLO DEL ESQUEMA DE CLASIFICACIÓN DE NOTICIAS, BASADO EN NERC.

En el tercer capítulo, se presenta el desarrolló del esquema de clasificación propuesto por el equipo de trabajo. Antes de entrar al desarrollo de las técnicas de PLN y AA, primero se realizó el proceso manual de etiquetación manual de las 200 noticias, este corpus sirve como entrenamiento para las dos técnicas antes mencionadas. A continuación se explica el proceso de cada una de las fases del esquema de clasificación de noticias.

3.1 Problemática.

La Universidad Técnica Particular de Loja en su portal Web, ha venido publicando noticias que se van generando en la institución. Dentro del proceso de publicación existen encargados de postearlas. Esta información se visualiza en la página, donde el usuario tiene la opción de buscar la información de su preferencia; este buscador presenta noticias que a veces no cumple con las expectativas del lector.

El enfoque de este proyecto es dar una mejor visualización de las noticias publicadas por la UTPL, en donde el usuario pueda observar la noticia mediante categoría, tales como: académica, investigativa, social/cultural y emprendimiento.

En base a la problemática descrita, se propone el diseño y desarrollo de un esquema de clasificación de noticias, mediante las técnicas NERC, tiene como característica principal, la ejecución de dos técnicas que son: procesamiento de lenguaje natural y aprendizaje automático, las mismas que utilizarán NERC, para la realización de patrones y reglas, para un mejor y optimo clasificador. Finalmente se propone la comparación de las dos técnicas, indicando el porcentaje de precisión y recall de clasificación de las noticias.

3.1.1 Diagrama de la presentación actual de las noticias UTPL.

A continuación se representa de forma gráfica, como el usuario final observa las noticias publicadas en el portal de la Universidad Técnica Particular de Loja, y como se da el proceso para que las mismas sean publicadas.

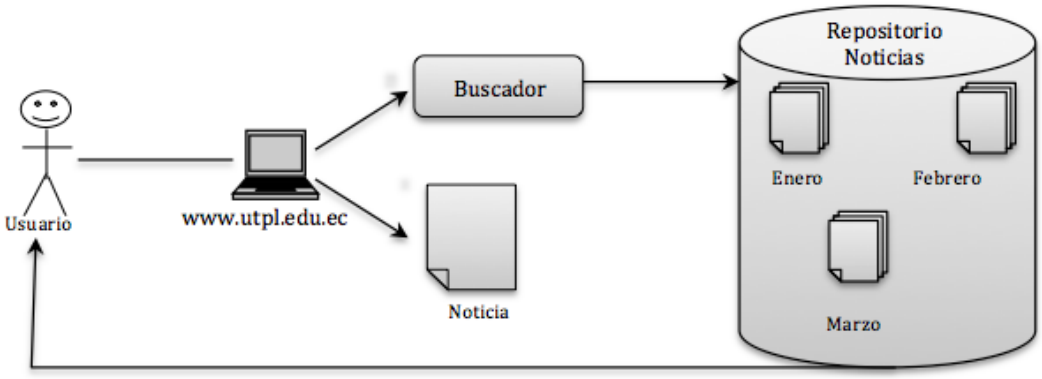

Figura 21. Presentación actual de las noticias UTPL.
Elaboración: Las Autoras

Como se puede observar en la Figura 21, el usuario puede acceder a la información de la noticia, a través del portal de la UTPL; en donde la noticia se encuentra posteada de acuerdo a como se va generando la misma durante el mes. El usuario también tiene la opción de buscar una noticia de acuerdo a la temática o persona relacionada en la noticia, el resultado que obtiene es una lista de noticias, que contienen la palabra enviada a buscar; puesto que en muchos de los casos no satisface la necesidad real del usuario.

3.2 Creación del corpus etiquetado manualmente.

La tarea de extracción de información es compleja y no se han resuelto todos los problemas que involucra. Sin embargo, en ciertos dominios, como los estudiados por la MUC, la complejidad disminuye. En dominios específicos de este tipo se pueden predefinir plantillas con espacios de inserción que codifican los eventos relevantes. La plantilla comúnmente se diseña tomando en cuenta las siguientes categorías:

- **Entidades:** personas, organizaciones, lugares, fechas, etc.
- **Atributos** (de las entidades) como título de una persona, tipo de organización, etc.
- **Relaciones** (que existen entre las entidades), por ejemplo: la organización X está ubicada en el país.
- **Eventos** (en los cuales las entidades participan), por ejemplo: la empresa X _firmó un acuerdo con la empresa Y.

En este proceso se determinan de forma manual y mediante mapas conceptuales los diferentes patrones para definir a qué tipo de categoría pertenece la noticia. Las cuatro categorías determinadas son: académica, investigativa, social/cultural y emprendimiento.

Una vez que se tiene el corpus de doscientos documentos, mediante la herramienta GATE, se procede a la etiquetación de las entidades nombradas que se encuentran en cada noticia y las palabras claves, para mediante su respectivas relaciones poder definir a que categoría corresponden.

3.2.1 Etiquetado de nombres de entidades y palabras claves.

Para la etiquetación de los nombres de entidades, se basó en las reglas que plantean en las conferencias MUC (Message Understanding Conference), en donde se definen cinco tipos de nombres de entidades con los cuales se trabaja en este proyecto, estas son: persona, locación, organización, fecha y título de persona.

Dentro del proceso de etiquetación se agrupa cada entidad encontrada a donde pertenece; por ejemplo si se encuentra la entidad Esmeraldas se la etiqueta dentro del grupo locación.

Para el caso de las entidades tanto como persona y organización, para que la clasificación de noticias sea más eficiente se considera lo siguiente:

A continuación se explica el proceso manual de etiquetación, para dos noticias una académica y la otra emprendimiento, en donde se expone un cuadro resumen de las entidades y palabras claves encontradas. Además se expone el mapa conceptual que resulta de la noticia para ser considerada dentro de un tipo de categoría.

Primer ejemplo: Noticia académica.

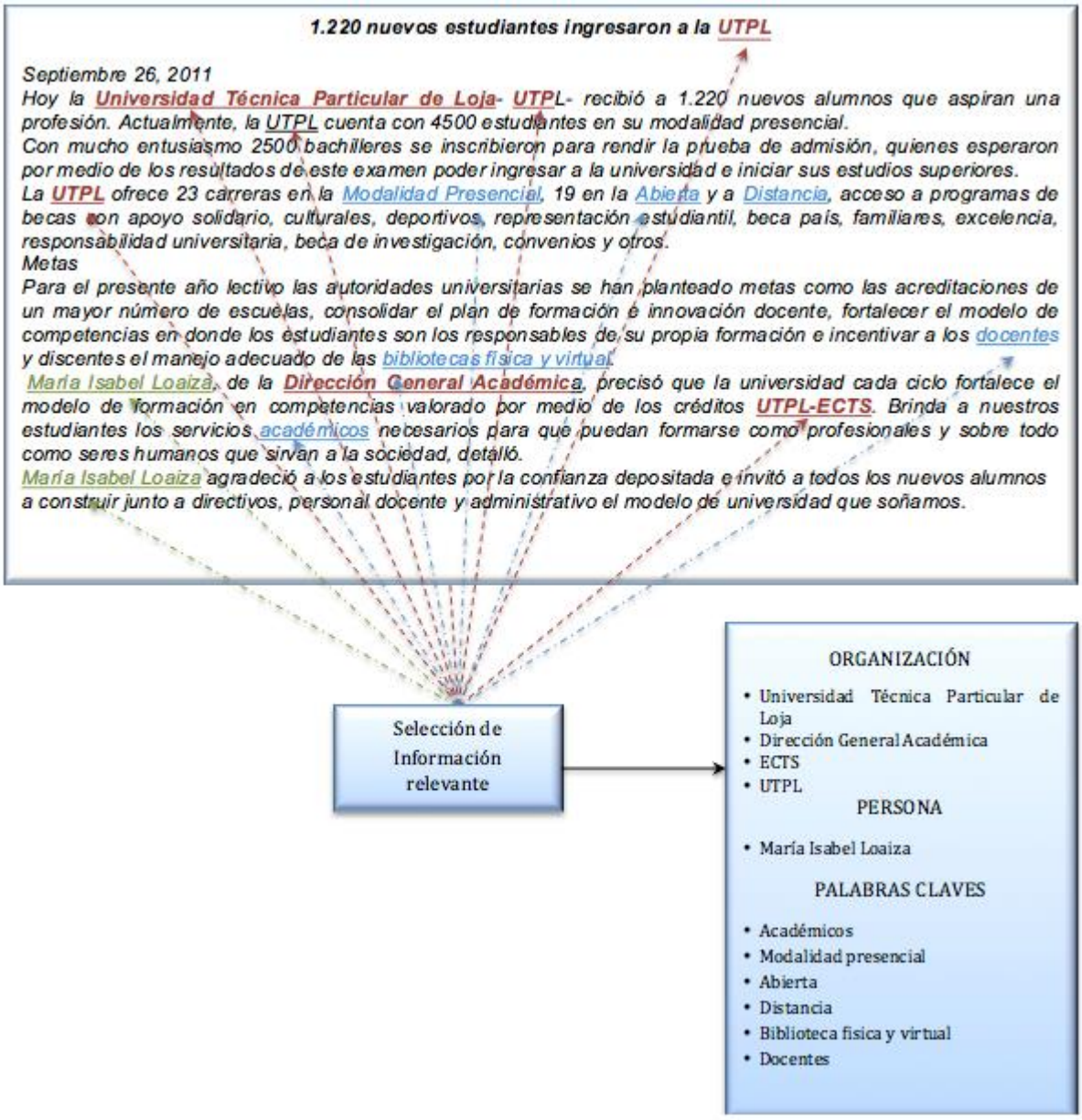

Figura 22. Etiquetación de noticia académica.
Elaboración: Las Autoras

Una vez identificada las entidades que intervienen en la noticia, se procede a reconocer la relación que existe entre ellas. Como se muestra en la Figura 23, cada vez que encuentra al nombre entidad persona: *"María Isabel Loaiza"* se la asocia con el nombre de entidad organización: *"Dirección General Académica",* y a la vez también se relaciona con la entidad organización UTPL. Las entidades de persona y organización se relacionan, indicando un patrón que permitirá clasificar la noticia, en este caso se la identifica como una noticia académica. Conociendo las actividades que realiza la UTPL se puede deducir que esta relación encontrada tiene mucho sentido, debido a que la UTPL cuenta con el departamento de Dirección General Académica, que es el encargado de las diferentes actividades académicas dentro y fuera de la Universidad, también se sabe que *"María Isabel Loaiza"*, trabaja dentro del mismo departamento, y si en otra noticia se encuentra estas entidades, se sabe que es una noticia académica. Para reforzar el conocimiento de cuando una noticia es académica, se extrae palabras claves como *"modalidad, abierta, presencial, docentes, académicos, biblioteca, virtual y física",* que ayudarán a clasificar de mejor manera más precisa. En la UTPL cuenta con modalidad de estudio presencial y a distancia, tiene docentes para ambas modalidades y posee una biblioteca física y virtual. Estas palabras como se puede ver identifican una actividad académica.

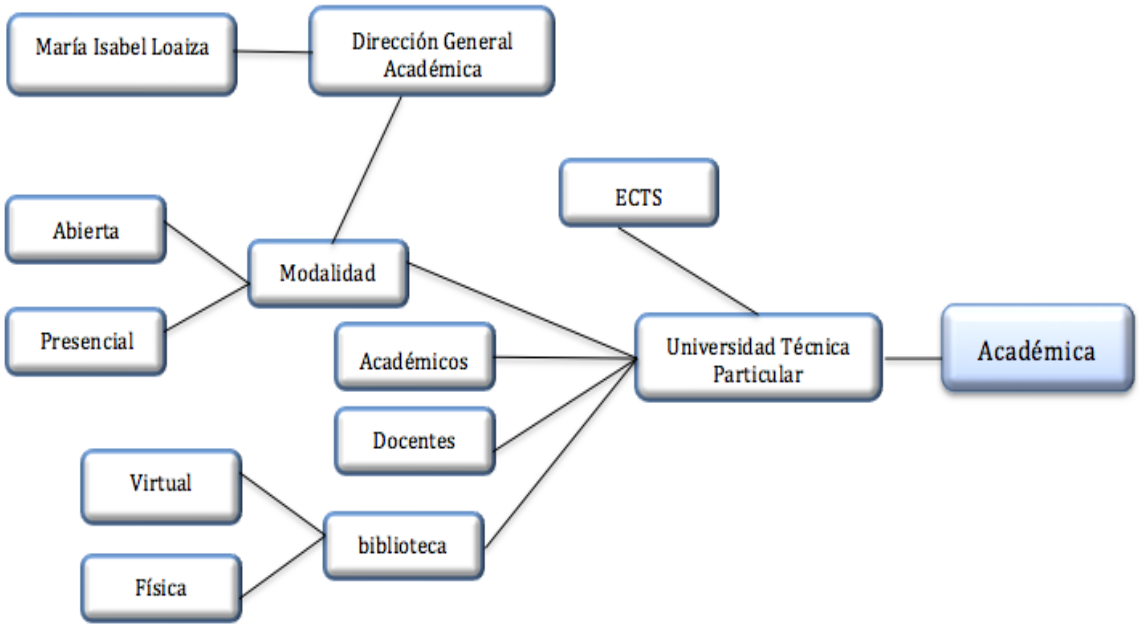

Figura 23. Mapa conceptual de noticia Académica.
Elaboración: Las Autoras

Segundo ejemplo: Noticia de emprendimiento.

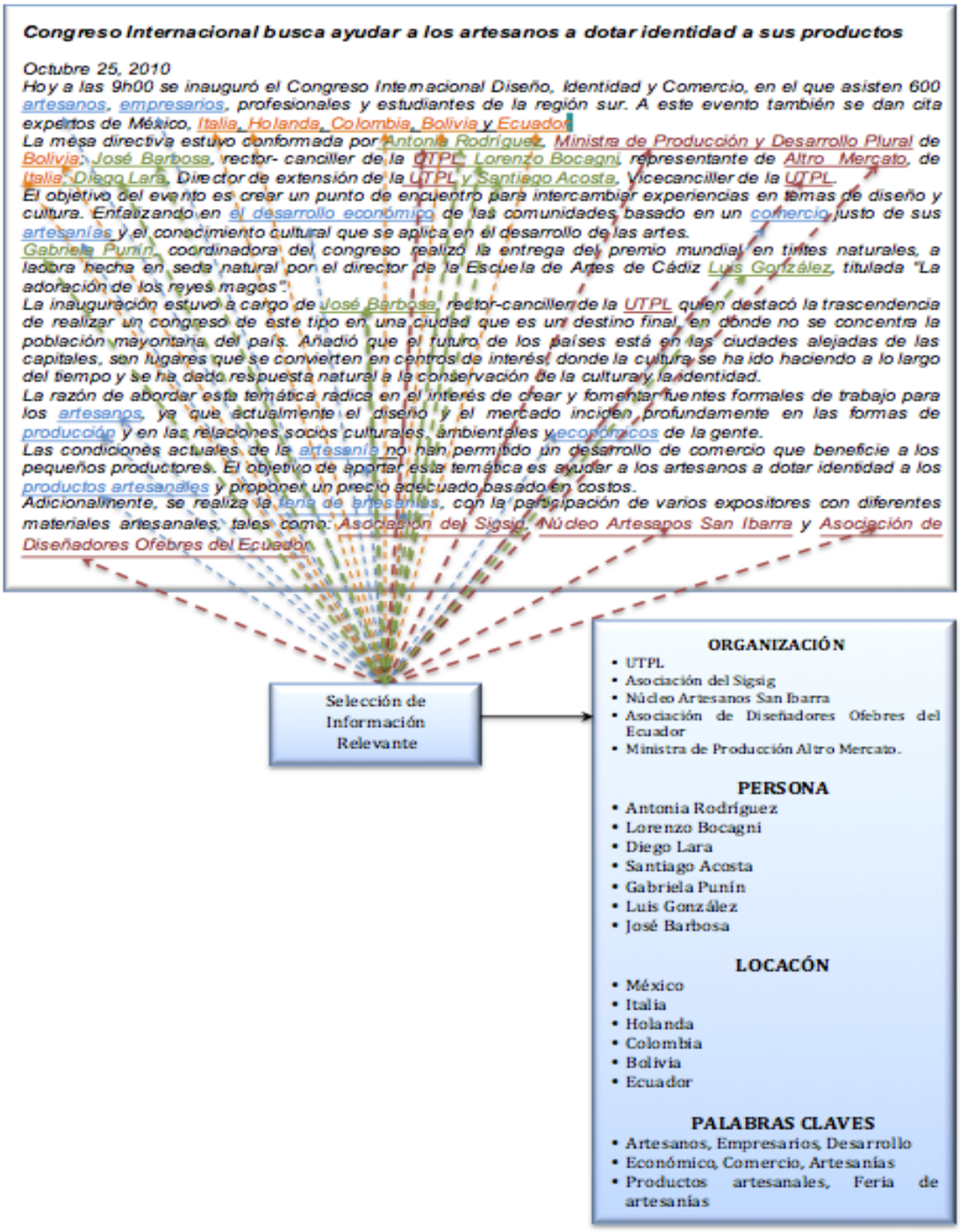

Figura 24. Etiquetación de noticia emprendimiento.
Elaboración: Las Autoras

De igual forma que el ejemplo anterior, se identifica las entidades que intervine en la noticia y se procede a reconocer la relación que existe entre ellas. Como se muestra en la Figura 24 en este caso se la identifica como una noticia de emprendimiento. Comprendiendo que UTPL apoya a los diferentes proyectos de emprendimiento y gestión de los pequeños y grandes empresarios, con este conocimiento se identificó las palabras claves: *"desarrollo,*

económico, artesanal, empresarial, comercio, producto, feria", las mismas ayudaran a clasificar mejor las noticias. También se identifica organizaciones de empresas que cada vez que las encuentre en una noticia dentro de la UTPL se sabrá que es una noticia de emprendimiento.

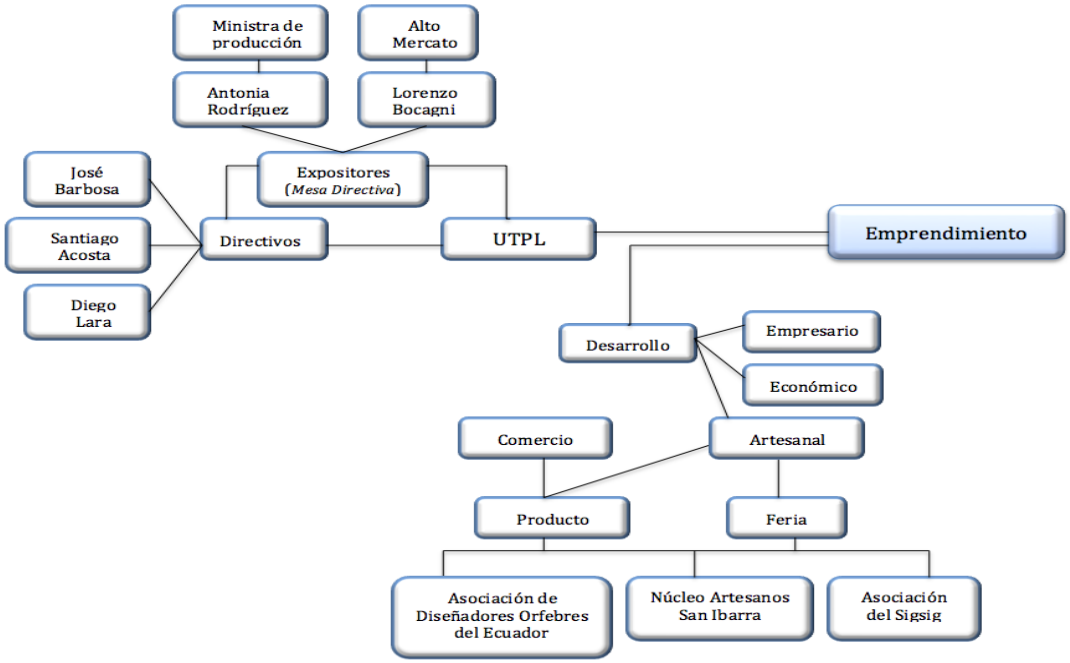

Figura 25. Mapa conceptual de noticia Académica.
Elaboración: Las Autoras

A continuación se muestra en la Figura 26, como se realiza el etiquetado manual en la herramienta GATE. Para ver todo el procedimiento de etiquetado, ver Anexo de GATE.

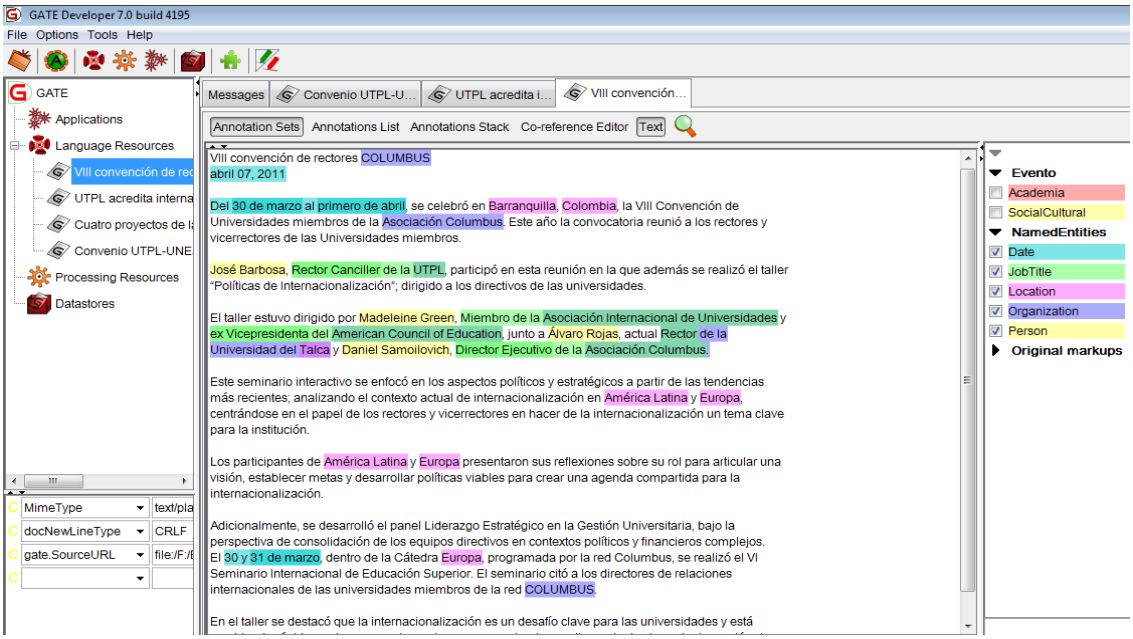

Figura 26. Etiquetación de nombres de entidades en la herramienta GATE.
Elaboración: Las Autoras

A continuación se presenta el diagrama que se considera para la etiquetación manual de entidades y palabras claves, las mismas que permiten definir cuando una noticia es: académica, investigativa, social/cultural o de emprendimiento.

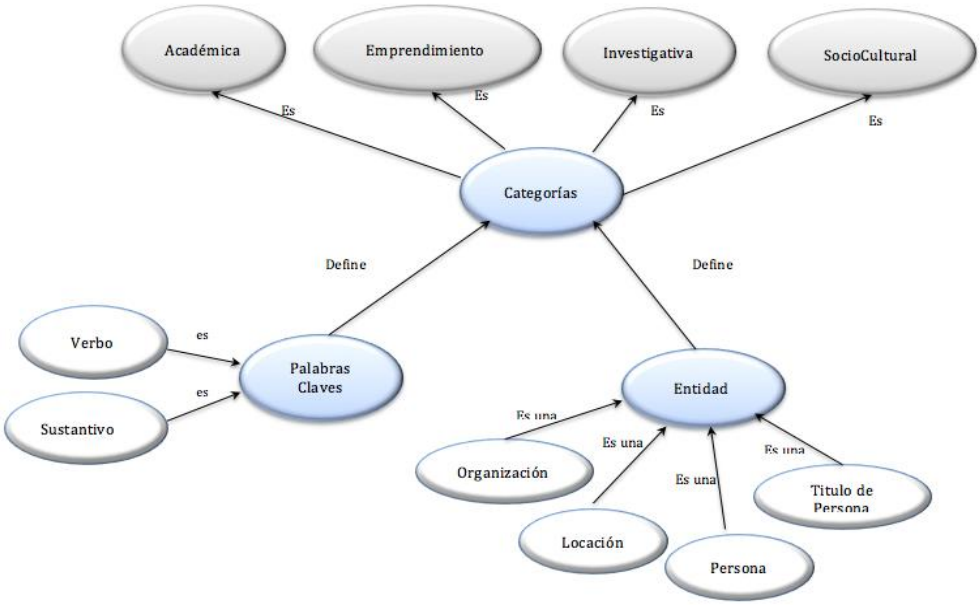

Figura 27. Diagrama general para clasificación de las noticias por categorías.
Elaboración: Las Autoras

3.3 Clasificación de noticias mediante la técnica de PLN.

Según Sosa (1997), el Procesamiento de Lenguaje Natural se define como el reconocimiento y utilización de la información expresada en lenguaje humano a través del uso de sistemas informáticos. En base a esta definición, se seleccionó la herramienta GATE (General Architecture for Text Engineering), la cual permite trabajar el Procesamiento de Lenguaje Natural, para lo cual se presentará las noticias de un corpus de prueba, clasificadas en las cuatro categoría predichas, que son: académicas, investigativas, social/cultural y emprendimiento.

A continuación se expone todo el proceso que se sigue del Procesamiento de Lenguaje Natural (PLN), con la utilización de las herramientas Lucene y GATE.

3.3.1 Lucene.

Para el proceso de PLN, se vio la necesidad de utilizar una herramienta que ayude a identificar la mayor concurrencia de palabras, para poder determinar los patrones y clasificar en la categoría correspondiente. Lucene es la herramienta que a través de su motor y diferentes algoritmos de búsqueda permite obtener resultados como por ejemplo: consultas de frases, consultas comodín, consultas de proximidad, consultas de rango, entre otras.

Lucene, muestra los resultados mediante un documento de Excel, en donde muestra cada palabra con su respectivo peso, esto significa el número de veces que se repite la palabra en el corpus de noticias. En la herramienta se utiliza la consulta range queries (consultas de rango), para obtener las palabras que se encuentran repetidas en casi todas las noticias, para así eliminarlas manualmente de la lista; siendo que no sirve como un indicador de clasificación. Y esta fue la pauta para poder determinar de forma manual las categorías de las noticias. Ver Anexo de instalación de Lucene.

3.3.2 GATE (General Architecture for Text Engineering).

GATE es una herramienta que incluye un conjunto de métodos para PLN, en varios lenguajes, está escrita en java y permite el análisis de texto utilizando un componente llamado ANNIE (Nearly – New Information Extractions System), además contiene otro componente JAPE (Java Annotation Patterns Engine), el cual permite reconocer anotaciones utilizando gramáticas que están constituidas por patrones y reglas.

Para el desarrollo de la primera técnica denominada Procesamiento de Lenguaje Natural (PLN), se trabajó sobre la herramienta GATE; puesto que la misma permite la operación manual de la información, en donde se carga textos o corpus, para luego ser procesados con diversos recursos y finalmente ser evaluados. El entorno gráfico de GATE, permite hacer aplicaciones PLN, sin escribir ni una línea de código.

Modelo de componentes de GATE

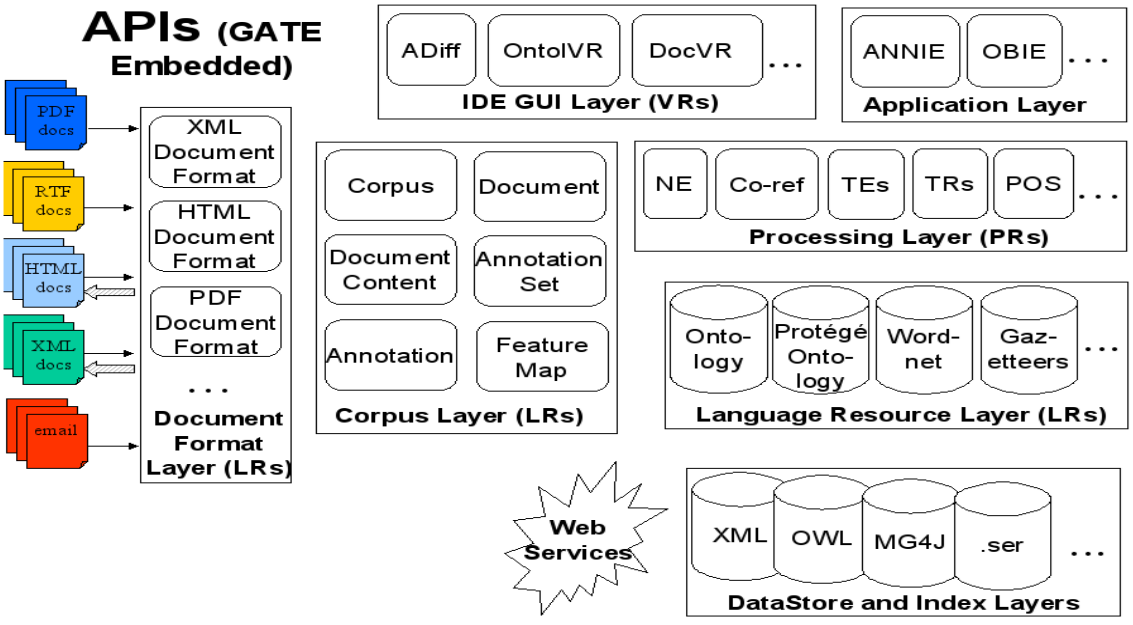

Figura 28. Componentes GATE.
Fuente: GATE "General architecture for text engieering" (2015). The University of Sheffield. Disponible.

En la figura 28, se puede observar que los componentes de GATE, se encuentran por capas reutilizables. GATE trabaja con diferentes formatos de archivos y también tiene la opción de transformar un archivo a otro formato; por ejemplo .PDF, .HTML, XML, etc.

3.3.2.1 Creación de Gazzetters.

Una vez terminado el proceso con la herramienta Lucene, se procede a la creación de listas o Gazzetters, que sirven posteriormente para crear las reglas (JAPE). Ver Anexo Gate

Los Gazzetters que se obtuvieron dentro de los nombres de entidades son:

- Personas.lst
- Organizaciones.lst
- Locaciones.lst

Así mismo se crea un Gazzetters, para cada categoría:

- Académica.lst
- Investigativa.lst
- Socio/Cultural.lst
- Emprendimiento.lst

3.3.2.2 Creación de reglas JAPE.

Para la creación de los JAPE se utiliza un lenguaje por cada regla, en la siguiente tabla se explica el significado de cada sentencia:

Tabla.12: Sentencias JAPE.

Rule: academicPersonasRuler	**// Aquí se establece el nombre a cada regla en este caso academicPersonaRuler**
Priority:20	// Se da el valor de la prioridad de la regla
Lookup.minorType==academic_personas	// Se realiza una anotación de búsqueda “Lookup” con la función “minorType”, que permite buscar del gazzeter academic_persona.
:academicoPerson	// Se asigna una etiqueta a la regla.
- - >	// Es el separador
: academicoPerson.PersonaAcademica = {kind= "academicPersonasRuler" }	// Se refiere a que academicoPerson, se le da una anotación de PersonaAcademica y una función a la clase "academicPersonasRuler"

Elaboración: Las Autoras

3.3.2.3 Componente ANNIE.

Para ejecutar el módulo de ANNIE (*Nearly-New Information Extraction System)*, se necesita un corpus de prueba sin ningún tipo de manipulación, es decir sin palabras etiquetadas manualmente; el cual servirá para probar los resultados y comprobar si funcionan las reglas que se crearon en JAPE. Ver Anexo GATE.

Como resultado se obtiene cada noticia del corpus etiquetada de acuerdo a la categoría que pertenece. Ver Figura 29.

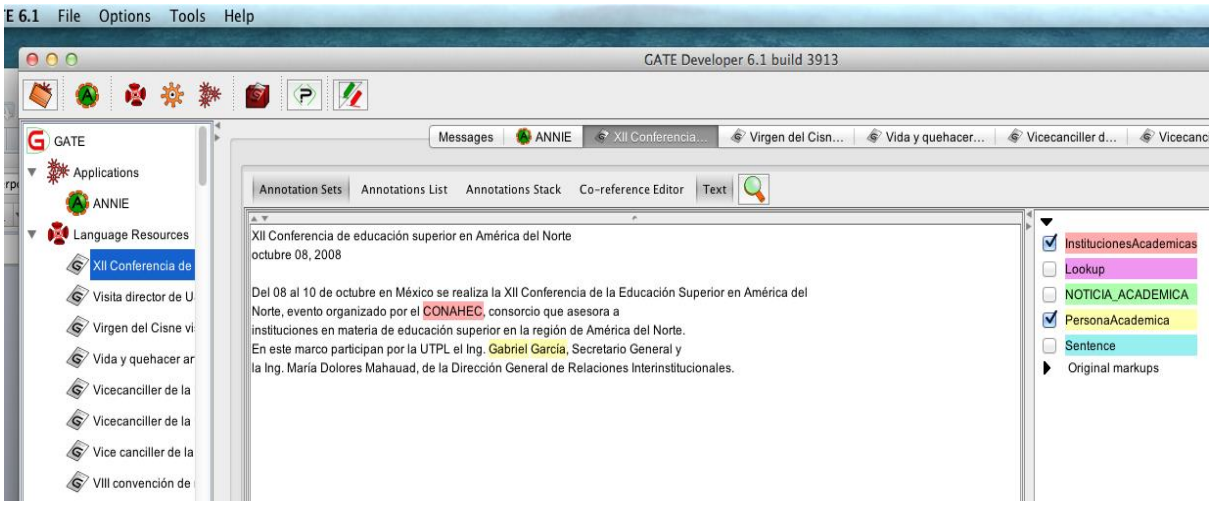

Figura 29. Resultados de ANNIE
Elaboración: Las Autoras

3.4 Clasificación de noticias mediante la técnica de AA.

3.4.1 Creación del DEMO de Clasificación.

Para está fase de construcción del DEMO de clasificación, se necesita tener instalado la herramienta WEKA, de donde se va a consumir el algoritmo de clasificación SMV, mediante la biblioteca LivSVM.jar y para el desarrollo de la interfaz gráfica se trabaja entorno de desarrollo NetBeans en el lenguaje Java. En esta parte ya se cuenta con el corpus de noticias preclasificada mediante la herramienta GATE. El mismo que será nuestro corpus de entrenamiento, para la obtención de patrones.

Una tarea común en el Procesamiento del Lenguaje Natural (PLN) y el aprendizaje automático es clasificación de documentos en categorías.

3.4.1.1 Proceso de desarrollo para el demo de clasificación de noticias.

Para el desarrollo del clasificador, primeramente se debe unificar nuestro corpus en un único archivo .arff para poder ser utilizado por la herramienta WEKA que es de donde se aplica el algoritmo SVM, como lo indica el caso de uso en la Figura 30.

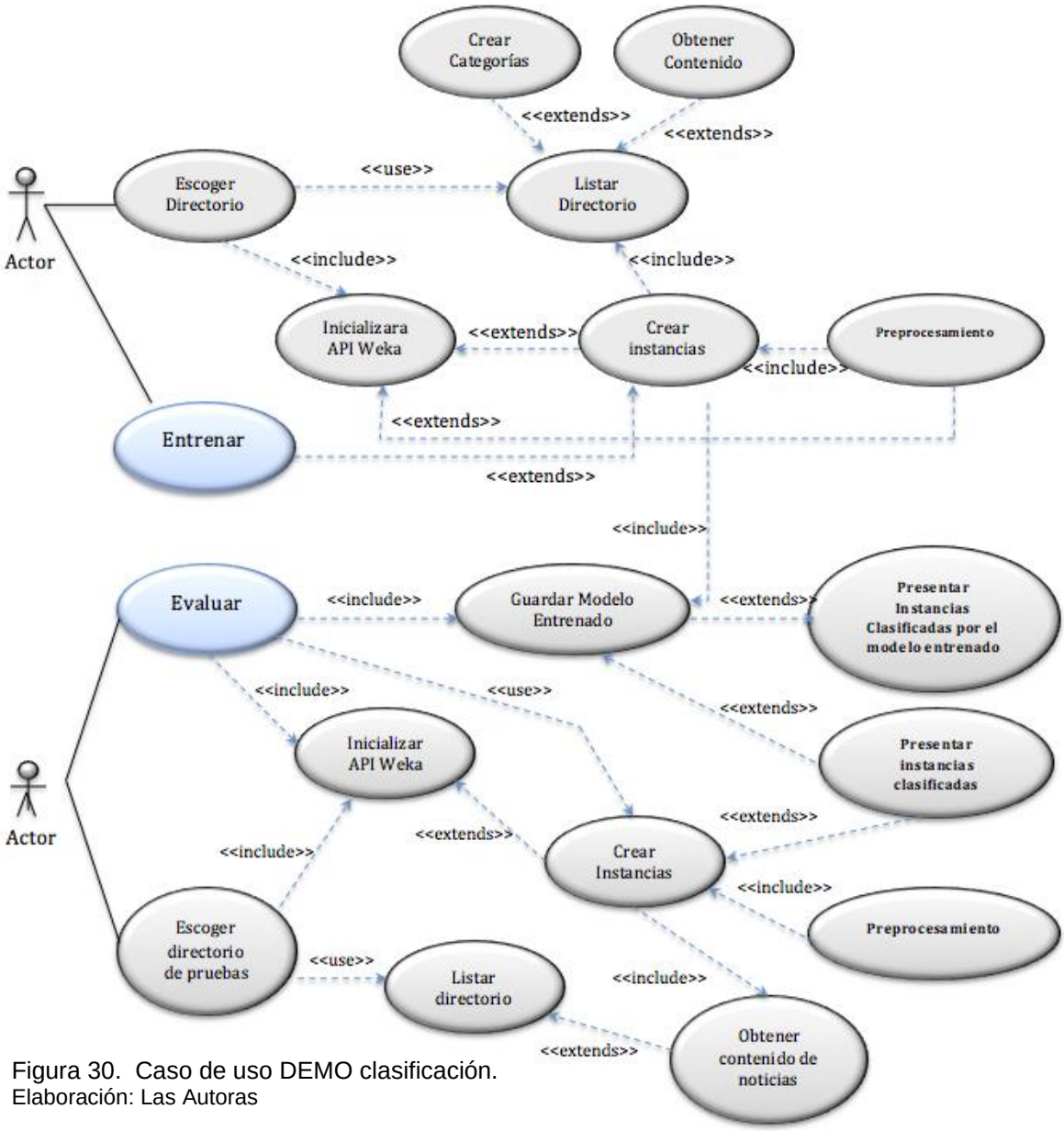

Figura 30. Caso de uso DEMO clasificación.
Elaboración: Las Autoras

Inicializa API_Weka: Mediante la interfaz se realiza la conexión con el API de Weka que permitirá hacer le llamado del Algoritmo SVM, para lo cual Crea una Instancia que se comunicara con Listar Directorio.

Listar Directorio: Este recibe los datos los mismos que son trasformados a un vector para luego ser utilizados en la extensión de categoría y contenido.

Crear categoría: Se analizan los datos y se establece una categoría de acuerdo a los patrones previamente establecidos.

Obtener Contenido: Se analizan los datos de acuerdo a los patrones y se identifican palabras claves que nos servirán para en un futuro determinar una categoría.

Guarda el modelo entrenado: Aquí se almacena la información obtenida para la determinación de las categorías.

Ingresa el corpus de prueba: Son los nuevos datos a evaluar para donde la interfaz inicializa la conexión con Weka para la ejecución del SVM.

A continuación se presenta el diagrama de clases, utilizado en la aplicación:

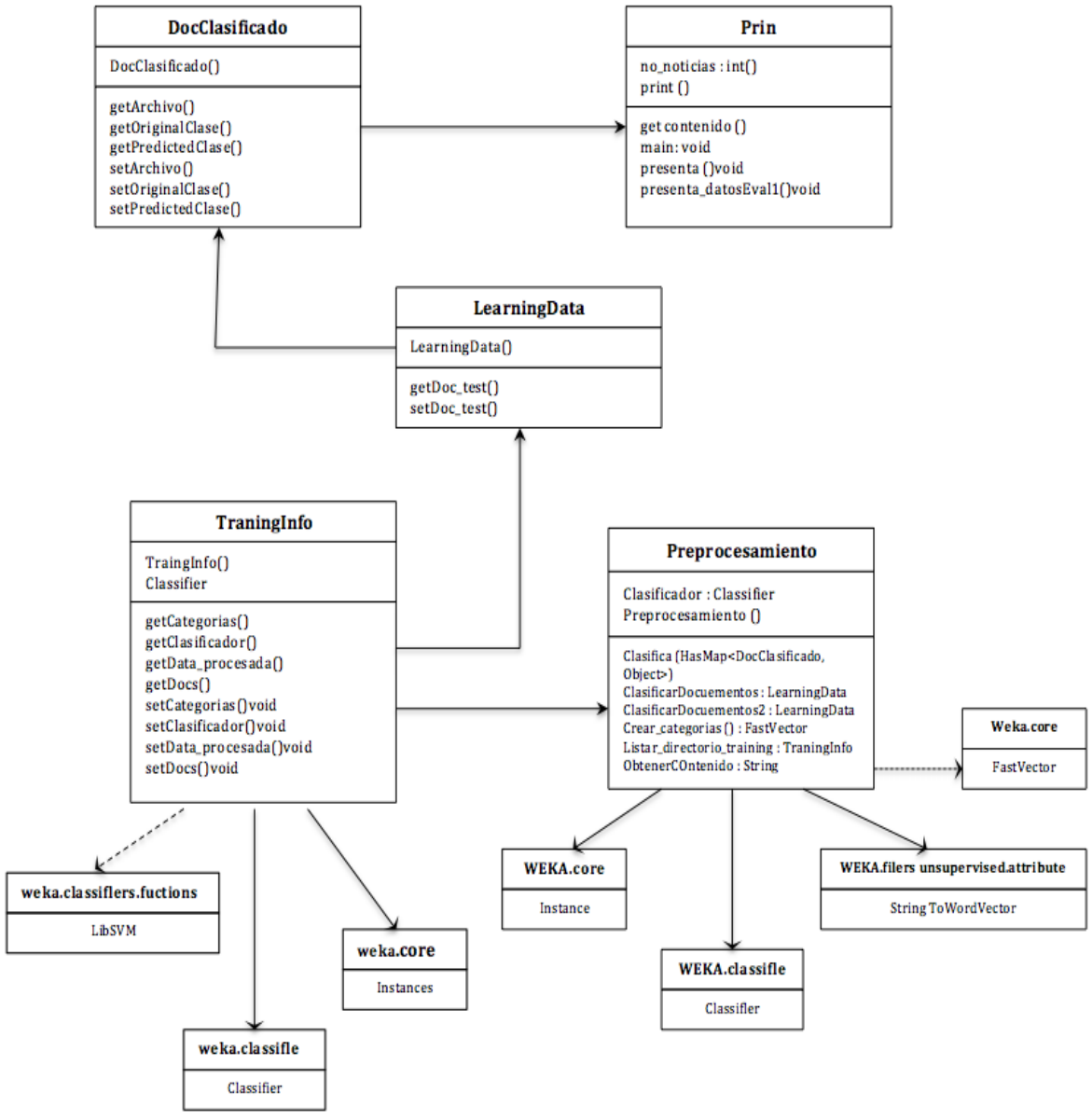

Figura 31. Diagrama de clases.
Elaboración: Las Autoras

Desde la interfaz de la herramienta, se escoge el directorio donde se encuentra los documentos organizados por carpetas y los cuales se van a clasificar, en el programa se lista recursivamente el directorio en donde se asocia el documento con la categoría a la que pertenece (carpeta que lo contiene). Las categorías así como los documentos se almacenan en un objeto HasMap (categorias, y docs) el cual asegura que los elementos contenidos en él sean únicos.

```
public TraningInfo listar_directorio_training( File[] listaFicheros ) throws Exception{
categorias = new HashMap<Object,Object>();
docs = new HashMap<Object,Object>();
int nodocs =0;
for (File fichero : listaFicheros) {
   if( fichero.isDirectory()  ){
      categoria = fichero.getName();
      categorias.put( categoria, 0 );
      File[] txts = fichero.listFiles();
                int no = 1;
          for(File txt: txts){
                 if( txt.getName().contains(".txt") ){
                 nodocs ++;
                 categorias.put( categoria, no++ );
                docs.put(txt, categoria);
                }
         }//end for
      }
}//end for
```

Posteriormente se proceden a crear un dataset en memoria con dos atributos: clase y contenido del documento.

```
private Instances crearDataset_NO_Procesado() {
          // Crear lista de clases
    FastVector  clases = crear_categorias();
    // Crear lista de atributos (cabecera del dataset)
    FastVector atributos = new FastVector(2);
    atributos.addElement(new Attribute("CATEGORIA", clases));
    atributos.addElement(new Attribute("CONTENIDO", (FastVector) null));
    dataset_No_process = new Instances("Dataset_NO_Procesado", atributos, docs.size());
    dataset_No_process.setClass(dataset_No_process.attribute("CATEGORIA")); // Otra forma
    equivalente: dataset.setClassIndex(0);
    Iterator it = docs.entrySet().iterator();
          while (it.hasNext()) {
               Map.Entry doc = (Map.Entry)it.next();
             Instance instancia = crearInstancia( (File) doc.getKey(), (String )doc.getValue() );
            dataset_No_process.add(instancia);
                File nf = (File) doc.getKey();
            //System.out.println( "Doc ["+nf.getName() + "] =>" + doc.getValue());
          }
    return (dataset_No_process);
}
```

Una vez creado el dataset, se procede a aplicar filtros sobre los datos. Específicamente se aplica el filtro StringToWordVector. Esto permite que el clasificador pueda realizar su trabajo.

```
private StringToWordVector ConfFiltro_StringToWordVector (Instances dataset) throws Exception
{
      StringToWordVector filtroTemp = new StringToWordVector();
      // Establecer el formato del dataset de entrada
```

```
        filtroTemp.setInputFormat(dataset);
        // Indica que atributos procesar
        int[] idxAtributosFiltrar = {1};
        filtroTemp.setAttributeIndicesArray(idxAtributosFiltrar); // Otra forma equivalente:
        filtroTemp.setSelectedRange("2");
        return (filtroTemp);
}
```

Con el dataset procesado, se continúa con la construcción de un modelo de clasificación. Este modelo se guarda en memoria para posteriormente evaluar el clasificador con el corpus de prueba.

```
clasificador = new LibSVM();
//clasificador.setSVMType(new SelectedTag(LibSVM.SVMTYPE_ONE_CLASS_SVM,
LibSVM.TAGS_SVMTYPE));
clasificador.buildClassifier(DataSetProcess);
```

Para evaluar el modelo generado se procede de la misma manera. Se escoge el directorio donde está el corpus de pruebas se crea un dataset con dos atributos: Categoría en donde se configura el valor y el contenido del documento.

```
categorys_eval = new HashMap<Object,Object>();
doc_test = new HashMap<DocClasificado,Object>();
int nodocs =0;
   for (File fichero : listaFicheros) {
      if( fichero.isDirectory()  ){
        categoria = fichero.getName();
        categorys_eval.put( categoria, 0 );
        File[] txts = fichero.listFiles();
        int no = 1;
       for(File txt: txts){
          if( txt.getName().contains(".txt") ){
             nodocs ++;
             categorys_eval.put( categoria, no++ );
             DocClasificado docts = new DocClasificado( txt, categoria, "??" );
              doc_test.put(docts, categoria);
            }
       }//end for
  }
}//end for
LearningData ledata = new LearningData();
ledata.setDoc_test( Clasifica(doc_test) );
```

Las instancias del dataSet son sometidas al filtro StringTowordVector y luego pasan a ser evaluadas con el modelo, lo que se obtiene de este proceso es la clase predicha por el clasificador.

```
DocClasificado txt = (DocClasificado) key;
File arch     = txt.getArchivo();
String OriginalClase = txt.getOriginalClase();
String PredictedClase = txt.getPredictedClase();
instanciaInicial = crearInstancia(arch, OriginalClase);
filtro_StringToWordVector.input(instanciaInicial);
instanciaFiltrada = filtro_StringToWordVector.output();
idxClasePredicha = (int) clasificador.classifyInstance(instanciaFiltrada);
System.out.println( clasificador.getRevision());
clasePredicha = DataSetProcess.attribute("CATEGORIA").value(idxClasePredicha);
```

3.5 Fase de evaluación.

Para finalizar el proceso de desarrollo del esquema de clasificación de noticias, basado en NERC, es importante evaluar los resultados obtenidos para las dos técnicas aplicadas que son Procesamiento de Lenguaje Natural y Aprendizaje Automático; por lo cual se realizó un análisis de comparación de los resultados obtenidos para las dos técnicas, y concluir cuál de las dos técnicas es más eficiente en cuanto a la clasificación de noticias por categoría.

3.5.1 Métricas de evaluación.

La precisión y recall son métricas empleadas en la medida del rendimiento de los sistemas de búsqueda, recuperación de información y reconocimiento de patrones. En el caso de esta investigación nos enmarcamos en el último tema; ya que el objetivo es la clasificación de documentos mediante el reconocimiento d patrones. Se denomina precisión a la fracción de instancias clasificadas que son relevantes, mientras que recall (sensibilidad o exhaustividad), es la fracción de instancias relevantes que han sido clasificadas. Tanto la precisión como el recall son entendidas como medidas de la relevancia.

3.5.1.1 Precisión.

El concepto de precisión fue definido por primera vez por Ken (keen, 1995) el cual lo define como un factor de pertinencia, otros autores se refieren a este concepto como “ratio de aceptación”, entre todas las definiciones atribuidas a la precisión. (Salton, 1983) la define como “la proporción de material recuperado realmente relevante, del total de los documentos recuperados”, esta definición es ampliamente utilizada dentro de la RI. (Frakes, 1992) complementa que el valor de la relevancia está entre cero y uno, siendo la “recuperación perfecta” aquella que tenga un valor de uno, es decir todos los documentos relevantes han sido recuperados. La fórmula utilizada por Salton y empleada en el presente trabajo de investigación es la siguiente:

$$\textbf{\textit{Precisión}} = \frac{\textit{Documentos relevantes recuperados}}{\textit{Documentos recuperados}}$$

Para el caso de las pruebas en esta investigación hemos adecuado los términos para una mejor comprensión; por tanto la fórmula quedaría de la siguiente manera:

$$\textbf{\textit{Precisión}} = \frac{\textit{Verdaderos Positivos}}{\textit{Falsos positivos}}$$

Sin embargo, la medida de precisión está sujeta a dos problemas: el ruido documental o falsos positivos que no son más que aquellos documentos que se clasifican como relevantes

cuando en realidad no lo son, y el silencio informativo que en cambio son documentos relevantes que no han podido ser clasificados por diferentes circunstancias.

3.5.1.2 Recall.

El recall también es una métrica utilizada en Recuperación Información, este concepto busca la proporción del material relevante recuperado del total de los documentos que son relevantes en la muestra analizada sin importar si estos han sido recuperados por el sistema. Nuevamente se recurre a (Salton, 1983) para describir la forma de cálculo de este concepto:

$$\textbf{\textit{Recall}} = \frac{\textit{Documentos relevantes recuperados}}{\textit{Documentos relevantes}}$$

Para el caso de las pruebas en esta investigación hemos adecuado los términos para una mejor comprensión; por tanto la fórmula quedaría de la siguiente manera:

$$\textbf{\textit{Recall}} = \frac{\textit{Verdaderos Positivos}}{\textit{Falsos Negativos}}$$

Si el valor del cálculo de la retentiva es igual a uno, se tiene una exhaustividad máxima ya que se ha encontrado todo lo relevante que había en la muestra, por tanto el ruido y silencio informativo serán nulos y se ha procedido a obtener una “recuperación perfecta”.

3.5.2 Resultados de la pruebas.

Pruebas para la técnica de Procesamiento de Lenguaje Natural

Los resultados de precisión que se presentan para el caso de la técnica de PLN son:

Tabla.13: Resultados prueba de precisión PLN.

Categorías	Verdaderos Positivos	Falso Positivo	Precisión
Académica	168	174	97%
Social-Cultural	152	168	90%
Emprendimiento	62	71	87%
Investigativa	42	46	91%
Totales	**424**	**459**	**91%**

Elaboración: Las Autoras

Según la ecuación, se muestra los resultados para cada categoría, en el siguiente cuadro:

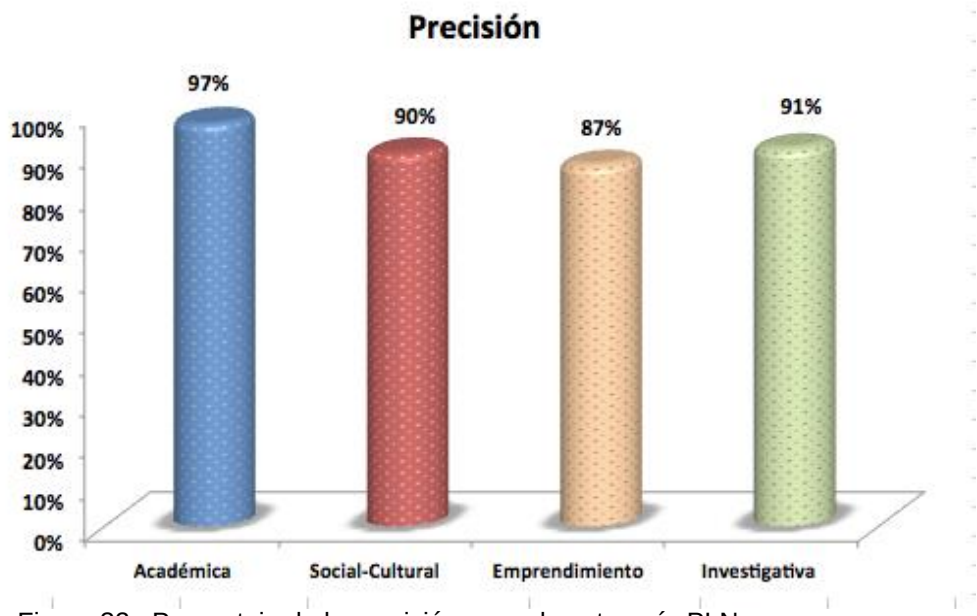

Figura 32. Porcentaje de la precisión en cada categoría PLN.
Elaboración: Las Autoras.

Análisis e interpretación:

Según las pruebas de precisión general, realizadas al esquema de clasificación de noticias mediante la técnica de Procesamiento de Lenguaje Natural es del 91%, esto significa que es un porcentaje aceptable de clasificación de noticias a la categoría que pertenece.

Si en la técnica de PLN, todos las noticias son clasificadas en la categoría correcta, se interpreta que la técnica aplicada es completamente precisa; pero esto no se puede dar debido a que existen falsos positivos y ruido documental, que es cuando se clasifican noticias en una categoría, cuando en realidad pertenecen a otra categoría, también se da el caso del silencio informativo donde una noticia no es clasificada en ninguna categoría, esto se puede dar por ejemplo cuando se tiene muy poca información, y no se puede formar un patrón de ninguna categoría de noticia.

Los resultados de recall que se presentan para el caso de la técnica de PLN son:

Tabla.14: Resultados prueba de recall PLN.

Categorías	Verdaderos Positivos	Falsos negativos	Recall
Académica	168	171	98%
Social-Cultural	152	164	93%
Emprendimiento	62	67	93%
Investigativa	42	43	98%
Totales	**424**	**445**	**95%**

Elaboración: Las Autoras

Según la ecuación, se muestra los resultados para cada categoría, en el siguiente cuadro:

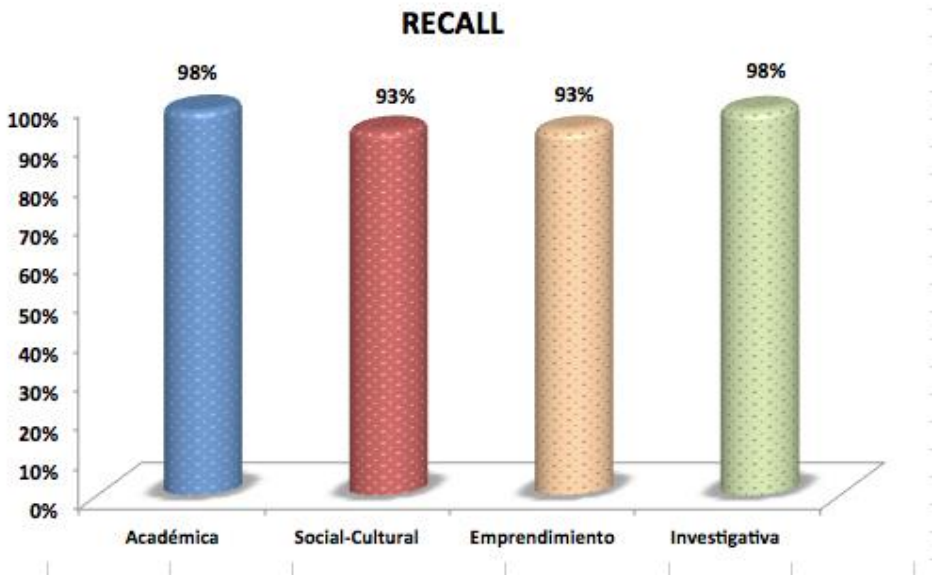

Figura 33. Porcentaje de recall en cada categoría PLN.
Elaboración: Las Autoras

Análisis e interpretación:

Según las pruebas de recall general, realizadas al esquema de clasificación de noticias mediante la técnica de Procesamiento de Lenguaje Natural es del 95%, esto significa que es un porcentaje aceptable de clasificación de noticias a la categoría que pertenece.

Si en la técnica de PLN, el resultado fuera 1, se tendría un recall máximo posible, y esto indicaría que se ha clasificado todas las noticias en la categoría que corresponden, por lo tanto no se tendría ni ruido, ni silencio informativo: siendo que la clasificación estaría perfecta; sin embargo en este caso si existe ruido y silencio informativo, por lo que no todas las noticias han sido clasificadas en la categoría que pertenecen.

Pruebas para la técnica de Aprendizaje Automático

Los resultados de precisión que se presentan para el caso de la técnica de AA son:

Tabla.15: Resultados prueba de precisión AA.

Categorías	Verdaderos Positivos	Falso Positivo	Precisión
Académica	138	159	87%
Social-Cultural	135	179	75%
Emprendimiento	58	86	67%
Investigativa	24	35	69%
Total	**355**	**459**	**75%**

Elaboración: Las Autoras

Según la ecuación, se muestra los resultados para cada categoría, en el siguiente cuadro:

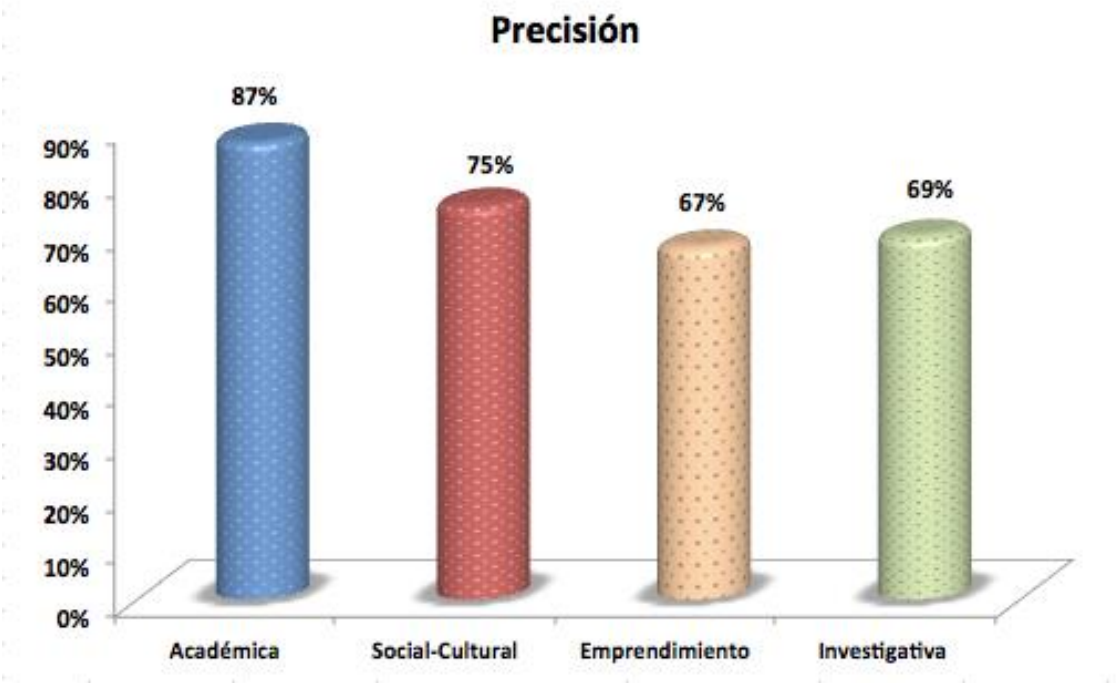

Figura 34. Porcentaje de recall en cada categoría PLN.
Elaboración: Las Autoras.

Análisis e interpretación:

Según las pruebas de precisión general, realizadas al esquema de clasificación de noticias mediante la técnica de Aprendizaje Automático es del 75%, esto significa que es un porcentaje aceptable de clasificación de noticias a la categoría que pertenece.

Si en la técnica de AA, todos las noticias son clasificadas en la categoría correcta, se interpreta que la técnica aplicada es completamente precisa; pero esto no se puede dar debido a que existen falsos positivos y ruido documental, que es cuando se clasifican noticias en una categoría, cuando en realidad pertenecen a otra categoría, también se da el caso del silencio informativo donde una noticia no es clasificada en ninguna categoría, esto se puede dar por ejemplo cuando se tiene muy poca información, y no se puede formar un patrón de ninguna categoría de noticia.

Los resultados de recall que se presentan para el caso de la técnica de AA son:

Tabla.16: Resultados prueba de recall AA

Categorías	Verdaderos Positivos	Falsos negativos	Recall
Académica	138	156	88%
Social-Cultural	135	179	75%
Emprendimiento	58	86	67%
Investigativa	24	34	71%
Total	**355**	**455**	**75%**

Elaboración: Las Autoras

Según la ecuación, se muestra los resultados para cada categoría, en el siguiente cuadro:

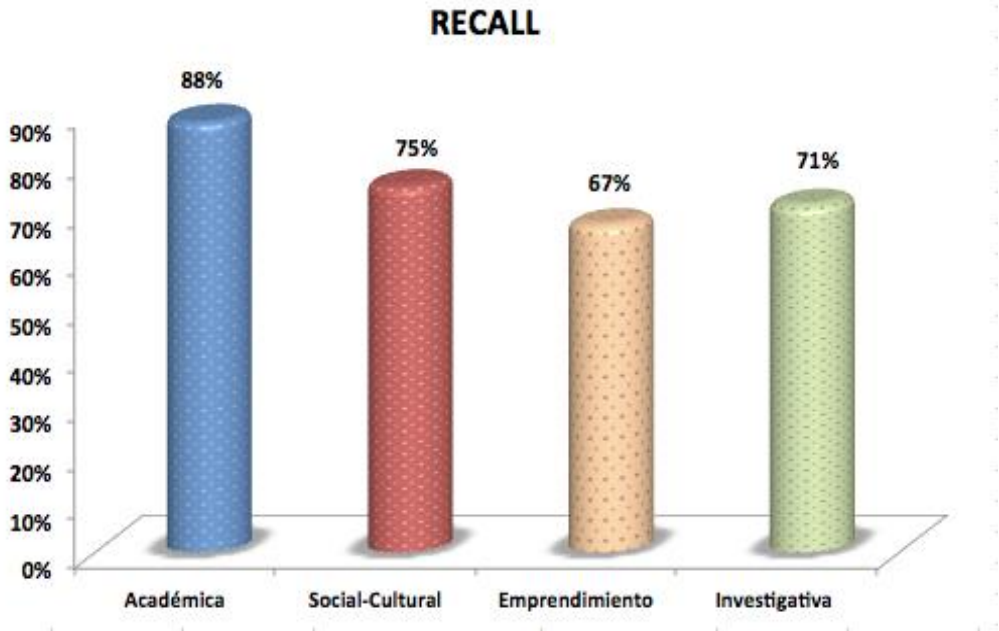

Figura 35. Porcentaje de recall en cada categoría con AA.
Elaboración: Las Autoras

Análisis e interpretación:

Según las pruebas de recall general, realizadas al esquema de clasificación de noticias mediante la técnica de Aprendizaje Automático es del 75%, esto significa que es un porcentaje aceptable de clasificación de noticias a la categoría que pertenece.

Si en la técnica de AA, el resultado fuera 1, se tendría un recall máximo posible, y esto indicaría que se ha clasificado todas las noticias en la categoría que corresponden, por lo tanto no se tendría ni ruido, ni silencio informativo: siendo que la clasificación estaría perfecta; sin embargo en este caso si existe ruido y silencio informativo, por lo que no todas las noticias han sido clasificadas en la categoría que pertenecen.

3.5.3 Cuadro comparativo de la precisión entre Procesamiento de Lenguaje Natural y Aprendizaje Automático.

Tabla.17: Precisión en cada categoría con PLN y AA.

Categorías	PLN	AA
Académica	97%	87%
Social-Cultural	90%	75%
Emprendimiento	87%	67%
Investigativa	91%	69%
Promedio Total	**91%**	**75%**

Elaboración: Las Autoras

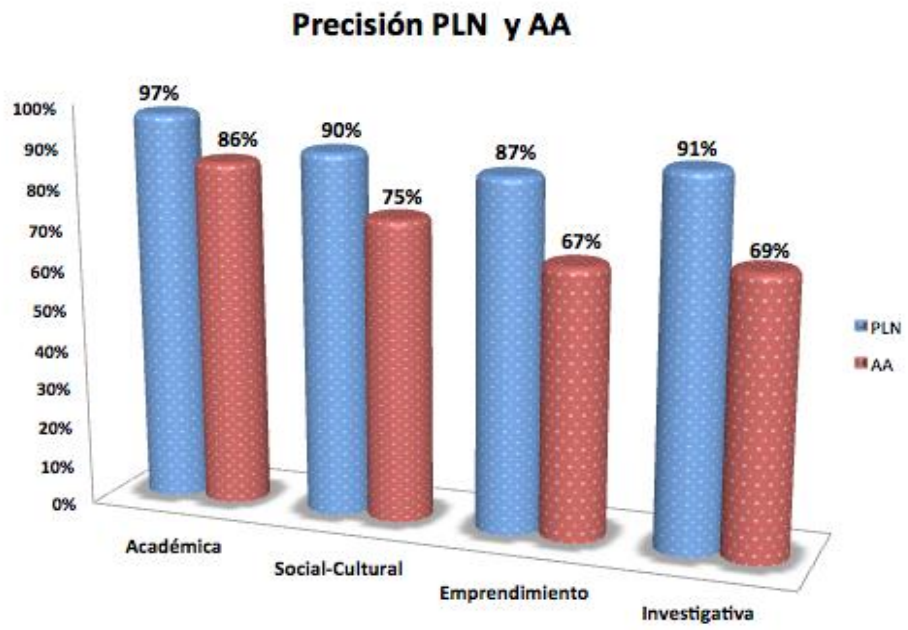

Figura 36. Porcentaje de precisión en cada categoría con PLN y AA.
Elaboración: Las Autoras

Análisis e interpretación:

Según las pruebas de precisión realizadas en las dos técnicas, se obtuvo mayor precisión en las noticias de categoría "Académica". Esto se debe a que existe más volumen de noticias de esta categoría; por tanto la generación de patrones o palabras claves es más alta. También se puede observar mayor precisión en la técnica de PLN, esto es porque la depuración que se realiza es manual y se puede controlar; pero así mismo se convierte en una desventaja porque si cambian la estructura organizacional el dominio con la que se trabaja en este caso la UTPL, bajaría la precisión. Con respecto a la técnica AA, la precisión es menor; sin embargo es aceptable su porcentaje, en esta técnica se aplica aprendizaje supervisado tal es el caso que si también existen cambios estructurales dentro del dominio, se tendría problemas al momento de clasificar.

3.5.4 Cuadro comparativo del recall entre Procesamiento de Lenguaje Natural y Aprendizaje Automático.

Tabla.18: Recall en cada categoría con PLN y AA.

Categorías	PLN	AA
Académica	98%	88%
Social-Cultural	93%	75%
Emprendimiento	93%	67%
Investigativa	98%	71%
Promedio Total	**95%**	**75%**

Elaboración: Las Autoras

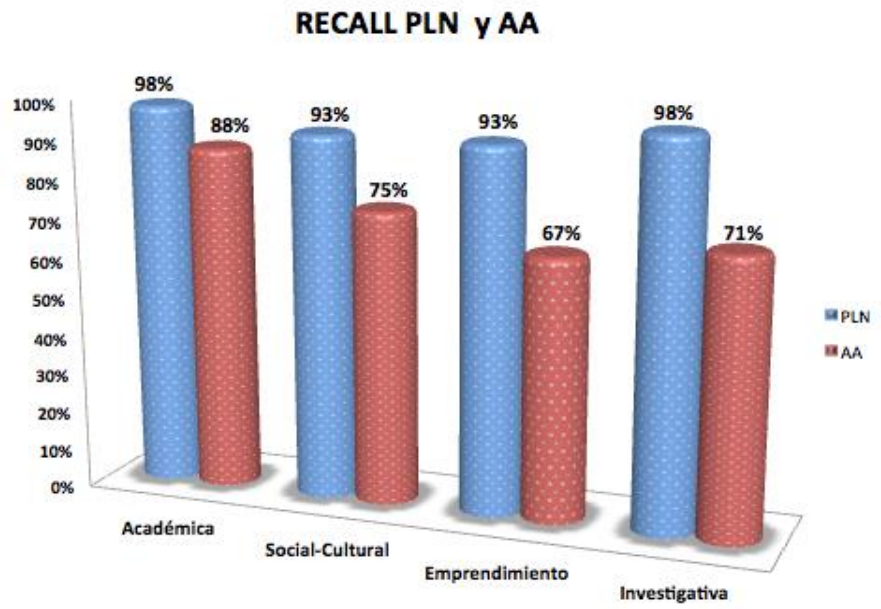

Figura 37. Porcentaje de recall en cada categoría con PLN y AA.
Elaboración: Las Autoras

Análisis e interpretación:

Según las pruebas de recall realizadas en las dos técnicas, se obtuvo mayor precisión en las noticias de categoría “Académica” e “Investigativa”. Esto se debe a que existe más volumen de noticias de estas dos categorías; por tanto la generación de patrones o palabras claves es más alta. También se puede observar mayor recall en la técnica de PLN, esto es porque la depuración que se realiza es manual y se puede controlar; pero así mismo se convierte en una desventaja porque si cambian la estructura organizacional el dominio con la que se trabaja en este caso la UTPL, bajaría el recall. Con respecto a la técnica AA, el recall es menor; sin embargo es aceptable su porcentaje, en esta técnica se aplica aprendizaje supervisado tal es el caso que si también existen cambios estructurales dentro del dominio, se tendría problemas al momento de clasificar.

CAPÍTULO IV

CONCLUSIONES Y RECOMENDACIONES

CONCLUSIONES

1. El aplicar un sistema supervisado tiene un alto costo de ingeniería, porque se tiene que estar actualizando la información, por ejemplo se tiene que generar nuevas reglas en el caso de tener nueva información en el dominio. Y si se quiere aplicar a otro ámbito de información se tiene que empezar de nuevo todo el proceso.

2. Si es factible aplicar las técnicas NERC dentro de las metodologías de Procesamiento de Lenguaje Natural y Aprendizaje Automático para un proceso de clasificación de documentos no estructurados en el contexto universitario.

3. La clasificación mediante la metodología de PLN da mejores resultados al obtener mayor precisión y recall, sin embargo tiene una desventaja de que la mayoría del proceso es manual y su costo es mayor; mientras que la metodología de AA a pesar que tiene menor precisión y recall es aceptable, tiene menor costo y es más adaptable a cambios futuros

4. El esquema de clasificación propuesto presenta una precisión de 91% en la metodología de PLN y una precisión de 75% en la metodología AA. En el caso de PLN es mayor la precisión debido a que se puede controlar manualmente la generación de patrones o palabras claves; sin embargo se consideraría como una desventaja. En el caso de AA, el algoritmo de clasificación SVM, se encarga de generar los patrones sin embargo su precisión es aceptable. El esquema de clasificación propuesto presenta un recall de 95% en la metodología de PLN, y un recall de 75% en la metodología AA. Existen noticias que no se etiquetan en ninguna de las cuatro categorías determinadas; esto se debe al contexto de la noticia, en ocasiones en realidad no pertenece a ninguna categoría.

RECOMENDACIONES

1. Se recomienda implementar a futuro el esquema de clasificación propuesto en esta investigación, dentro del entorno Web de la página oficial de la UTPL.

2. El esquema de clasificación podría ser adaptado a otro tipo de información universitaria como por ejemplo con información de competencias universitarias.

3. Se sugiere utilizar un aprendizaje semi-supervisado para mejorar los costos y operatividad del sistema, con el fin de cubrir los futuros cambios del dominio donde se esté trabajando. y no se tenga que reestructurar toda las reglas cada que vez que se reestructure su operatividad

4. Se sugiere integrar en un proyecto futuro las dos técnicas PLN y AA, para mejorar la clasificación de contexto y así mejorar costo y tiempo de investigación, para cumplir los objetivos propuestos.

BIBLIOGRAFÍA

1. Allen, J. (1995). *Natural Language Understanding". Redwood City.* Benjamin/Cummings.

2. Balbontín, Á. y Sánchez, J. (2002). *SPNER – Reconocedor de entidades nombradas para el español.*

3. Calunya, U. P. (Noviembre de 2010). *Freeling Home*. Obtenido de http://nlp.lsi.upc.edu/freeling/

4. Candel Contardo D. (2011) "Algoritmo tipo SMO para la AD-SVM aplicado a clasificación multicategoría" Universidad Técnica Federico Santa María Departamento de Informática Valpara íso – Chile

5. Ceccaroni, Luigi. (2009). Inteligencial Artificial Adquisición automática del conocimiento.

6. Christofer, D., Manning y Hinrich, S. (1999). Foundations of statistical Natural Language Processing,Second Printing. The MIT Press Cambridge. Massachusetts.

7. Covington, M. A. (2002). "Natural Language Processing for Prolog Programmers". Artificial Intelligence Programs The University of Georgia Athens, Georgia. NewJersey 07632: Artificial Intelligence Programs The University of Georgia Athens.

8. Croft, W., y Lewis, D. (1987). An approach to natural language processing for document retrieval, in Proceedings ACM SIGIR International. New Orleans.

9. Di Deco, J. (2012). Estudio y aplicación de técnicas de aprendizaje automático orientadas al ámbito médico. Madrid.

10. Elmasri, R., y Navathe, S. (1997). Fundamentals of Database Systems", Seconds Edition, Addison-Wesley Publishing Company. Massachusetts: Inc. Reading.

11. Fagan, J. (1987). Experiments in Automatic Phrase Indexing for Document Retrieval: A Comparison of Syntactic and Non-Syntactic Methods. Cornell University: Departamento de ciencias de la computacion.

12. Fernández, Iván (2009). Conocimiento en Línea. Lucene: Tutorial. Versión 0.2. Universidad Politécnica de Cataluña.

13. Garcia, M. y Gamayo, P. (2011). Resolución de Correferencia de Nombres de Persona para Extracción de Información Biográfica. España: Universidad Santiago de Compostela.

14. Gelbukh, A. y Galicia, H. (2007). Investigaciones en análisis sintactico para el español. (Primera Edisción ed.). Instituto Politecnico Nacional.

15. Grice, H. (1975). Logic and conversation; Syntax and semantics (Vol. 3). New York: Academic Press.

16. Grishman, R. (1986). Computational Linguistics: an introduction. Cambridge,: Cambridge University Press.

17. Hernández, P. (2004). Aplicación de árboles de decisión en modelos de riesgo crediticio. Revista colombiana de estadística, 27(2), 139-151.

18. Kotsiantis, P. (s.f.). Supervised machine learning: A review of classification techiquez (Vol. vol. 31). Informática.

19. Lewis, D. D. (1998). Naive (Bayes) at forty: The independence assumption in information retrieval. In Machine learning: ECML-98 (pp. 4-15). Springer Berlin Heidelberg.

20. Li Yaoyong, Bontcheva Kalina, Cunningham Hamish (Septiembre 01, 2008). "Adapting SVM for Data Sparseness and Imbalance: A Case Study on Information Extraction" Department of Computer Science, The University of Sheffield Regent Court, 211 Portobello, Sheffield S1 4DP, UK.

21. Liu, X., Zhang, S., Wei, F. y Zhou, M. (2011, June). Recognizing named entities in tweets. In Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1 (pp. 359-367). Association for Computational Linguistics.

22. Locke, W. y Booth , A. (1995). Machine Translation of Languages, Technology Press of MIT ans Wiley. Cambridge, Mass.

23. Manaris, B. y Slator, B. (1996). Interactive Natural Language Processing: Building on Success. Computer, IEEE.

24. Marrero, M., Sánchez, S., Lara, JM y Andreadakis, G. (2009).Evaluación de los sistemas de extracción de entidades nombradas. Avances en Lingüística Computacional, Investigación en Ciencias de la Computación ,41 , 47-58.

25. Martinez R., J. (2008). Sistema de Clustering de Named Entities. Universidad Politécnicas de Catulunya. Departamento LSI. Barcelona España. Volumen 1.

26. Molina, M. (2004). Desambiguación en procesamiento del lenguaje natural mediante técnicas de aprendizaje automático. Universidad Politécnica de Valencia, Universidad Politécnica de Valencia.

27. Moreiro González, J. A. (2002). Aplicaciones al análisis automático del contenido provenientes de la teoría matemática de la información.

28. Moreno, A. (1998). "Lingüística Computacional. Introducción a los modelos simbólicos, estadísticos y biológicos. Madrid: Editorial Sintesis.e

29. Muñoz, V. (Septiembre 01, 2008). Herramientas para la Extracción de Información bajo la arquitectura GATE. México.

30. Pazienza, M. (1997). Information Extraction: A Multidisciplinary Approach to an Emerging Information Technology. International Summer School, SCIE-97.

31. Ramírez, Y. (s.f.). Un enfoque híbrido al Reconocimiento de Nombres de Entidades para el español. La Habana, Cuba.

32. Sánchez, C. (2008). Clasificación de Entidades Nombradas. Tonantzintla, Puebla: INAOE.

33. Serra Sepúlveda, S. (2012). Gramática y diccionario: contornos, solidaridades léxicas y colocaciones en lexicografía española contemporánea.

34. Télles, A. (2005). Extracción de Información con Algoritmos de Clasificación. Tonantzintla, Pue.: INADE.

35. Tomás, D., Bisbal, E., Vicedo, J. L., Moreno, L. y Suárez, A. (2005). Una aproximación multilingüe a la clasificación de preguntas basada en aprendizaje automático. Procesamiento del Lenguaje Natural, 35, 391-400.

36. Torres J. y García, J. (2012).Clasificación de patrones con memorias asociativas bidireccionales ab (Doctoral dissertation).

37. Zafra, A. (2009). Modelos de Programación Genética Gramatical para Aprendizaje con Múltiples Instancias. Granada: Universidad de Granada. Departamento de Ciencias de la Computación e Inteligencia Artificial.

38. Zubiaga A. (2008), Aproximaciones a SVM semisupervisado multiclase para la clasificación de páginas web. UNED: Departamento de Lenguajes y Sistemas Informáticos ETS. De Ingeniería Informática.

ANEXOS

Anexo 1: Configuraciones para instalar GATE

Instalación del Programa Cygwin

Se procedió a instalar el programa Cygwin. El mismo que permite una gran variedad de formas para formar un path. El Cygwin se vale de una aplicación gráfica (el setup.exe) para añadir, quitar y actualizar paquetes. El setup.exe acepta parámetros de línea de comandos. Si se ejecuta setup-h, en el directorio donde este, se creará un fichero setup.log mostrándonos las opciones disponibles para una instalación desatendida.

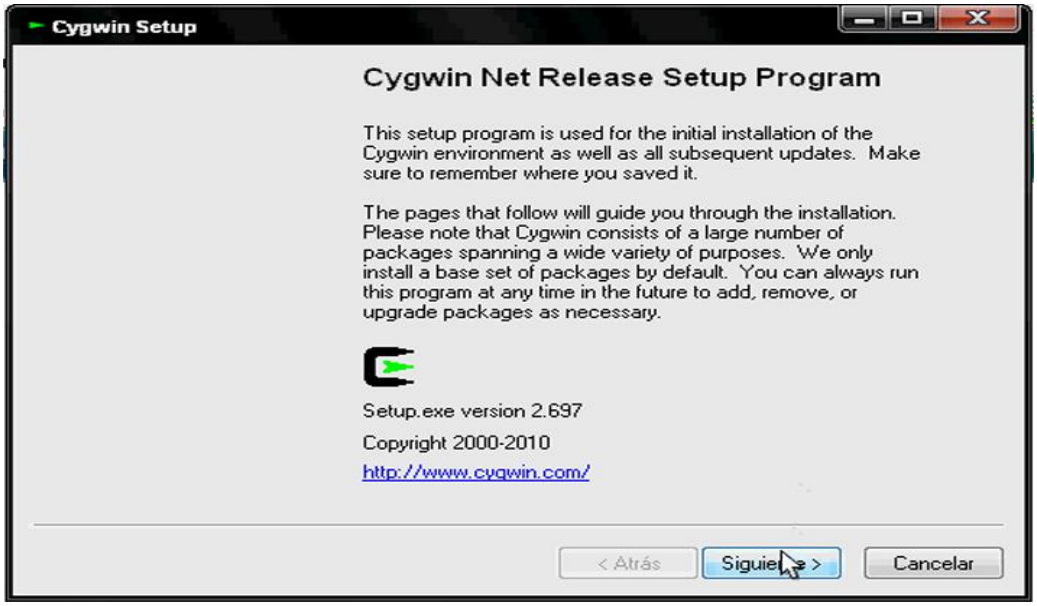

Figura 1. Instalando Cygwin.

Los paquetes que descarga el setup.exe en el *Local Package Directory* son unos ficheros .tar.bz2 que contienen los ficheros del paquete junto con unos scrips opcionales para /etc/postinstall y /etc/preremove que se ejecutan en el momento que su nombre indica.

```
2007/07/17 19:45:50 Starting cygwin install, version 2.573.2.2
2007/07/17 19:45:50 Current Directory: C:\Temp\Cygwin
2007/07/17 19:45:50
Command Line Options:
 -D --download                          Download from internet
 -L --local-install                     Install from local directory
 -s --site                              Download site
 -R --root                              Root installation directory
 -q --quiet-mode                        Unattended setup mode
 -h --help                              print help
 -l --local-package-dir                 Local package directory
 -r --no-replaceonreboot                Disable replacing in-use files on next
                                        reboot.
 -n --no-shortcuts                      Disable creation of desktop and start
                                        menu shortcuts
 -N --no-startmenu                      Disable creation of start menu shortcut
 -d --no-desktop                        Disable creation of desktop shortcut
 -A --disable-buggy-antivirus           Disable known or suspected buggy anti
                                        virus software packages during
                                        execution.
Ending cygwin install
```

Figura 2. Starting Cygwin.

Posterior a la instalación del mismo se configura las variables de entorno necesarias.

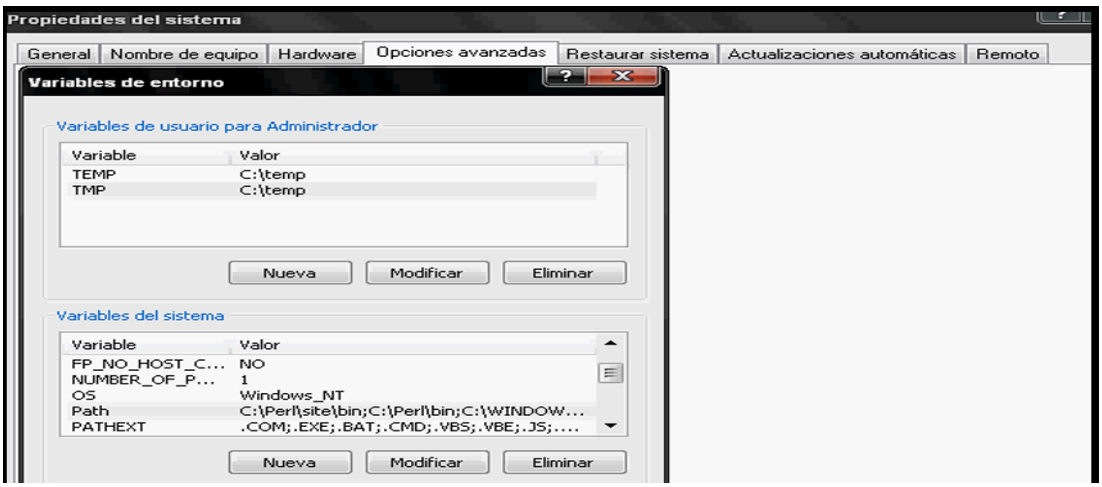

Figura 3. Configurando variables de entorno.

Instalación del TreeTagger

El TreeTagger es una herramienta para anotar textos con información de part-of-speech y lema, desarrollado dentro del proyecto TC en el Institute for Computational Linguistics of the University of Stuttgart. Ha sido utilizado con éxito para taggear textos en alemán, inglés, francés, italiano, español, griego, y francés antiguo, y es fácilmente adaptable a otros lenguajes si se dispone de un lexicon y corpus marcado manualmente.

Para la instalación del TreeTagger dentro de la URL: http://www.ims.uni-stuttgart.de/projekte/corplex/TreeTagger/ se tiene que ubicar en los archivos para Windows XP, una vez descargado el archivo *tree-tagger-windows-3.2.zip*, se lo descomprime y se copia la carpeta en el directorio C:\, seguidamente se descarga los *Parameter Files* para el idioma que se necesita, en este caso en Español.

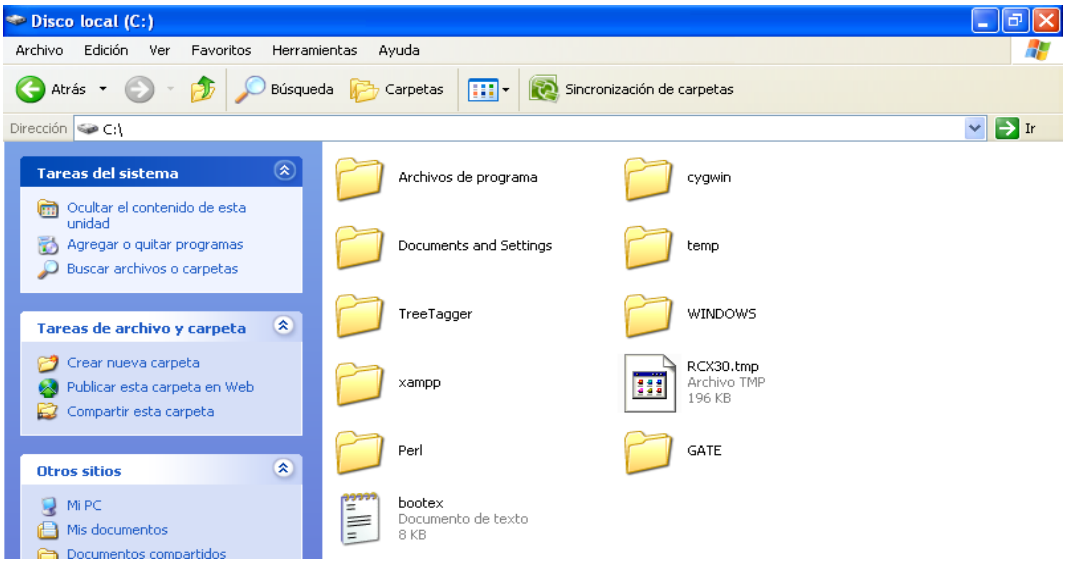

Figura 4. Instalando TreTagger.

Una vez descargados así mismo se descomprime y se los ubica dentro de la carpeta ***lib*** que se encuentra dentro de la carpeta de TreeTagger. Se debe tener en cuenta que la batería de etiquetado en español tiene saltos de línea que pueden ocasionar problemas, entonces se debe editarla. Para ello nos ubicamos en la carpeta de TreeTagger y dentro de la carpeta ***bin*** nos ubicamos en el documento en español se realiza click derecho y editar, y se borra los espacios entre los perl; esto sólo en el caso de que sea necesario.

Para poder probar la correcta instalación de TreeTagger es necesario instalar ***ActivePerl,*** se accede a la URL: http://aspn.activestate.com/ASPN/Downloads/ActivePerl/ , se elige para windowds XP, se procede a descargar y ejecutar, tomando en cuenta que durante el proceso de instalación debe estar la Location: C:\Perl\; esta instalación toma bastante tiempo por lo que se recomienda esperar el tiempo necesario.

Figura 5. Activando Perl.

Ahora para probar si la herramienta TreeTagger funciona correctamente, se va a la carpeta ***bin*** que se encuentra dentro de la carpeta TreeTagger y se crea un nuevo documento .txt dentro del cual escribiremos alguna frase en español. Posteriormente se abre la línea de comandos desde la ruta Inicio>Todos los programas>Accesorios>Símbolos del Sistema y se siguen los pasos como muestra la imagen. De esta manera se puede comprobar que si está funcionando el TreeTagger.

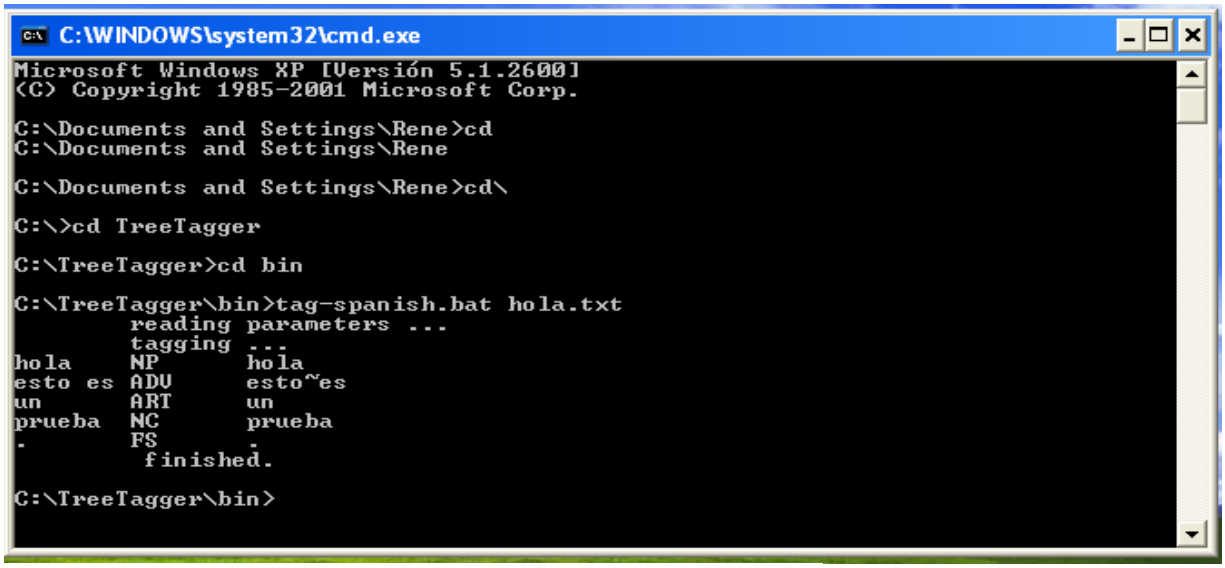

Figura 6. Prueba TreTagger.

Anexo 2: Instalación de GATE

La página oficial para descargar GATE es: http://gate.ac.uk/.

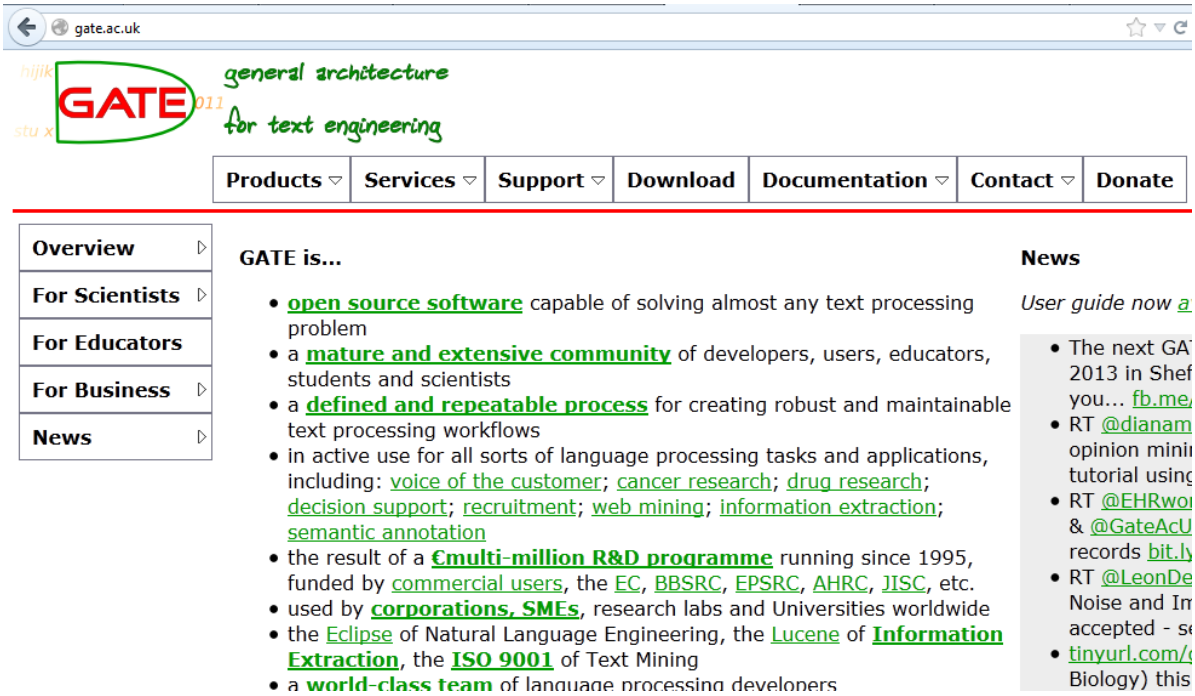

Figura 1. Página oficial de GATE.

La instalación del GATE es rápida y sencilla.

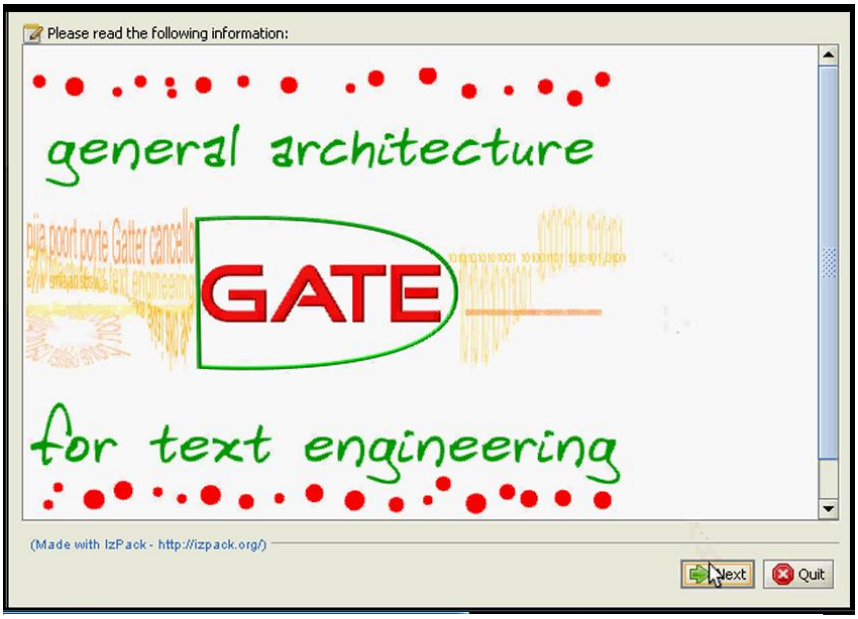

Figura 2. Instalando GATE Parte 1.

Es importante que durante la instalación se cambie el directorio y debe quedar así: C: \GATE

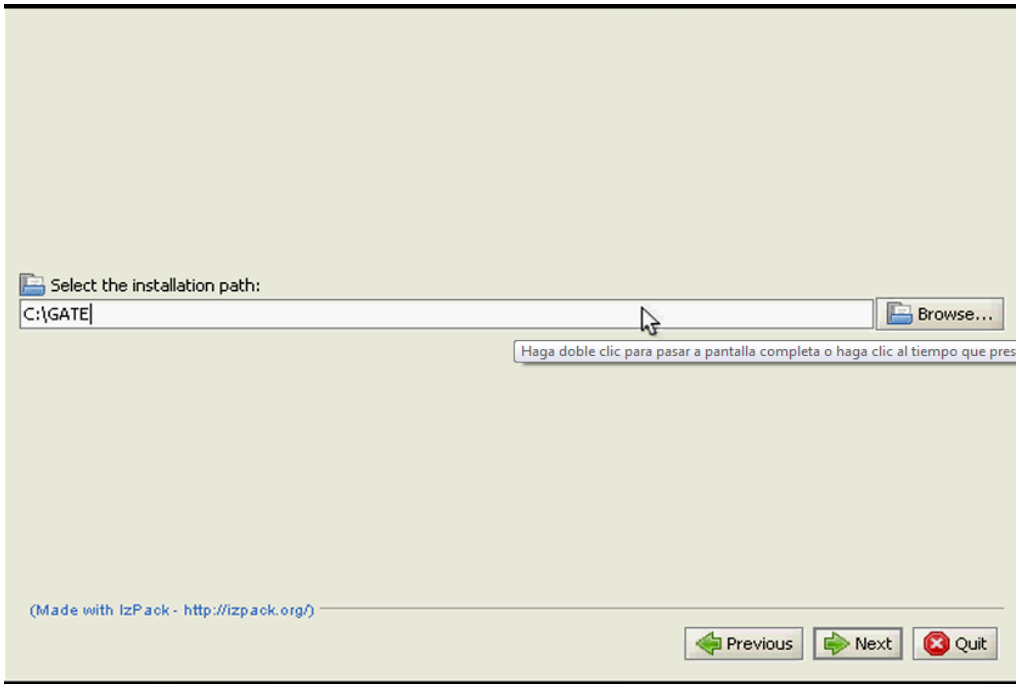

Figura 3. Instalando GATE Parte 2.

Los paquetes que se van elegir son los que muestra la imagen:

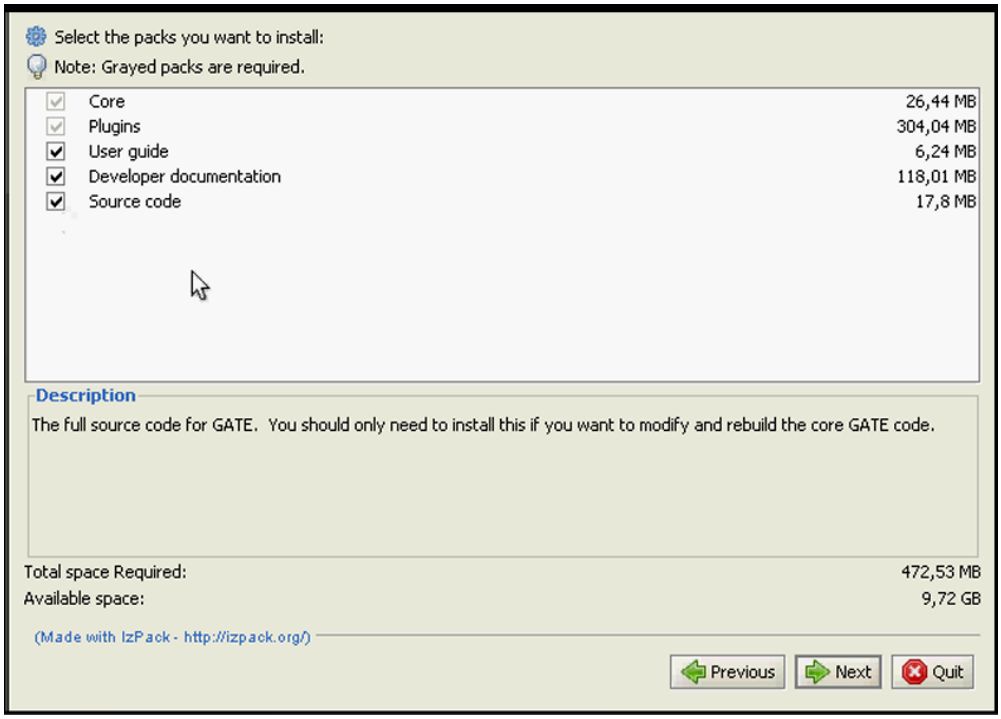

Figura 4. Instalando GATE Parte 3.

Una vez instalado GATE tenemos como muestra en la gráfica siguiente.

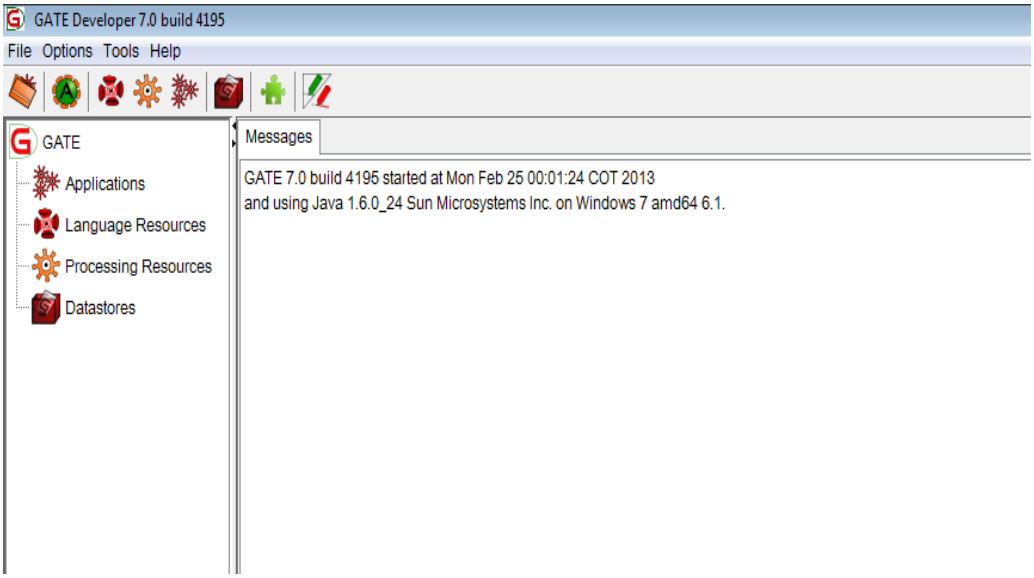

Figura 5. Instalando GATE Parte 4.

Integración de Treetagger con GATE

Para la integración se siguen los siguientes pasos:

1. Ingresamos a Manage CREOLE PLUGIS

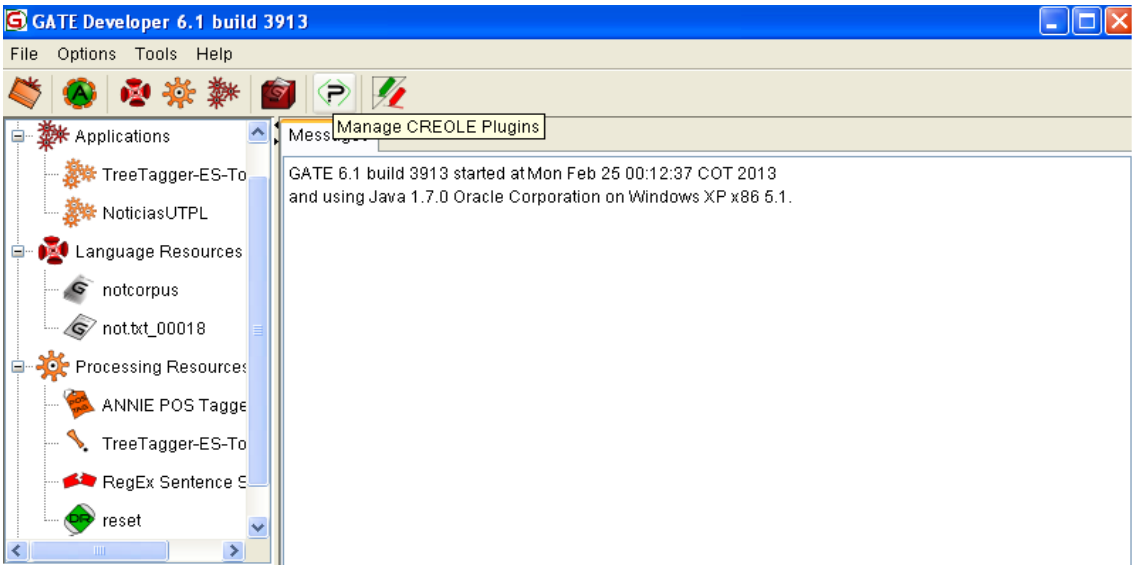

Figura 6. GATE y TreTagger.

Se elige los plugins a utilizar según se necesite en este caso elegimos:

ANNIE

Anotation Merging

Jape_Compiler

Machine_Learning

Tagger_Framework

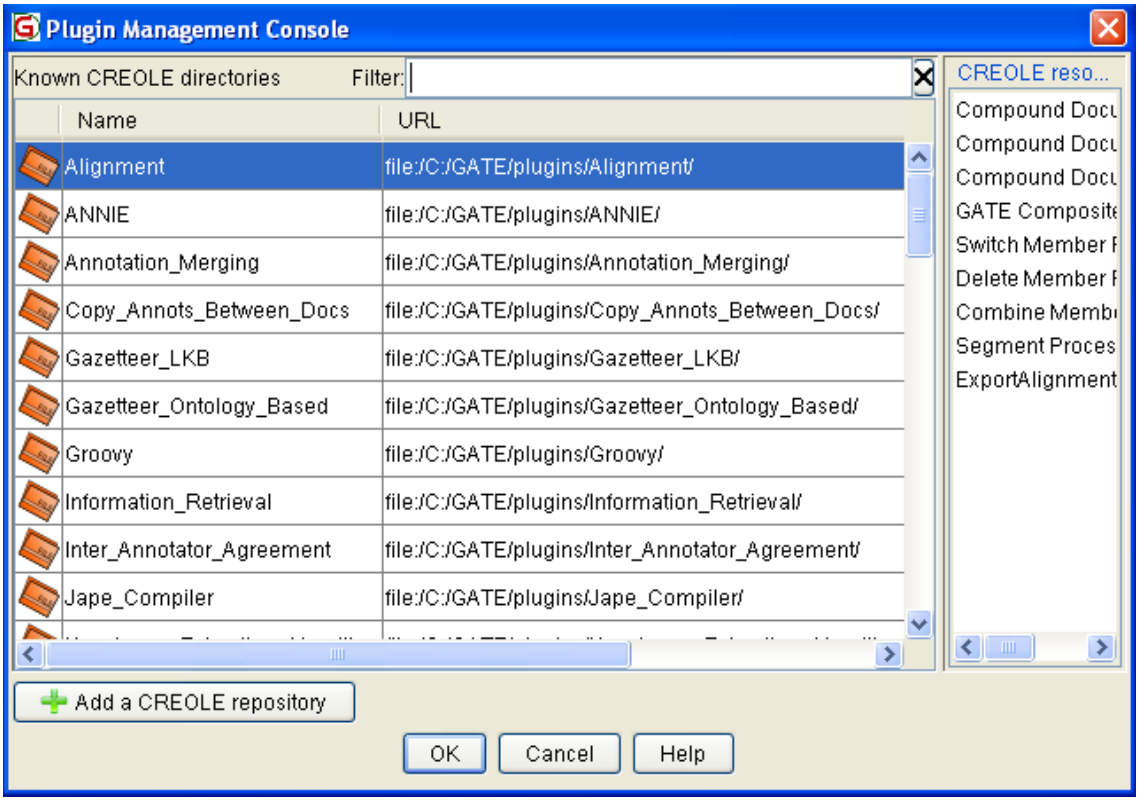

Figura 7. Adicionando Plugin ANNIE.

Luego se cierra el GATE y se vuelve a ejecutar. Y luego se procede a realizar las pruebas.

Anexo 3: Funcionamiento de Gazetteer

Para subir las listas de palabras etiquetadas, se tiene que poner en un archive txt, con extensión UTF-8. Como muestra la figura a continuación.

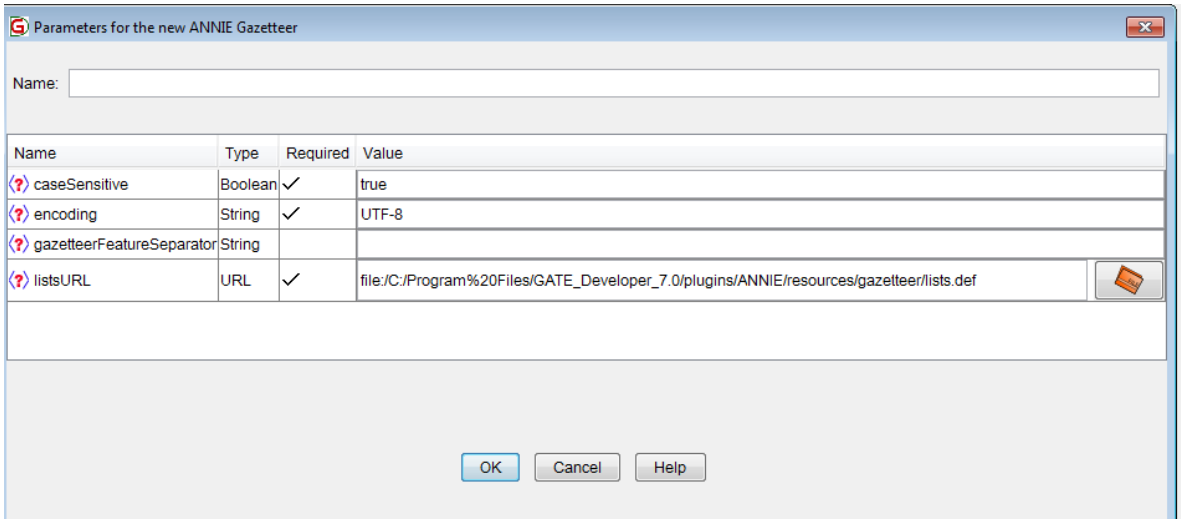

Figura 1. Configurando ANNIE Gazetteer.

Una vez subida el archivo, se pondrá observar la pantalla divida en dos partes. En la primera se visualiza el nombre de la lista. Y en la segunda los valores que contiene la misma.

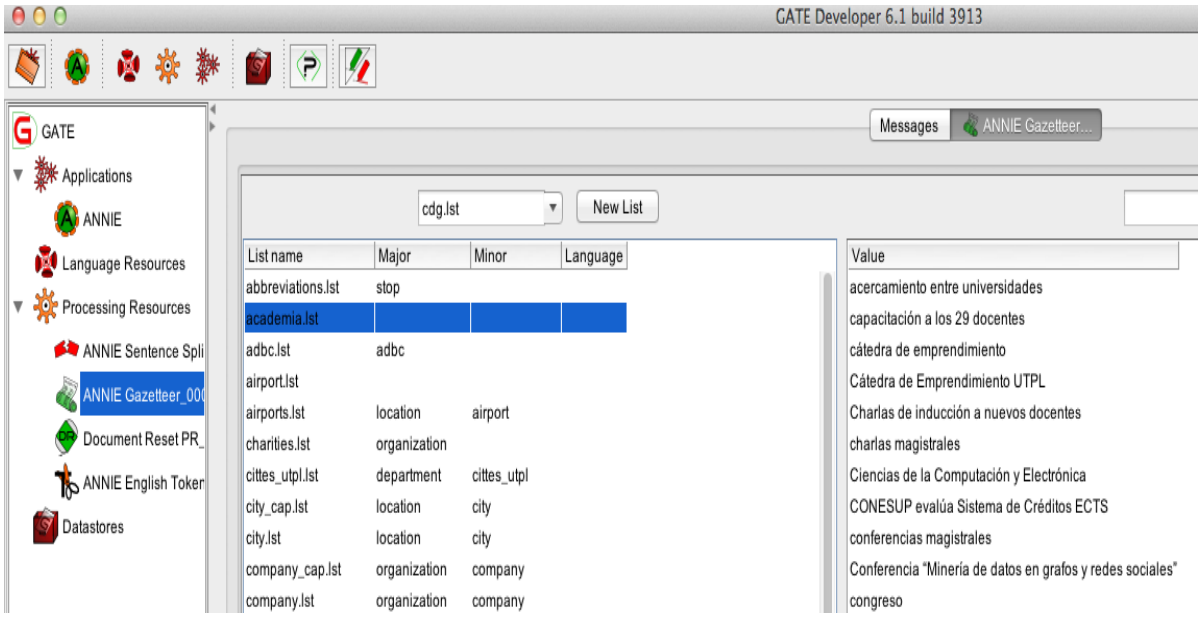

Figura 2: Visualización de ANNIE Gazetteer.

Esta lista se guarda, en la carpeta de GAZETTER, con extensión lst. Como se muestra a continuación.

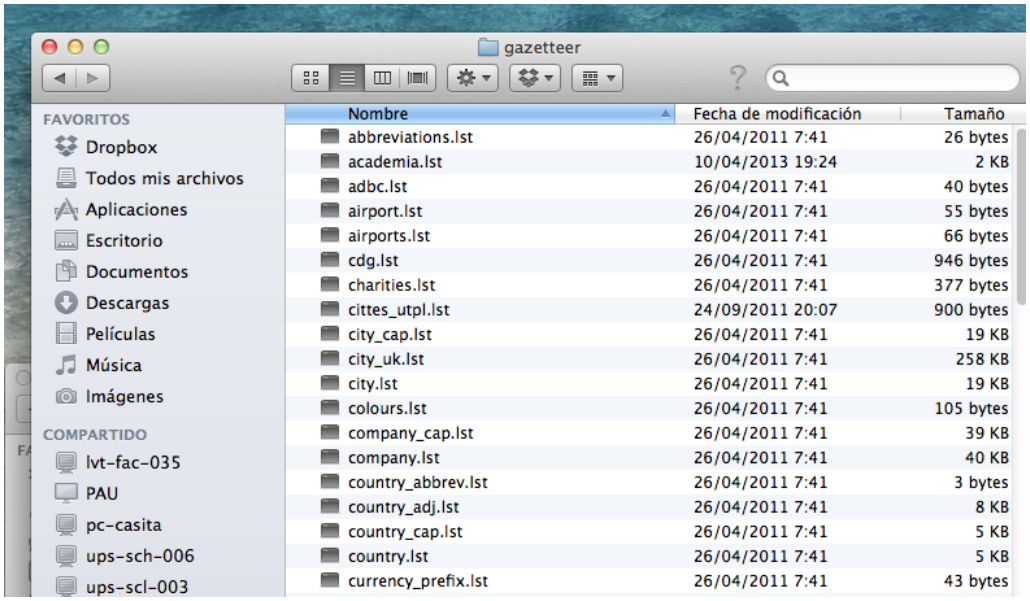

Figura 3. Vista de Gazetteers creados

A continuación se muestra el archivo, en extensión lst

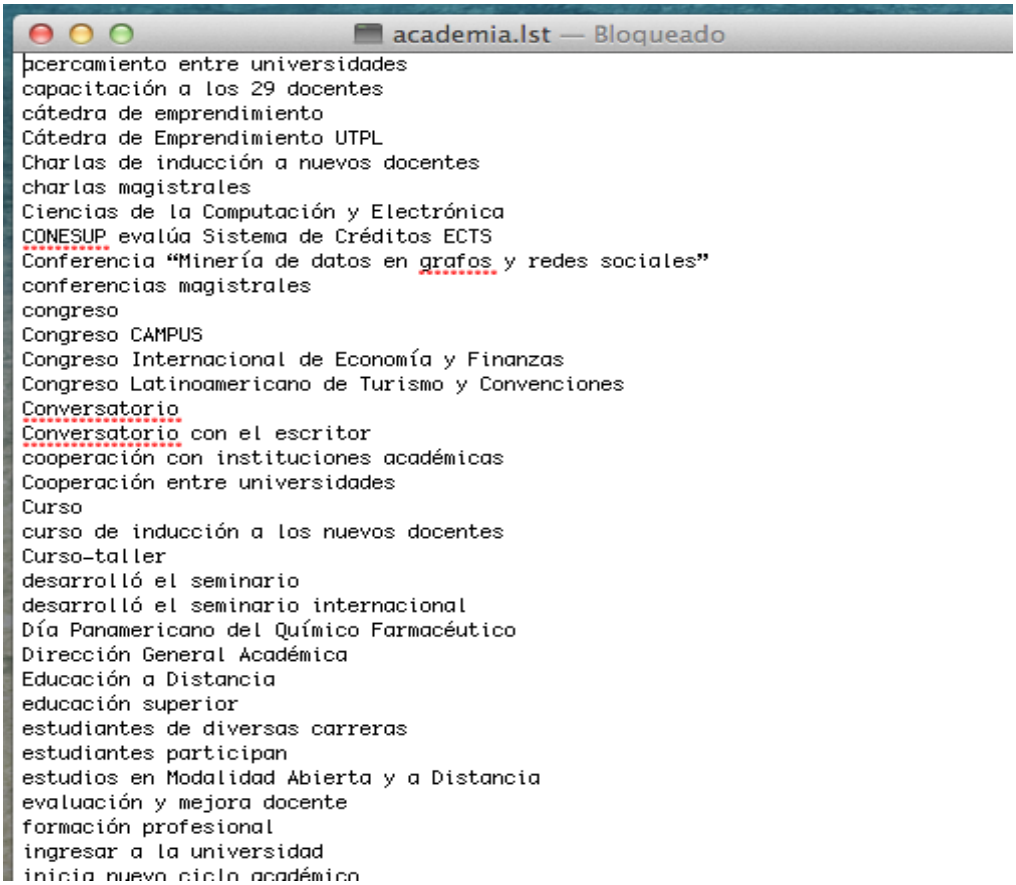

Figura 4. Ejemplo de Gazetter.

Anexo 4: Creación de las reglas JAPE

A continuación se muestra las reglas creadas para el corpus noticias UTPL

```
Phase:  noticas
Input:  Token Lookup Chunk
Options: control = appelt

Rule: academica
( (
        {Lookup.majorType == academica}
 )
 ({Token.string == "modalidad"}{Token.string == "abierta"}|{Token.string == "modalidadToken.string ==
"presencial"})
)
-->
:persona.responsable = {kind="Academica"}

Rule: socicial
(  (
        {Lookup.majorType == socialcultural}
 )
 ({Token.string == "aniversario"}|{Token.string == "rector"}|{Token.string == "festividades"}) // patrones
de texto.
)
-->
:persona.responsable = {kind="SocialCultural"}

Rule: gestion  //
(  (
        {Lookup.majorType == gestion}
 )
 ({Token.string == "participo"}|{Token.string == "presento"})
)
-->
:persona.responsable = {kind="gestion"}

Rule: investigacion
(  (
        {Lookup.majorType == investigacion}
 )
 ({Token.string == "centro"}{Token.string == "de"}{Token.string == "investigación"}|{Token.string ==
"investigación"})
)
-->
:persona.responsable = {kind="investigación"}

at gate.creole.gazetteer.LinearDefinition.load
```

Anexo 4: Metodología de PLN

En este anexo se detalla cómo se aplica la metodología de Procesamiento de Lenguaje Natural para la clasificación de noticias por categoría:

1. Se procede a descargar el instalador de: https://gate.ac.uk/ de acuerdo a las características del ordenador. La herramienta GATE es multiplataforma y existe versiones para Windows, MAC os, Linux.

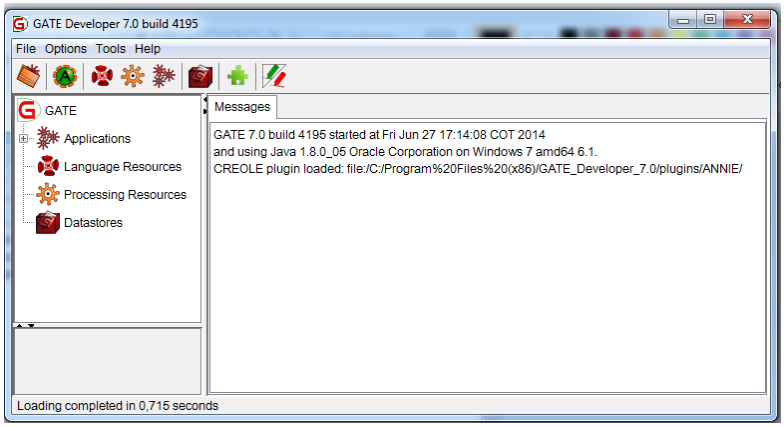

Figura.1 Ventana principal de GATE.

2. Una vez instalada la herramienta GATE, se procede a realizar la aplicación de PLN; donde el objetivo es una clasificación de noticias de la Universidad Técnica Particular de Loja, en cuatro categorías que son: Académica, Investigativa, Socio/Cultural y Emprendimiento.

2.1. Cargar el corpus de noticias a la herramienta GATE:

Antes de crear el corpus en la herramienta, manualmente se tiene en una carpeta 200 noticias en formato .txt, tomadas del portal de noticias de la UTPL, de diferentes fechas y con diferentes categorías. Considerando que las categorías asignadas son: Académica, Social/Cultural, Investigativa y Emprendimiento.

Ahora si ya se puede crear un corpus en GATE, para lo cual primero se crea un corpus vacío y luego se agrega los documentos o noticias. Como se muestra en la Figura 2, en la herramienta hay que ir a *Language Resourses —> click derecho New —> se elige GATE Corpus.*

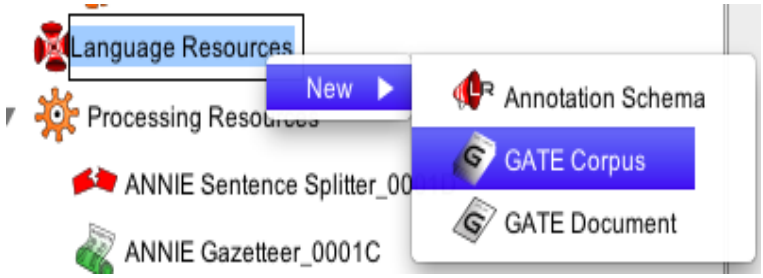

Figura.2 Crear un nuevo corpus.

Aquí aparece una ventana en donde se ubica el nombre de nuestro corpus, como indica la Figura 3, se finaliza dando click en ok.

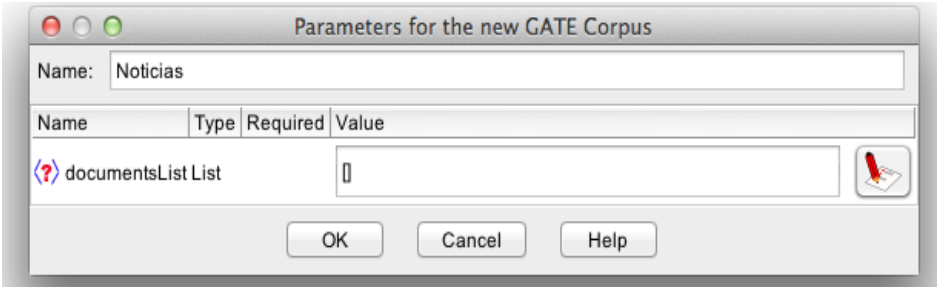

Figura.3 Asignar nombre a corpus.

Una vez creado el corpus vacío, se procede a cargarlo con archivos; en este caso con las noticias en formato .txt, previamente estos documentos deben estar guardadas en una carpeta. Al ubicarse sobre el corpus recién creado con nombre “noticias”, se realiza click derecho, se despliega una lista, de la misma se elige la opción *Populate* como indica la Figura.4

Figura.4 Proceso para cargar archivos al corpus.

Se abrirá una ventana en la cual se debe ubicar la dirección o URL donde se encuentren las noticias. Luego se da click en *Extensiones,* desplegándose una ventana donde permite teclear el tipo de formato de los documentos en este caso se teclea “txt” posteriormente a esto se da click en el botón *“Add”,* luego ok, y se regresa a la ventana anterior. Algo muy importante que se debe considerar es la codificación del documento, como se puede

observar el la figura 5, en la parte de “Encoding” se debe escribir UTF-8 o ISO-8859-1; si se deja en blanco GATE abrirá el documento con el enconding default del sistema operativo, finalmente se da click en ok. Ver Figura.5.

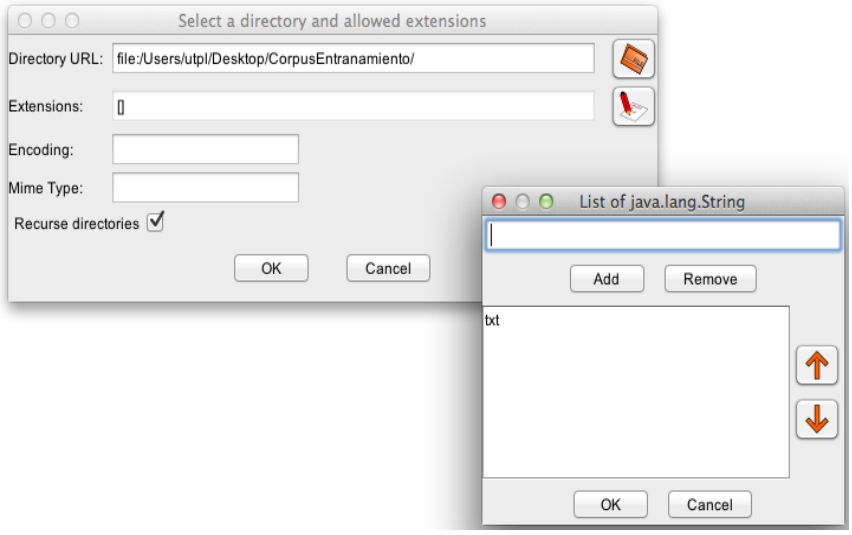

Figura 5. Cargando archivos al corpus.

Como se puede observar en la Figura 6, ya se tiene un corpus con 200 noticias, con el cual se va a proceder a realizar la etiquetación de Nombres de Entidades.

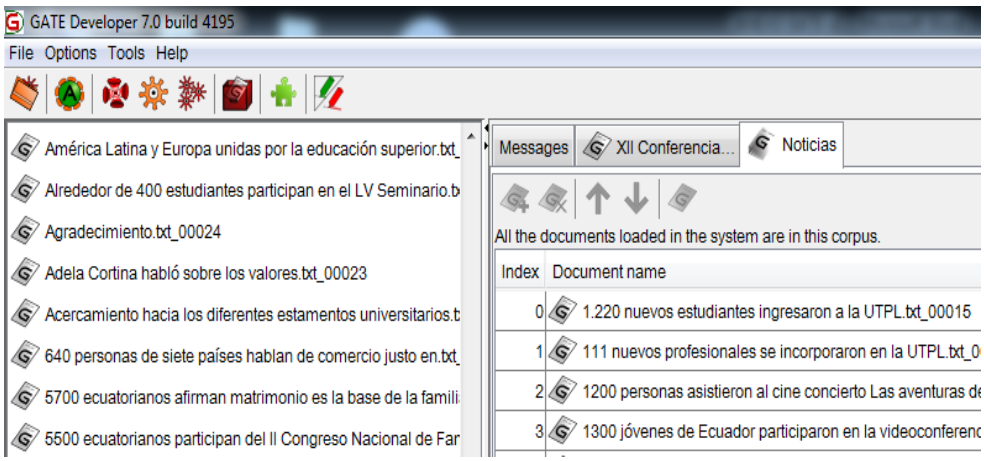

Figura 6. Corpus creado con 200 noticias.

2.2. Proceso de etiquetado manual mediante la herramienta GATE:

Una vez creado el corpus procedemos al etiquetado manual de las entidades, dentro de este proceso se tiene como entrada un documento en texto plano y como resultado se obtiene un documento estructurado en formato XML.

Se elige la noticia a etiquetarse, dando doble click sobre la misma, como muestra la Figura 7, se da click en *Annotation Sets*

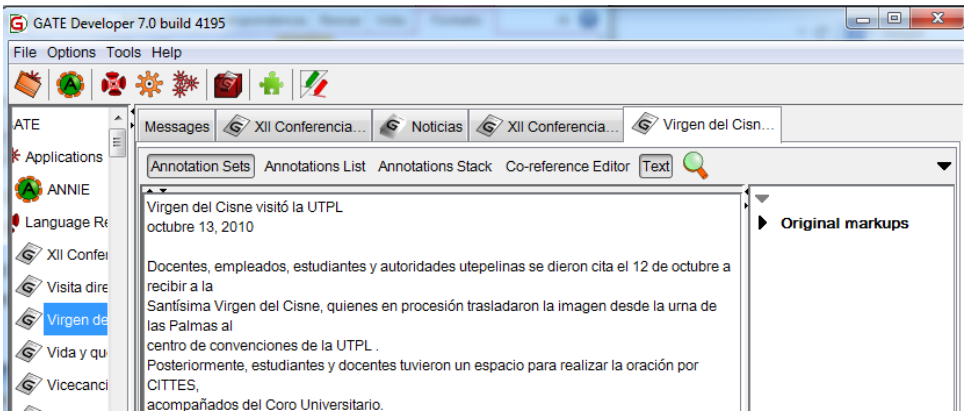

Figura 7. Noticia lista para etiquetarse.

Como se puede ver en la Figura 7, en la parte derecha de la pantalla de la herramienta, permite crear nuevas Anotaciones, en este caso Nombres de Entidades (NamedEntities) y Evento (Categorial a la que pertenece la noticia). Para lo cual en la parte inferior derecha, se escribe el nombre, y se da click en el botón *New,* tal como se observa en la Figura 8.

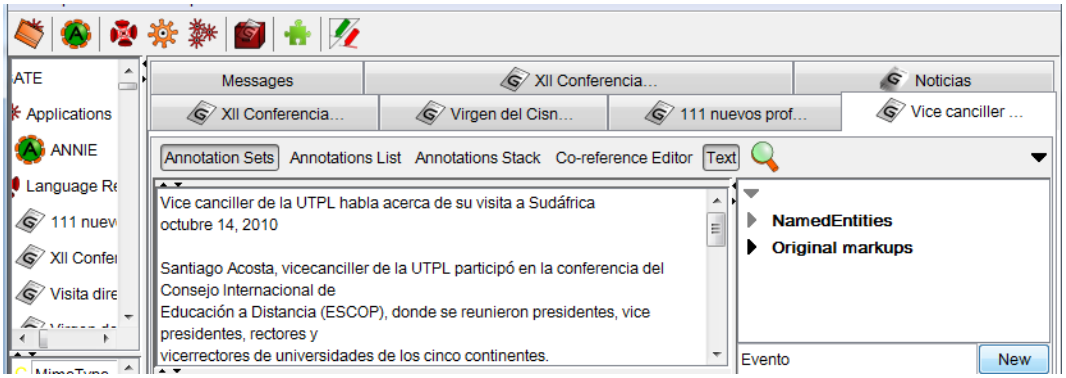

Figura 8. Creación de listas NamedEntities y Evento.

2.2.1. Etiquetado de Nombres de Entidades (Named Entities):

Para este proceso se tuvo que guiar bajo las definiciones de las conferencias MUC (Message Understanding Conference), donde se definen los siguientes nombres de entidades:

1. Nombres de Personas.
2. Organizaciones.
3. Locaciones.
4. Fechas.
5. Títulos de Persona.

Se ubica en la noticia y se identifica los diferentes nombres de entidades y se empieza a etiquetar según el tipo: nombre de persona, organización, locación, fecha y título de persona. En el ejemplo identificamos la entidad UTPL que es una organización como se muestra en la Figura.9.

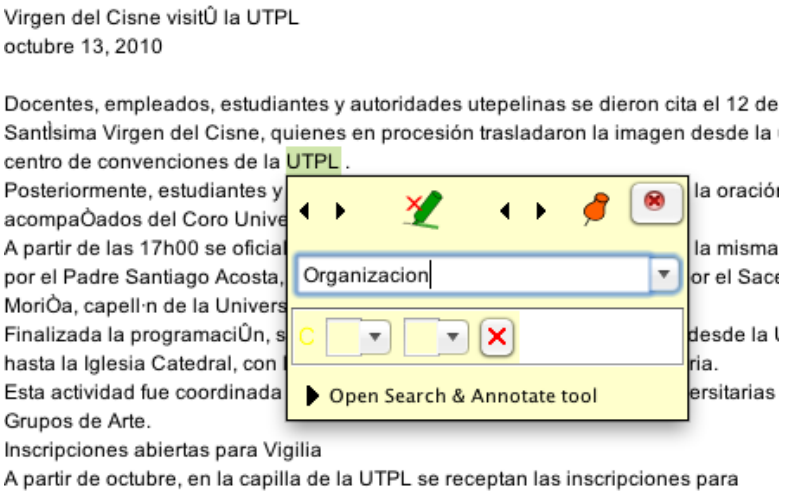

Figura 9. Proceso de etiquetado de NamedEntities.

El procedimiento es sencillo, se elige la palabra y automáticamente se presenta una ventana; como lo indica el ejemplo se le da el nombre al que pertenece “Organización” y queda guardado, como se muestra en la Figura 10. Se puede notar que ya se ha etiquetado como “Organización” a las palabras: CITTES, UTPL (3 veces); y como “Persona” se etiquetó: Santiago Acosta, Juan Manuel y Jesús.

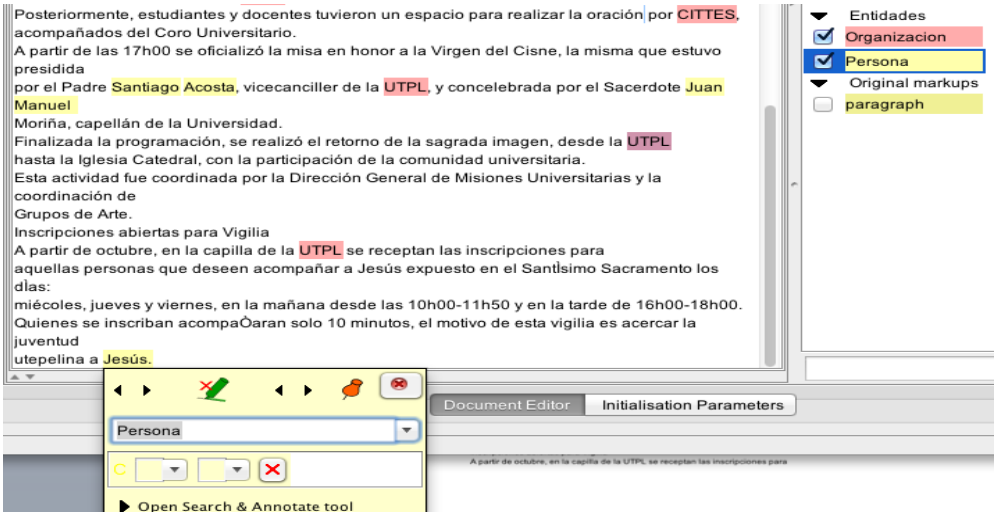

Figura 10. Palabras etiquetadas sobre la noticia.

2.2.2. Etiquetado de Eventos:

Para el proceso de Eventos, se tomó en consideración cuatro categorías de noticias UTPL: académica, investigativa, social/cultural y emprendimiento. Cada noticia pertenece a una sola categoría, y de acuerdo a un previo listado de palabras se etiquetó la noticia por ejemplo como se observa en la Figura 11; la noticia es de categoría SocialCultural. El proceso es igual a como se explicó en anteriormente para NamedEntities.

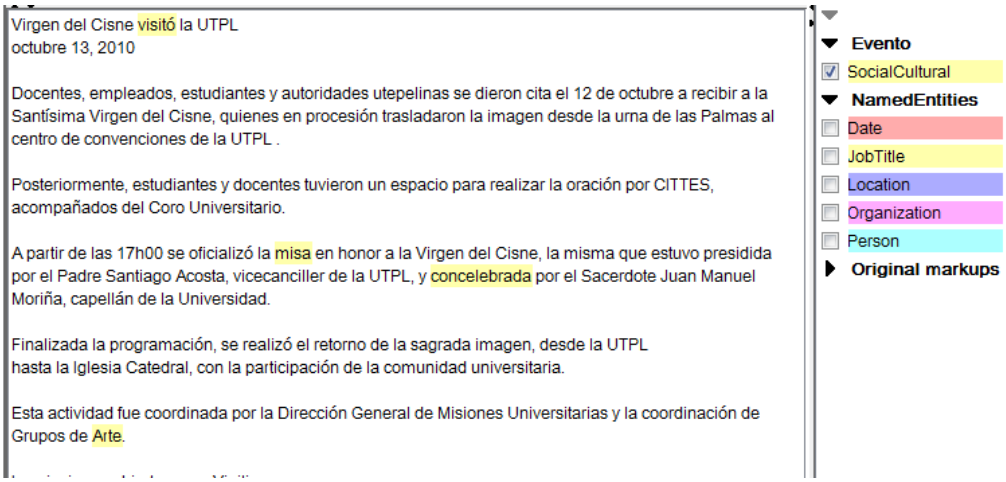

Figura 11. Proceso de etiquetado de Evento.

2.2.3. Guardar documento etiquetado:

Una vez etiquetado las 200 noticia, se procedió a guardar el archivo, el mismo que gracias a la herramienta GATE nos da en formato xml. Para lo cual se hace click derecho en la noticia etiquetada, desplegándose una ventana y se elige la opción *Save as XML,* ver Figura 12.

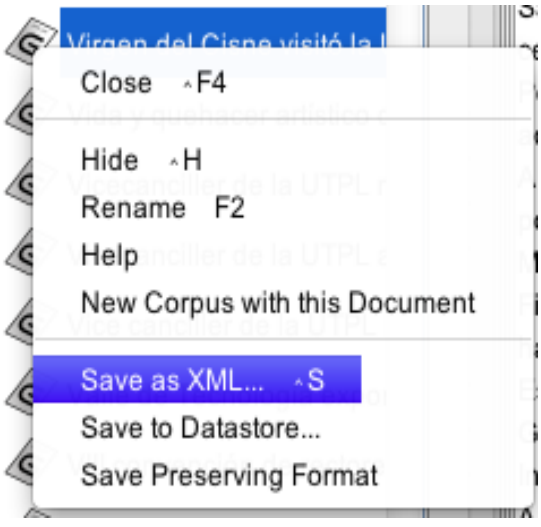

Figura 12. Guardar noticia en formato XML.

Luego aparece una ventana y se escoge la dirección donde se quiere guardar el archivo y se pulsa guardar. Ver Figura 13.

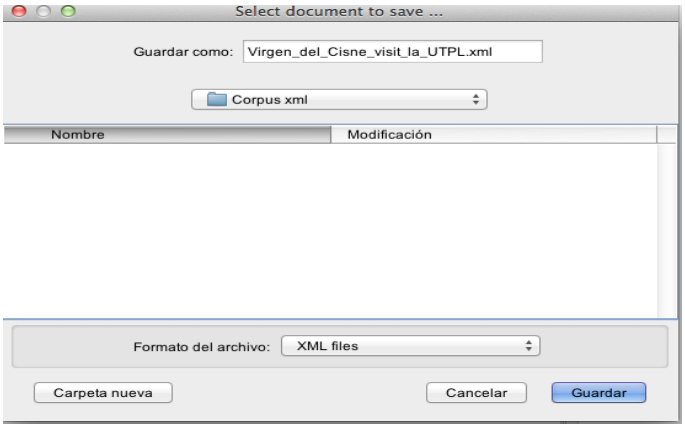

Figura 13. Noticia etiquetada guardad en carpeta.

Para esta etapa ya se tiene definido todos los nombres de entidades y categorías que intervienen en el corpus noticias, esto es gracias al etiquetado manual con la herramienta GATE.

2.3. Creación de listas o Gazzetteer en GATE

En este punto se creará los Gazzetteer en GATE, que consiste en un conjunto de listas (ficheros de textos plano) en las que se representa conjuntos de nombres, tales como nombres de ciudades, organizaciones, días de la semana, etc., y que se utilizan para reconocer en el texto dichas entidades. Estas listas pueden ser editadas e incluso se pueden crear otras nuevas

Primero se crea un archivo con extensión .lst como indica la Figura 14.

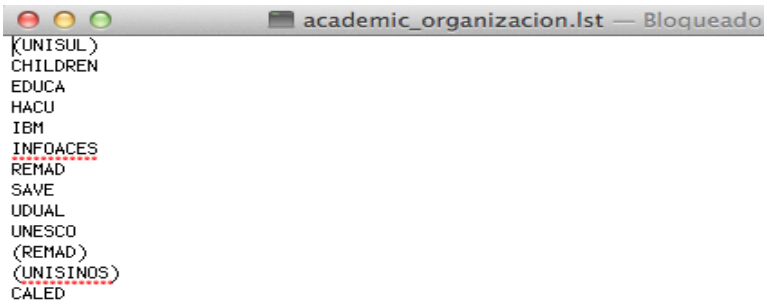

Figura 14. Creación de los Gazzetteer.

Una vez creado los gazzetteer necesarios para la aplicación, se debe ubicar en el directorio donde se encuentra la carpeta de *GATE*, → *plugins*, → *ANNIE* →*resourse* → *gazzetteer*; una vez abierta esta carpeta buscamos el archivo *lists.def*, es aquí donde vamos a agregar los nuevos gazzetteer. Ver Figura 15.

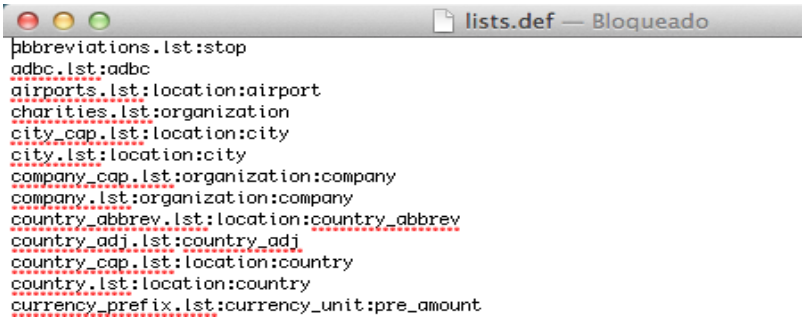

Figura 15. Modificación de archivo lists.def.

Para agregar el nuevo gazzettteer se debe colocar el nombre de la lista con la extensión .lst dos puntos y nuevamente el nombre de la lista como se muestra en la Figura 16.

```
org_base.lst:org_base
org_key_cap.lst:org_key:cap
org_key.lst:org_key
org_pre.lst:org_pre
org_spur.lst:spur
percent.lst:percent
person_ambig.lst:person_first:ambig
```

Figura 16. Formato para agregar las nuevas listas.

2.4. Creación de reglas o expresiones regulares utilizando JAPE.

JAPE (Java Annotation Patterns Engine) es uno de los recursos más importantes de GATE, es un transductor de estados finitos para las anotaciones de los documentos. Permite reconocer expresiones regulares para las anotaciones de los documentos.

2.4.1. Establecer relaciones entre entidades.

Primero se establecen patrones de cómo se encuentran la entidades dentro de las noticias. A continuación se presenta un ejemplo con un extracto de noticia académica.

María Isabel Loaiza, de la Dirección General Académica, precisó que la universidad cada ciclo fortalece el modelo de formación en competencias valorado por medio de los créditos UTPL-ECTS. brinda a nuestro estudiantes los servicios académicos necesarios para que puedan formarse como profesionales y sobre todo como seres humanos que sirvan a la sociedad, detalló...

Se Identifica la relación de las entidades, dando resultado las siguientes reglas:

María Isabel Loaiza: Entidad Persona.
Dirección General Académica: Entidad Organización
ECTS: Entidad Organización.

Para una mejor comprensión, a continuación se grafica la relación de entidades que se da en esta noticia. Ver Figura 17.

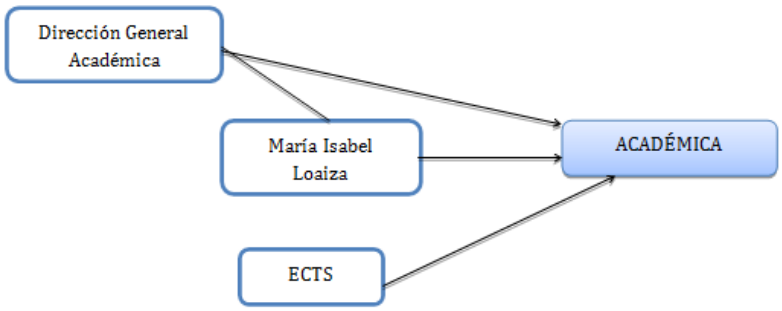

Figura 17. Relación de entidades.

Una vez identificadas todas las relaciones entre entidades en todas las noticias de categoría académicas", se unifica y se las coloca en un nuevo Gazzetter. Y se realiza el mismo proceso para todas las categorías: investigativa, social/cultural y emprendimiento. En este ejemplo el nuevo patrón quedaría de la siguiente manera. Ver Figura 18.

Figura 18. Patrón de categoría académica.

2.4.2. Crear archivos JAPE.

Una vez establecidos todos los patrones, se procede a crear los archivos extensión .jape. Ver Ejemplo Figura 19.

academic.jape

```
Phase:TokenUse
Input:  Lookup Token
Options: control = brill


Rule: academicPersonasRuler
Priority: 20
(
{ Lookup.minorType== academic_personas }
): academicoPerson
-->
:academicoPerson.PersonaAcademica = {kind= "academicPersonasRuler" }

Rule: academicOrganizacionRuler
Priority: 20
(
{ Lookup.minorType== academic_organizacion }
): academicoOrganizacion
-->
:academicoOrganizacion.InstitucionesAcademicas = {kind= "academicOrganizacionRuler" }
```

Figura 19. Ejemplo de un archivo jape.

Para la creación de los JAPE se utiliza un lenguaje por cada regla, en la siguiente tabla se explica el significado de cada sentencia:

Tabla 1. Reglas JAPE

Rule: academicPersonasRuler	**// Aquí se establece el nombre a cada regla en este caso academicPersonaRuler**
Priority:20	// Se da el valor de la prioridad de la regla
Lookup.minorType==academic_personas	// Se realiza una anotación de búsqueda "Lookup" con la función "minorType", que permite buscar del gazzeter academic_persona.
:academicoPerson	// Se asigna una etiqueta a la regla.
- - >	// Es el separador
: academicoPerson.PersonaAcademica = {kind= "academicPersonasRuler" }	// Se refiere a que academicoPerson, se le da una anotación de PersonaAcademica y una función a la clase "academicPersonasRuler"

Elaboración: Las Autoras

2.4.3. Crear archivos JAPE.

Una vez creadas las reglas en los archivos .jape, se procede a cargar. Para lo cual en la herramienta GATE, nos ubicamos en ***Processing Resources***, se da click derecho y se despliega una lista de la cual se elige la opción ***JAPE Transducer*** , tal como indica la Figura 20.

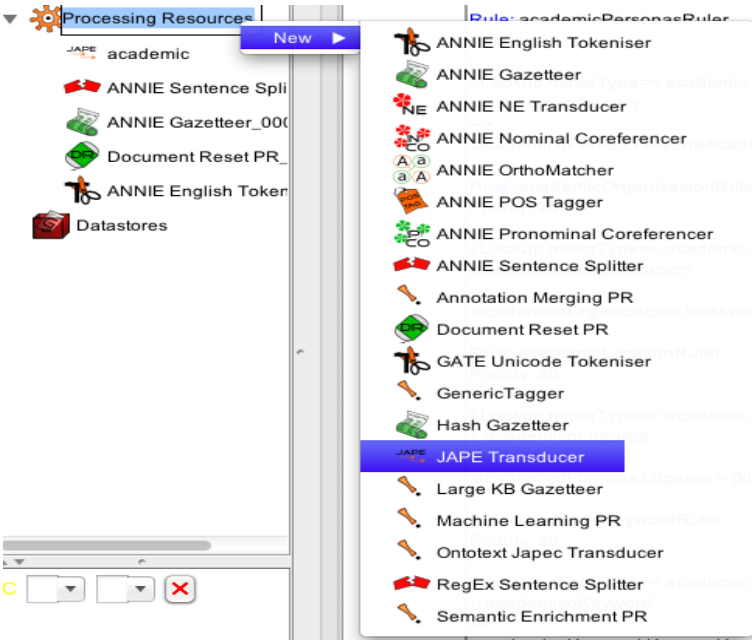

Figura 20. Cargando archivos jape.

Posterior se presenta una ventana, en donde se asigna un nombre de la categoría, en este caso digitamos “Emprendimiento”, luego hay que ir a la opción grammarURL, para poder ubicarse en la dirección donde está el archivo .jape, se elige “emprendimiento.jape” y se da click en la opción OK como lo muestra la Figura.21.

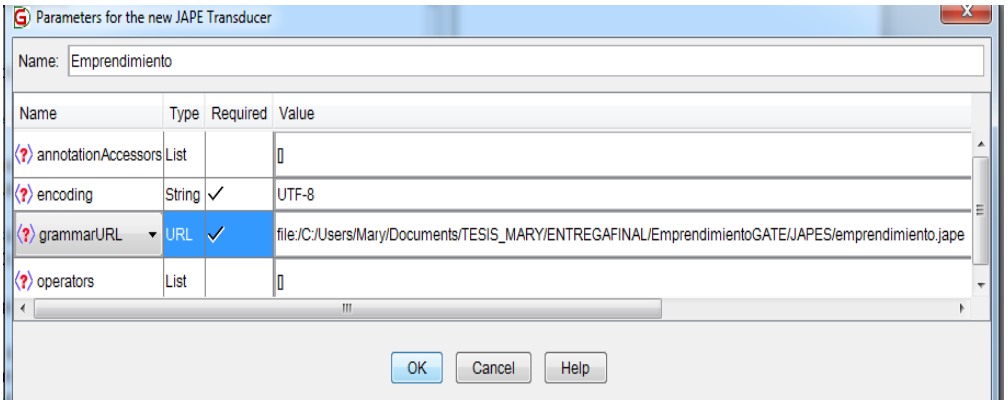

Figura 21. Ubicando archivo jape para ser cargado.

Una vez cargados los JAPE, en la herramienta se debe mostrar como indica la Figura.22. En este caso se tiene el archivo “Emprendimiento y academic”.

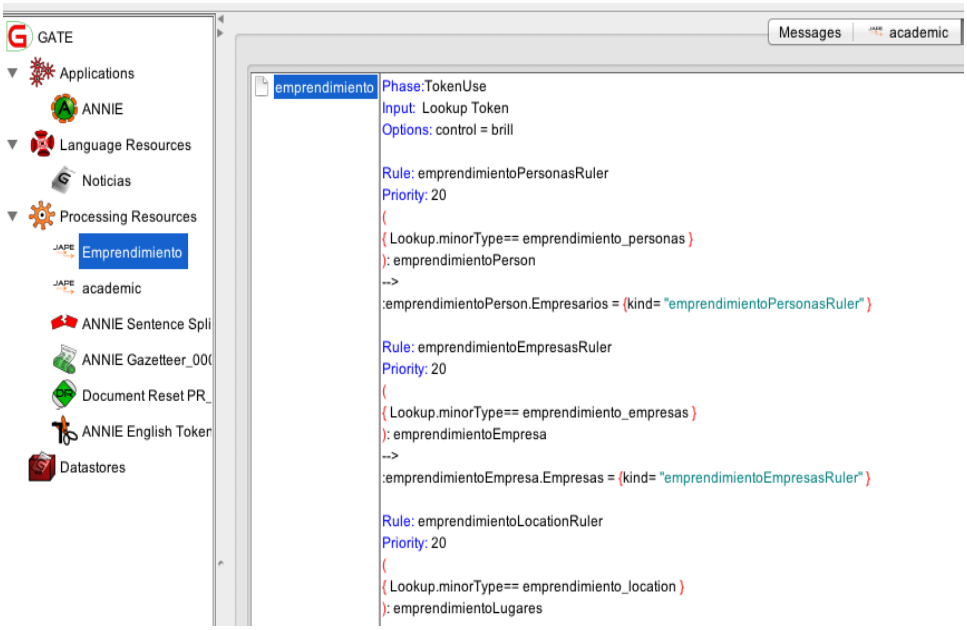

Figura 22. Archivos jape cargados.

2.5. Ejecución de ANNIE, sobre un corpus de prueba.

Una vez finalizado el proceso de creación de los gazzetteer y los Japes, se crea un nuevo corpus nuevo sin etiquetar y se precedemos a correr el ANNIE.

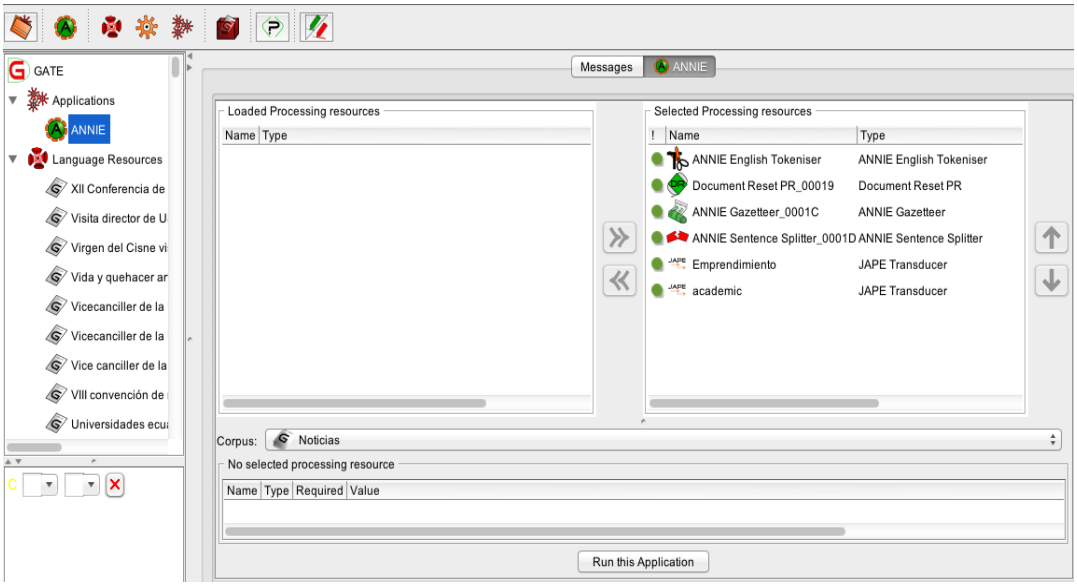

Figura 23. Archivos jape cargados.

Como se puede ver en la Figura 23, al ubicarse en ANNIE, los japes de la aplicación se los debe pasar a la parte derecha, y en la parte donde dice *Corpus,* debe estar el nombre del corpus de prueba en este caso se llama “Noticias”. Y se procede a correr ANNIE, eligiendo el botón *Run this Aplication..* Como podemos observar en la Figura 24.

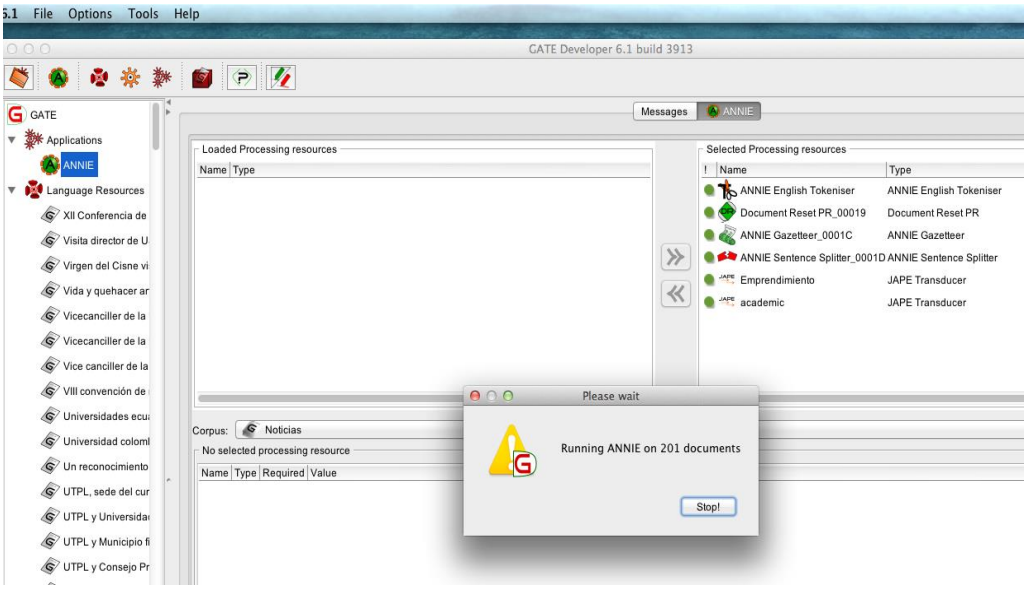

Figura 24. Corriendo ANNIE.

Una vez terminado este proceso, al ubicarnos en cualquier noticia, se nos va a mostrar, las palabras etiquetadas de acuerdo a las listas que se han generado. En este caso como se puede ver en Figura 25, se etiquetó una organización y un nombre de persona; y también se ha categorizado la noticia de tipo académica, cumpliendo con el objetivo esperado.

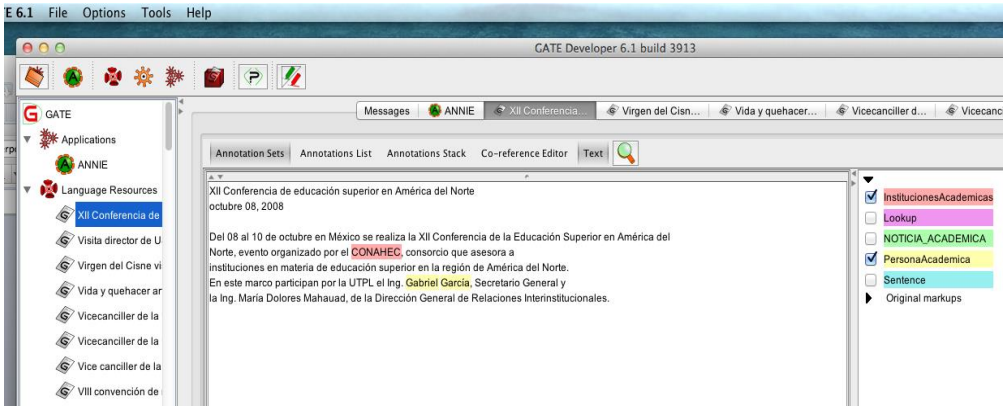

Figura 25. Resultados de ANNIE.

Anexo 2. Metodología de AA

En este anexo se detalla el proceso que se realizó para obtener nuestro demo de clasificación.

1. Se debe tener instalado WEKA ya que de esta herramienta vamos a consumir el algoritmo SMV. Para el desarrollo de la interfaz gráfica se lo efectuó en el entorno de desarrollo NetBeans en el lenguaje java.

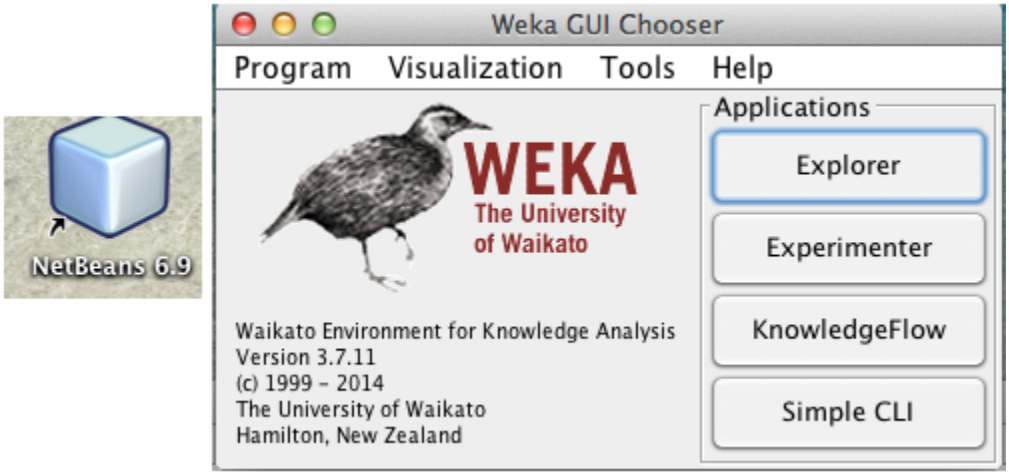

Figura.1 Herramientas para AA

2. Una vez instaladas las herramientas procedemos con el desarrollo del demo.

Clase de Clasificador

```
// Primero se importan paquete de librerías de weka.jar  y LibSMV.jar
package text_clasificador;
import java.io.BufferedReader;
import java.io.File;
import java.io.FileOutputStream;
import java.io.FileReader;
import java.io.IOException;
import weka.classifiers.lazy.IBk;
import weka.core.Attribute;
import weka.core.DenseInstance;
import weka.core.FastVector;
import weka.core.Instance;
import weka.core.Instances;
import weka.core.SelectedTag;
import weka.core.converters.ArffSaver;
import weka.filters.Filter;
import weka.filters.unsupervised.attribute.StringToWordVector;

public class ClasificadorSPAM {

// Constantes

public static String ETQ_SPAM = "spam";
public static String ETQ_NO_SPAM = "no-spam";

 // Parámetros
```

```
private int numVecinos;

   // Variables del objeto

private IBk clasificador;
private StringToWordVector filtro;
private Instances datasetInicial;
private Instances datasetFiltrado;

   // Constructor
   public ClasificadorSPAM(int numVecinos) {
      this.numVecinos = numVecinos;
   }

// Métodos privados
/**Crea e inicializa el clasificador k-nn (num. vecinos + ponderación distancia)     */

   public void inicializar() {
      this.clasificador = new IBk();
      this.clasificador.setKNN(this.numVecinos);
      this.clasificador.setDistanceWeighting(new
     SelectedTag(IBk.WEIGHT_INVERSE, IBk.TAGS_WEIGHTING));
   }
```

1) Crea un dataset con una instancia y con 2 atributos (TEXTO+CLASE) para cada fichero del directorio indicado.

2) Filtra ese dataset con un StringToWordVector para transforma el atributos TEXTO en un conjunto de atributos numéricos (1 valor por cada palabra/token seleccionada)

3) Con el dataset resultado de ese filtrado se entrena el clasificador

4) Las instancia de los datasets temporal no se necesitan, pero se debe mantener su cabecera (vector de Attribute)

```
param pathDirectorio
throws java.lang.Exception

 public void entrenarDirectorio(String pathDirectorio) throws Exception {
// Crear y cargar contenido textual de los ficheros
  datasetInicial = crearDatasetInicial();
      File directorio = new File(pathDirectorio);
      cargarFicherosEntrenamiento(directorio);
      System.out.println("\t... datos de entrenamiento cargados");
// Filtrar (en bloque) dataset inicial -> convertir texto en vector numérico
filtro = crearFiltro(datasetInicial);
      datasetFiltrado = Filter.useFilter(datasetInicial, filtro);
      datasetFiltrado.setClass(datasetFiltrado.attribute("CLASE"));
// Otra forma equivalente: datasetFiltrado.setClassIndex(0);
      System.out.println("\t... datos de entrenamiento filtrados");
// Entrenar clasificador
      clasificador.buildClassifier(datasetFiltrado);
      System.out.println("\t... clasificador entrenado");
// Para depuración -> guardar en disco los ARFF son archivos propios de WEKA
      volcarDatsets();
// Liberar las instancias (ya no son necesarias [IBk hace su propia copia])
// Bastará con mantener las cabeceras de los datasets (Attribute)
[usadas en evaluarDirectorio()]
      datasetInicial.delete();
      datasetFiltrado.delete();
   }
```

Recorre los ficheros del directorio indicado, los convierte en instancias WEKA y los clasifica.

1) Crea una instancia inicial TEXTO+CLASE

2) La filtra con el StringtoWordVector creado con el conjunto de entrenamiento

3) Clasifica la instancia filtrado y obtiene el índice de la clase predicha

```
   param pathDirectorio
   throws java.lang.Exception

public void evaluarDirectorio(String pathDirectorio) throws Exception {
      Instance instanciaInicial;
      Instance instanciaFiltrada;
      int idxClasePredicha;
      String clasePredicha;

      System.out.println("CLASIFICACION DEL DIRECTORIO ["+pathDirectorio+"]");
      File directorio = new File(pathDirectorio);
      File[] listaFicheros = directorio.listFiles();
      for (File fichero :listaFicheros) {
          instanciaInicial = crearInstanciaInicial(fichero);

 // Filtrar instancia a clasificar (una a una) -> covertir texto en vector numérico
          filtro.input(instanciaInicial);
          instanciaFiltrada = filtro.output();

//clasificar la instancia filtrada

                          idxClasePredicha = (int) clasificador.classifyInstance(instanciaFiltrada);

// Mostrar valor predicho

clasePredicha = datasetFiltrado.attribute ("CLASE") .value(idxClasePredicha);
          System.out.println("FICHERO ["+fichero.getName()+"] -> CLASE: "+clasePredicha);
      }
   }
```

Crea un dataset (inicialmente vacío) con 2 atributos para almacenar los datos de entrenamiento.

```
    - Atributo CLASE: atributo nominal con 2 valores: {spam, no-spam}.
    - Atributo TEXTO: atributo de tipo String.
    - Estable CLASE como atributo clase
    return

   private Instances crearDatasetInicial() {

 // Crear lista de clases

FastVector clases = new FastVector(2);
       clases.addElement(ETQ_SPAM);
       clases.addElement(ETQ_NO_SPAM);

  // Crear lista de atributos (cabecera del dataset)

   FastVector atributos = new FastVector(2);
   atributos.addElement(new Attribute("CLASE", clases));
   atributos.addElement(new Attribute("TEXTO", (FastVector) null));

   Instances dataset = new Instances("original", atributos, 100);
   dataset.setClass(dataset.attribute("CLASE")); // Otra forma

       return (dataset);
   }
```

```
     Crea y configura un filtro StringToWordVector
    param dataset
    return
    throws java.lang.Exception

private StringToWordVector crearFiltro(Instances dataset) throws   Exception {
                                StringToWordVector filtroTemp = new StringToWordVector();
```

Establecer el formato del dataset de entrada

```
filtroTemp.setInputFormat(dataset);

Indica que atributos procesar

  int[] idxAtributosFiltrar = {1};
  filtroTemp.setAttributeIndicesArray(idxAtributosFiltrar);          //          Otra          forma          equivalente:
filtroTemp.setSelectedRange("2");
      return (filtroTemp);
  }
```

Recorre el directorio con los ficheros de entrenamiento creando una instancia (CLASE, TEXTO) para cada uno

```
   param directorio

 private void cargarFicherosEntrenamiento(File directorio) {
      Instance instancia;

      File[] listaFicheros = directorio.listFiles();
      for (File fichero : listaFicheros) {
         instancia = crearInstanciaInicial(fichero);
         datasetInicial.add(instancia);
      }
  }
```

Crea una instancia (CLASE, TEXTO) para el fichero indicado

```
 param pathFichero
 return retorna el parámetro del fichero.

  private Instance crearInstanciaInicial(File fichero) {
      String texto = extraerTexto(fichero);
      String clase = extraerClase(fichero.getName());

                          Instance instancia = new DenseInstance(datasetInicial.numAttributes());
      instancia.setDataset(datasetInicial);
      instancia.setValue(datasetInicial.attribute("TEXTO"), texto);
      instancia.setClassValue(clase);
      return(instancia);
  }
```

Identifica la clase de un fichero: los ficheros SPAM tiene el nombre "spmsgXXXX.txt"

```
    @param pathFichero
    @return

  private String extraerClase(String nombreFichero) {
      if (nombreFichero.startsWith("spmsg")) {
         return (ETQ_SPAM);
      } else {
         return (ETQ_NO_SPAM);
      }
  }
```

Extrae línea a línea el texto del fichero

```
 param pathFichero
 return

private String extraerTexto(File fichero) {
    StringBuffer buffer = new StringBuffer();

    try {
                            BufferedReader in = new BufferedReader(new FileReader(fichero));
        String linea = in.readLine();
        while (linea != null) {
            buffer.append(linea + " ");
            linea = in.readLine();
        }
        in.close();
    } catch (Exception e) {
        System.err.println("Error en lectura de fichero [" + fichero + "] \n" + e.getMessage());
        System.exit(0);
    }

    return (buffer.toString());
}
```

Escribe los datasets en archivos ARFF

```
throws java.io.IOException

private void volcarDatsets() throws IOException {
    ArffSaver guardarARFF = new ArffSaver();

    guardarARFF.setInstances(datasetInicial);
    guardarARFF.setDestination(new FileOutputStream("\\NetBeansProjects\\text_clasificador\\inicial.arff"));
    guardarARFF.writeBatch();

    guardarARFF.setInstances(datasetFiltrado);
    guardarARFF.setDestination(new FileOutputStream("\\NetBeansProjects\\text_clasificador\\filtrado.arff"));
    guardarARFF.writeBatch();

    System.out.println("\t[los datasets creados se han volcado en inicial.arff, filtrado.arff]");
}

// Metodos estáticos

public static void main(String[] args) {

    try {

        ClasificadorSPAM miClasificador = new ClasificadorSPAM( 5 );
        System.out.println("Crear clasificador ... ");
        System.out.println("Entrenar clasificador ... ");
       miClasificador.entrenarDirectorio("\\NetBeansProjects\\text_clasificador\\train");
        System.out.println("Clasificar ficheros ... ");
       miClasificador.evaluarDirectorio( "\\NetBeansProjects\\text_clasificador\\test");
    } catch (Exception e) {
        System.err.println("Error en clasificación: " + e.getMessage());
    }
}
private static void mensajeError() {
    System.out.println("ERROR: parametros incorrectos");
    System.out.println("formato: java ClasificadorSPAM [num. vecinos] [dir. entrenamiento] [dir. evaluacion]");
}

private void inicializar(String string) {
    throw new UnsupportedOperationException("Not yet implemented");
```

```
    }
}
```

La clase principal

```
package text_clasificador;
import java.io.File;
import java.io.FileNotFoundException;
import java.io.IOException;
import java.util.HashMap;
import java.util.Iterator;
import java.util.Map;

public class Text_clasificador {
    public static HashMap<String,Number> categorias = null;
    public static String  categoria = "";
    public static void main(String[] args) throws FileNotFoundException, IOException, Exception {
java.awt.EventQueue.invokeLater(new Runnable() {
            public void run() {
                new prin().setVisible(true);
            }
        });
    }

    public static  void listar_directorio( File[] listaFicheros ){
        categorias = new HashMap<String,Number>();
        int no = 1;

        for (File fichero : listaFicheros) {
                if( fichero.isDirectory()  ){
                    categoria = fichero.getName();
                    categorias.put( categoria, 0 );
                    if( fichero.getName().contains(".txt") ){
                        categorias.put( categoria, no++ );
                    }

                    listar_directorio( fichero.listFiles() );

                }else{

                    if( fichero.getName().contains(".txt") ){
                        categorias.put( categoria, no++ );

                    }
                }

        }//end for

         Iterator it = categorias.entrySet().iterator();
         while (it.hasNext()) {
                Map.Entry pairs = (Map.Entry)it.next();
                 System.out.println( "Categoria ["+pairs.getKey() + "] =>" + pairs.getValue());

        }
    }

}
```