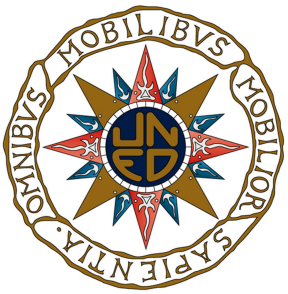
Universidad Nacional de Educación a Distancia
ETS. de Ingenieria Informatica
Departamento de Lenguajes y Sistemas Informáticos



Enlazado de tuits con entidades

Autor
Rafael Castro Barmina

Directores
Dra. Raquel Martínez Unanue
Dr. Víctor Fresno Fernández

Trabajo Final del Máster Universitario En I.A. Avanzada
Septiembre, 2014

Agradecimientos

A mi familia por su constante apoyo durante la realización de este trabajo.

A los directores de este trabajo, por sus valiosas observaciones y gran disponibilidad para ayudar y guiarme en todo momento.

A Diego Ceccarelli, por su ayuda con Dexter.

Resumen

El enriquecimiento de textos con información relevante es hoy una técnica fundamental para mejorar la calidad de muchas actividades de análisis de textos. Una manera de extraer información a partir de texto no estructurado es enriquecerlo con información relevante.

En este trabajo se realiza un estudio del problema de enlazado de entidades en el contexto de los tuits y se hace una propuesta de un pipeline para el enlazado de tuits con entidades extraídas desde Wikipedia.

Índice general

Índice de figuras

Capítulo 1

Introducción

Este capítulo introductorio presenta la motivación para la realización de este trabajo de investigación, así como la descripción del problema abordado y los objetivos propuestos. Finalmente se presenta cómo está organizado el resto de la memoria.

1.1. Motivación

El enriquecimiento de textos con información relevante es hoy una técnica fundamental para mejorar la calidad de muchas actividades de análisis de textos. Una manera de extraer información a partir de texto no estructurado es enriquecerlo con información relevante. La tarea de enriquecimiento de textos consiste en vincular conceptos y frases dentro de un texto con datos y recursos disponibles en algún repositorio de información, como la Web, por ejemplo. Estos elementos textuales, denominados menciones, conforman el punto de partida para la búsqueda y selección de recursos y representan los enlaces a la información encontrada (entidades).

El principal beneficio del enriquecimiento de textos automático es su capacidad para incorporar recursos a un texto de forma automática. Dicho proceso detecta menciones candidatas dentro de un texto y las vincula a unidades de información contextual, llamadas entidades, que aportan datos y recursos complementarios. Una vez reconocidos los elementos del texto sobre los que se trabajará, debe procederse a su enriquecimiento. Básicamente, la tarea implica la búsqueda, recopilación, extracción e integración de recursos e información relativos a una entidad dentro de uno o más repositorios de información y sirve como base de muchas actividades de análisis de textos tales como el análisis de sentimientos, la minería de opiniones, traducción automática, etc.

Una de las fuentes de datos más importante en la actualidad, de información en

tiempo real, son las redes sociales, en especial Twitter[1]. Twitter es la segunda aplicación social más popular y ha experimentado un crecimiento exponencial a lo largo de los últimos años. Twitter tiene millones de usuarios activos y con cientos de millones de tuits publicados a diario, se ha convertido en una plataforma popular para capturar y transmitir las experiencias mundiales actuales. Estadísticas recientes muestran que se publican cientos de millones de tuits por día. El gran tamaño y popularidad de Twitter ha empezado a atraer una cantidad significante de investigación desde varias perspectivas.

Los tuits a menudo ofrecen la información más reciente sobre cualquier fenómeno del ámbito mundial, mucho antes de ser cubierta por los medios, por lo que, mientras más rápido sea identificada esta información relevante, más valor tendrá, por lo tanto es importante desarrollar técnicas de enlazado de tuits con entidades, para poder extraer esta información.

Un sistema de enlazado típico realiza esta tarea en dos etapas: identificación de menciones y desambiguación. El proceso de identificación de menciones encuentra un conjunto de menciones candidatas en el documento de entrada y produce una lista de entidades candidatas por mención. Luego, el proceso de desambiguación selecciona las menciones más relevantes, y las entidades más probables entre las entidades candidatas.

El proceso de identificación explota un catálogo o base de conocimiento de entidades dada, para divisar las posibles menciones de entidades en la entrada, sea un texto, página web o tuit. Una manera común de realizar esta tarea es mediante el uso de un catálogo o repositorio de información.

La amplitud del repositorio y la precisión del sistema de detección de menciones tienen un impacto fuerte sobre el poder de recuperación del sistema de enlazado.

Wikipedia se escoge a menudo por la gran cantidad de información que abarca y ya que su número de páginas está creciendo constantemente. Además, los datos en Wikipedia están estructurados, permitiendo un mejor acceso automático a los datos, que digamos la Web, pero con una estructura menos formal que catálogos como Dbpedia[2], WordNet[3], CYC[4] y OpenCYC[5], lo cual dificulta un poco la tarea de extracción de información en comparación a estos, pero a la vez permite que personas sin conocimiento previo de la estructura del catálogo puedan editar o añadir información nueva, haciéndola uno de los repositorios de información más populares y con información más actual.

[1] `http://twitter.com`
[2] `http://dbpedia.org/`
[3] `http://wordnet.princeton.edu/`
[4] `http://www.cyc.com/`
[5] `http://www.cyc.com/platform/opencyc`

1.2. Terminología Básica

Antes de continuar, es necesario introducir algunas definiciones de uso común. En Recuperación de Información el término documento hace referencia, de forma genérica, a la unidad de texto almacenado por el sistema y disponible para su recuperación. De este modo, dependiendo de la aplicación o de su ámbito de uso, se trataría de artículos de prensa, páginas web, documentos legales, tesis doctorales, etc., bien completos, bien segmentados. Podemos, por ejemplo, procesar por separado cada uno de los capítulos de un libro o cada una de las secciones de un documento si consideramos que su longitud total es excesiva. Por su parte, colección denota el repositorio de documentos disponible para resolver las necesidades de información del usuario. Cada una de las unidades léxicas (palabras) que componen un documento —y por extensión, la colección —se denomina termino.

Entidad: definimos como entidad una página en Wikipedia no ambigua y terminal (o sea, una página que no es una categoría, lista de desambiguación o página de redirección). Nótese que una entidad puede estar representada por enlaces con texto diferente (variaciones). Mención: definimos como mención de entidad, o mención, al texto del enlace a la entidad y su contexto, donde su significado está definido inequívocamente por una entidad específica.

Repositorio: un repositorio de conocimiento, en nuestro contexto, consiste de un conjunto estructurado de entidades, cada una de las cuales puede tener una lista de posibles variaciones.

1.3. Enlazado de textos con entidades

El enlazado de textos con entidades consiste en: dado un texto, se intenta identificar los fragmentos de texto (también llamados menciones) que se refieren a alguna entidad que aparece listada en una base de conocimientos dada, por ejemplo Wikipedia. La ambigedad del lenguaje natural hace que esto sea una tarea no trivial, ya que la misma entidad puede estar mencionada por diferentes fragmentos de texto, como “UNED” o “Universidad Nacional de Educación a Distancia”. Por otro lado la misma mención puede referirse a diferentes entidades, por ejemplo, “Universidad” puede referirse a muchas universidades en concreto o a la película “Universidad de Monstruos”.

Un repositorio de conocimiento consiste de un conjunto estructurado de entidades, cada una de las cuales puede tener una lista de posibles variaciones. La tarea de enlazado se considera generalmente como un puente entre el texto no estructurado y un repositorio de conocimiento en formato digital. Esta tarea se ha aplicado con diferentes formatos digitales como texto y multimedia, y para diferentes tipos de texto como archivos, noticias y reportes en la Web. Una manera simple utilizada a menudo para enlazar texto con conceptos es buscar una correspondencia léxica entre partes del texto y los títulos de los conceptos. Sin

embargo, esta manera simple tiene muchas desventajas, incluyendo ambigedad (cuando se mezclan varios conceptos con la misma mención) y una posible pérdida de especificidad, cuando se identifican conceptos sin mucho aporte semántico.

En este trabajo se utiliza como repositorio estructurudo de entidades Wikipedia: cada artículo de Wikipedia (que no sea una categoría, desambiguación, lista o página de redirección) es considerado una entidad terminal, y el texto del enlace en Wikipedia une una fuente de posibles menciones con la entidad en sí. El proceso de identificación debe detectar todas las menciones y las posibles entidades asociadas con estas.

Notación y estructura de Wikipedia

Un texto delimitador para una página p de Wikipedia es un texto usado en otra página de Wikipedia para referirse a p. En Wikipedia, puede ser el título de p, uno de sus sinónimos o acrónimos, o puede ser una frase completamente diferente sintácticamente. A causa de la polisemia, un mismo delimitador puede aparecer muchas veces en Wikipedia y puede apuntar a páginas diferentes.

El contenido de Wikipedia tiene una estructura jerárquica y en forma de grafo gracias a los enlaces que conectan las diferentes entidades entre sí. Todo el contenido de Wikipedia está representado por páginas de un tipo u otro. Los artículos mantienen una estructura predefinida que permite extraer un sumario. Una vez que un artículo en particular es identificado, se pueden reunir conceptos relevantes siguiendo los enlaces a este y los enlaces desde este. Desafortunadamente, muchas veces estos enlaces no siguen relaciones semánticas, y se hace difícil separarlos.

Los textos delimitadores de los enlaces hacia un artículo proveen una fuente de sinónimos y otras variaciones de la mención. El número de enlaces hacia un artículo también es una medida útil para filtrar conceptos muy específicos o poco útiles.

Las redirecciones son páginas con el solo propósito de conectar un artículo con títulos alternativos.

Casi todos los artículos de Wikipedia están organizados en una o más categorías. Todas las categorías de Wikipedia descienden de un nodo común llamado “Fundamental”.

Cuando artículos múltiples tienen el mismo nombre, se utiliza un tipo especial de artículo para separarlos, las desambiguaciones. Cada página de acepción, tiene un texto corto explicando en que se diferencia esta de las demás acepciones.

Los delimitadores, que son el texto utilizado en los enlaces en Wikipedia, son muy útiles ya que representan la sinonimia, la polisemia y las diferentes variaciones de cada mención. Las páginas de desambiguación también realizan esta tarea, pero los delimitadores ya están etiquetados, por lo que no requieren procesamiento adicional del texto. Además el número de enlaces a la página ofrece una medida de cuan representativa es cada acepción.

Enlazado de documentos con Wikipedia

Para cada documento dado, es muy probable que Wikipedia sepa al menos algo sobre los temas discutidos por este, y pueda añadir información adicional. Si los temas presentes en Wikipedia son identificados en el documento, estos podrían ser enlazados entre sí, de manera que el usuario puede ahondar en estos y obtener más conocimiento.

Los temas detectados también pueden mejorar la manera de como un documento es modelado y comprendido por sistemas automáticos. Buscadores, sistemas de recomendación, etc. típicamente representan los documentos como conjuntos de palabras sin una estructura específica. Al consultar Wikipedia, estos sistemas podrían seguir los conceptos ahí presentes y las relaciones entre estos.

Así se resuelve la sinonimia, ya que todos los sinónimos apuntan al mismo concepto. La polisemia también es resuelta, ya que gracias a los nodos adyacentes en el grafo se puede inferir el significado correcto. Al navegar las relaciones entre los conceptos, se identifican los temas principales. Todo esto produce una representación del documento que puede ser automáticamente leída y es extremadamente informativa. El reto es detectar estos temas y crear los enlaces correspondientes de manera precisa y efectiva.

Selección de candidatos

El proceso de detección de menciones en un documento comienza por seleccionar todos los n-gramas dentro de este, y consulta si estos están presentes en algún texto delimitador de Wikipedia. Para eliminar términos no útiles como artículos gramaticales y preposiciones, comúnmente se utilizan stoplists, que son listas de palabras funcionales carentes de significado real. Hay listas de estas para cada lengua. Otra forma de filtrado, es utilizando la probabilidad de que el n-grama sea un enlace si se encuentra en un artículo de Wikipedia.

Desambiguación de candidatos

Una vez que los términos candidatos han sido identificados, se verifican las posibles entidades entre todos los textos delimitadores. Aquí se da la ambigedad ya que una mención puede señalar a muchas entidades posibles. Para desambiguar estos candidatos, generalmente se utiliza algún mecanismo de votación, que permite escoger unos pocos, o el candidato más relevante.

Los dos pasos anteriores producen un conjunto de asociaciones entre términos en el documento y los artículos de Wikipedia que los describen. La tarea final consiste en escoger los temas que son verdaderamente relevantes al documento para ser enlazados.

Enlazado de tuits con entidades

Los tuits son mensajes cortos de texto, con un máximo de 140 caracteres. Además de ser muy cortos para contener un contenido semántico significativo, su lenguaje y sintaxis son muy diferentes de otros documentos en la web, ya que los usuarios de Twitter tienden a usar muchas abreviaturas y formas cortas de las palabras para ahorrar espacio, lo que hace aún más difícil la tarea de

extraer contenido semántico a partir de los tuits.

Los tuits son publicados públicamente por defecto, haciendo de twitter un enorme recurso de intercambios de información. Twitter incorpora el uso de menciones (para referenciar a otro usuario) y retuits (similares a un reenvío de correo).

Otra característica distintiva de Twitter es el uso de hashtags (#), los hashtags son usados como identificadores de mensajes relacionados con mismo tema. Al incluir un hashtag, el autor especifica el tema o temas del tuit. Dada la estructura fragmentada de la información en Twitter, a través de los hashtags se crean conversaciones, que siguen un mismo tema común.

La naturaleza de los tuits ofrece nuevos retos para la tarea de enlazado, los métodos tradicionales de identificación de entidades que actúan sobre documentos con formato correcto dependen mucho de las características lingísticas del texto, tales como la capitalización, las etiquetas POS de palabras previas, etc. Sin embargo, los tuits son cortos y usualmente informales. Muchas veces tienen faltas de ortografía y errores gramaticales. A causa de estas características, los métodos tradicionales no son muy efectivos para enlazar tuits.

Ya que la simple correspondencia léxica de las partes del texto con los títulos de los artículos no es precisa, especialmente para los tuits, por su estructura de texto corto y uso del lenguaje coloquial, etc., se necesita un mecanismo para mejorar la precisión del enlazado. Para esto se puede utilizar algún mecanismo de aprendizaje automático, produciendo un ranking entre las entidades candidatas para cada mención.

Las maneras de realizar la desambiguación de las entidades se pueden dividir en locales, donde se trata de desambiguar cada mención por separado, utilizando pistas como la similitud textual entre la mención y el título de cada página candidata, y globales, cuando todas las menciones en un documento son desambiguadas simultáneamente, llegando a un conjunto de desambiguaciones coherente.

Por ejemplo, si nos encontramos una mención de “raf” en el tuit y además encontramos las menciones “aviones” y “militar”, es mucho más probable que “raf” se refiera a las RAF, las fuerzas aéreas del Reino Unido, y no a la película RAF, basada en la organización de izquierda radical alemán.

Las formas de desambiguación globales explotan la estructura de grafo de Wikipedia para obtener coherencia en el conjunto resultante, siendo la más común la estimación de cuan relacionadas están las entidades en Wikipedia entre sí (tomándolas de par en par). Como se señaló anteriormente, el reto al enlazar tuits con entidades es que los tuits son mensajes informales, cortos y con ruido.

1.4. Objetivos

Los objetivos de este trabajo de investigación se pueden resumir en los siguientes puntos:

- Estudio del problema de enlazado de entidades en el contexto de los tuits.
- Estudio de aplicación de técnicas de NLP específicamente sobre tuits.
- Estudio y propuesta de un modelo de enlazado de tuits.
- Evaluación experimental del modelo propuesto.

Capítulo 2

Estado de la cuestión

En este capítulo se realiza un estudio del estado de la cuestión en el tema de enlazado de textos con entidades, en particular sobre textos cortos como los tuits, enumerándose los enfoques mas actuales y al final se hace una reflexión sobre los problemas persistentes. Además se investiga el estado de la cuestión de la normalización léxica específica de tuits, como fase anterior necesaria para la aplicación de técnicas de NLP sobre tuits.

2.1. Introducción

Dada la abundancia actual de contenido generado por usuarios, se ha creado un interés en procesar este contenido para extraer información. La naturaleza informal de dicho contenido y la diversidad geográfica y social de los usuarios se refleja en su lenguaje escrito, por ejemplo en microtextos como los de Twitter y los SMS, siendo frecuente la aparición de fenómenos lingüísticos como emoticonos para mostrar emociones, abreviaturas, sustituciones léxicas, entre otros, lo cual hace que la tarea de procesar texto en este formato sea muy difícil. Por ejemplo, en el caso particular de Twitter, el número máximo de caracteres por mensaje está limitado a 140, por lo que es común encontrar abreviaciones y contracciones no-estándar, esta limitación es heredada de los mensajes SMS, donde algunas palabras o silabas pueden ser representadas por letras o números que tienen la misma pronunciación pero cuyo tamaño es menor. Aunque el tipo de lenguaje utilizado en Twitter es diferente al de los SMS, siendo el de Twitter más complejo, ya que los SMS típicamente están dirigidos a una persona, muchas veces conocida, y en Twitter es a un público mayor, además los temas son diferentes. En un artículo reciente de Idibon[1] se analiza la complejidad del texto en Twitter comparado con literatura tradicional. A causa de esta complejidad y el ruido que tiene el contexto se dificulta la detección de palabras malformadas.

[1] `http://idibon.com/readability-social-media-literature`

Estos problemas están reflejados en varios artículos que muestran como sistemas del estado de la cuestión de procesamiento de lenguaje natural (NLP) funcionan con un rendimiento mucho menor sobre textos cortos. Por ejemplo, en etiquetado POS, la precisión del etiquetador de Stanford (Toutanova et al, 2003) disminuye desde un 97 % sobre texto del Wall Street Journal a una precisión de 85 % en Twitter (Gimpel et al, 2011). En la tarea de reconocimiento de entidades nombradas (NER), el reconocedor de Stanford entrenado sobre el dataset de CoNLL alcanza un 44 % de medida F (Ritter et al, 2011), obteniendo un 86 % sobre el conjunto de entrenamiento CoNLL (Finkel et al, 2005), etc.

Aplicar técnicas de NLP a los millones de tuits generados a diario sería de gran interés para extraer información a partir de estos, pero las alteraciones del lenguaje antes descritas lo dificultan. Una solución es transformar dicho texto a lenguaje estándar, es decir, normalizarlo. La normalización es un proceso que va a mejorar la aplicación posterior de técnicas lingüísticas, por lo que en este trabajo la primera fase del proceso de enlazado es una fase de normalización.

A continuación, se describirá el proceso de normalización sobre tuits y como ha sido abordado en este trabajo, haciéndose un breve repaso sobre el estado de arte en la normalización de tuits, antes de pasar al estado de la cuestión del enlazado de entidades.

2.2. Normalización de tuits

La tarea de normalización de este tipo de texto es una tarea difícil. Tiene similitud con la tarea de corrección ortográfica, pero se diferencia en que el texto es alterado muchas veces intencionalmente, para denotar énfasis, tener que teclear menos, por el tipo de identidad social o regional, etc.

La normalización es muchas veces imposible sin cambiar el sentido o la intención del texto (Eisenstein 2013). La variación en el lenguaje no es resultado de pasar texto estándar a través de un canal con ruido, sino que es una modificación pragmática e intencional (Du Bois, 2007). Al eliminar estas variaciones se eliminarían estas capas extras de contenido semántico.

Esto no niega el gran potencial que tiene la investigación de estas variaciones ortográficas a través de una combinación de contexto local, similitud entre cadenas de texto y otras técnicas basadas en autómatas finitos.

Se pueden distinguir tres tendencias principales a la hora de normalizar este tipo de textos. La primera emplea técnicas de traducción automática, la segunda se basa en corrección ortográfica y la tercera usa técnicas de reconocimiento del habla.

La aplicación de técnicas de traducción automática se ha demostrado útil para normalizar

SMS (Aw et al, 2006) tomando como idioma origen los textos no normativos y como idioma destino su equivalencia normalizada. Este sistema también se

ha usado para traducir textos en lenguaje SMS al español (López et al, 2010). Sin embargo, estas propuestas de traducción necesitan corpus relativamente grandes previamente normalizados y alineados para obtener buenos resultados (Kaufmann, 2010).

El uso del modelo de Shannon para canales con ruido (Shannon, 1948) se suele emplear en los sistemas de corrección automática para realizar una corrección ortográfica a nivel de palabra (Choudhury et al, 2007). Por último, las técnicas de reconocimiento automático del habla (RAH) se basan en la hipótesis de que la mayoría de las variantes léxicas no-estándar tienen una equivalencia homófona estándar (como en fone - phone). Empleando algoritmos fonéticos para codificar la pronunciación de la palabra a normalizar se genera una lista de candidatos homófonos de la cual se extrae la palabra normalizada mediante modelos del lenguaje (Gouws et al, 2011). Los sistemas de normalización no-supervisada basados en esta técnica han obtenido los mejores resultados (Han y Baldwin, 2011).

La mayoría de los sistemas de normalización sobre tuits atacan principalmente el problema de las OOV, las cuales son palabras fuera del diccionario, y tratan recuperar la forma canónica de estas. Por ejemplo, en (Han et al 2011) comparan la distribución de palabras OOV en Twitter y SMS con otros tipos de datos, mostrando que los datos de Twitter tienen una gran cantidad de estas, además, muchas de estas palabras malformadas son ambiguas, dificultando el uso de simples sistemas de reglas. La mayoría de las aproximaciones están basadas en diccionarios sobre los que aplicar métricas de similitud ortográfica para encontrar candidatos para sustituir una OOV. Entre los recursos léxicos utilizados se usan principalmente diferentes diccionarios o tesauros para buscar propuestas normalizadas tales como el diccionario de la DRAE (Diccionario de la Real Academia Española), el de la BNC (British National Corpus) y correctores ortográficos como Aspell[2] y Hunspell[3]. Para detectar entidades nombradas se utilizan catálogos de entidades como Wikipedia[4] o Freebase[5], además de compilaciones de nombres propios como la disponible en (`http://www.cs.cmu.edu/afs/cs/project/ai-repository/ai/areas/nlp/corpora/names/0.html`) y otras listas de entidades nombradas sui generis. Para detectar slang y abreviaturas informales comunes existen diccionarios compilados de variantes y slang, estos diccionarios ofrecen buena precisión pero bajo recall, al no abarcar todas las posibles modificaciones, dado el alto grado de creatividad léxica y el gran número de formas no-estándar observadas en Twitter; o listas de frecuencias a partir de corpus de tuits para detectar y normalizar cambios habituales propios de Twitter, por ejemplo en (Han et al, 2012, Owoputi et al 2013), se construye un diccionario de variantes léxicas posibles a partir de su distribución en un clúster de gran volumen (10 millones de tuits en inglés en el trabajo de Han) extraído de Twitter, ordenadas por su similitud como cadenas de texto, como ejemplo tomorrow

[2] `http://aspell.net/`
[3] `http://hunspell.sourceforge.net/`
[4] `http://wikipedia.com`
[5] `https://www.freebase.com/`

→ tmrw, 2moro.

Por tanto, lo más común en el estado de la cuestión es utilizar un conjunto de técnicas de normalización, también llamadas transductores, y haciendo uso de diferentes arquitecturas: en cascada, como pipeline o como módulos distribuidos. En el mismo artículo de Han, presentan un enfoque el cual consiste en generar primero una lista de formas canónicas léxicas candidatas, basada en la variación morfológica y fonética. Luego, todas estas formas candidatas son ordenadas con respecto a varias características extraídas a partir del contexto con ruido y la similitud entre los términos originales.

En este trabajo se utiliza como módulo de normalización TwitIE, el cual es parte de la suite de aplicaciones de NLP GATE (General Architecture for Text Engineering [6]) desarrollada por especialistas de la Universidad de Sheffield y utilizada ampliamente a nivel mundial. TwitIE tiene arquitectura de pipeline y consiste en un conjunto de módulos, que se pueden utilizar por separado o en grupo, y contiene un tokenizador para tuits, un diccionario de sustituciones, también llamado "gazetteer", un etiquetador POS (con modelos intercambiables, por defecto usa el de Stanford adaptado a Twitter) y un transductor de entidades nombradas. TwitIE ha sido escogido por su buen rendimiento y por la facilidad de modularización. Puede ser utilizado para extraer información directamente o servir de base para tareas más complejas de NLP.

En el capítulo tres será descrita su funcionalidad en más detalle y también como fue utilizado para la tarea de este trabajo específicamente.

2.3. Enlazado de entidades

La tarea de enlazar texto libre con conceptos contenidos en bases de conocimiento, y Wikipedia en especial, se ha hecho popular en años recientes, con muchos trabajos en el área del enlazado de menciones en un texto con conceptos (Mihalcea y Csomai, 2007; Milne y Witten, 2008a, b; Kulkarni y otros, 2009; He y otros, 2011; Ratinov y otros, 2011). A partir de estos artículos y otros tantos, ya se han implementado herramientas que se utilizan a nivel industrial, como Wikify! (Mihalcea y Csomai, 2007), Wiki3C (Jiang et al, 2013) y OpenCalais[7].

Uno de los retos mayores al realizar este proceso de enlazado, es el de la desambiguación de conceptos y entidades, en (Ratinov y otros, 2011) se hace una investigación a fondo de métodos locales (que resuelven una mención a la vez) y globales (que intentan hacer la desambiguación de varias menciones similares a la vez) para la desambiguación utilizando Wikipedia, tomando como base la estructura de links de Wikipedia, lo cual permite enlazar conceptos similares sin tener que analizar todo el contenido del artículo. Esta idea fue utilizada por primera vez en (Milne y Witten, 2008), donde los autores proponen utilizar como medida de similitud la cantidad de enlaces salientes que tienen en común

[6] `https://gate.ac.uk/`
[7] `http://www.opencalais.com/`

dos artículos que describen dos conceptos. Ratinov y coautores comparan variantes que utilizan esta información para una desambiguación global versus variantes locales. Muestran que aunque las variantes globales se pueden mejorar, con solo variantes locales se logra una buena precisión y nivel de cobertura (precision/recall).

La extracción y enlazado de conceptos a partir de tuits es un tema que se ha comenzado a tratar más recientemente. Dos de los problemas mayores de esta tarea son la gran cantidad de información producida en tuits en los últimos años y la inmensa cantidad de posibles variaciones de las menciones dada por la ausencia de un contexto de desambiguación más extenso.

En (Milne y Witten, 2008), se muestra que es más efectivo y permite una mejor recuperación, realizar la detección de posibles menciones y la desambiguación de estas en un solo paso conjunto.

Meij en su tutorial (Meij, 2013) formaliza este proceso de enlazado, dividiéndolo en 3 etapas, MD (detección de menciones que pueden ser enlazadas a una entidad en Wikipedia), LG (generación de links candidatos) y DA (desambiguación y mejora usando algún tipo de contexto) y propone en (Meij y otros, 2012), usando una colección de entrenamiento clasificada a mano, un método basado en aprendizaje supervisado con un conjunto de atributos específico que muestra una buena precisión. Su método solo considera la similitud entre la mención (tuit) y la entidad (artículo Wikipedia).

Dado el pequeño tamaño del texto de los tuits, se necesita un mecanismo para mejorar la precisión del enlazado, ya que un tuit pudiera potencialmente generar muchas entidades no relevantes si solo se toma en cuenta la correspondencia lexicográfica entre los términos del tuit y el título de la entidad en Wikipedia. Más concretamente, para la mención de detecciones, Meij divide el tuit en n-gramas candidatos para ser enlazados. A partir de esos se forma una lista ordenada, de la cual se seleccionan los mejores candidatos, de esta manera no hay que verificar cada combinación de n-grama, haciendo el algoritmo mucha más efectivo. En (Meij y otros, 2012) se comparan tres medidas de orden de los candidatos,

1. Basándose en la similitud lexicográfica entre el n-grama candidato y una entidad en Wikipedia, luego se comparan cuan significativas son las entidades correspondientes. También se usa una heurística, se escogen primero los n-gramas de más términos, si no hay entidad correspondiente, se analizan los n-gramas constituyentes, Esta es la medida más simple.

2. Basándose en un modelo del lenguaje, se encuentran relaciones entre n-gramas candidatos y entidades en Wikipedia, y se comparan estas.

3. La medida vista finalmente está constituida por métodos ya probados, como el de Milne y Witten, combinados con una medida de lo comunes que son los términos, esta medida está basada en la frecuencia relativa con la cual el n-grama es usado como texto de ancla (enlace) para una entidad concreta.

Esta última medida es la escogida en el tutorial como más efectiva para la detección de menciones, siendo la idea principal el mantener un conjunto ordenado de menciones candidatas a partir de la cual generar los enlaces.

En la segunda etapa, la de generación de enlaces candidatos a partir de la lista ordenada obtenida en el paso anterior, se utiliza un algoritmo supervisado de aprendizaje automático para determinar los enlaces a mantener. Más específicamente, este algoritmo tiene como entrada una colección de entrenamiento clasificada a mano (relaciones n-grama - entidad) y un conjunto amplio de características (de término, de entidad, de término-entidad) de los ejemplos de esta colección. Un ejemplo de característica de término es la probabilidad de que este sea usado como ancla a una entidad de Wikipedia, un ejemplo de una característica de concepto es el número de artículos de Wikipedia con enlaces hacia este y una característica mixta es la importancia del termino t para la entidad e, dada por TF x IDF(e,t), donde TF es la frecuencia con que aparece el término en el artículo e IDF es la frecuencia inversa del documento. Para más detalles sobre las características usadas ver (Meij y otros, 2012), aunque este conjunto es actualizado constantemente con nuevas características que a veces pueden reemplazar algunas existentes al contribuir a un enlazado mejor. Los algoritmos de aprendizaje utilizados en el tutorial son el de bosques aleatorios (random forests, RF) y el de gradient boosted regression trees (GBRT) basado en RF (ver C. J. C. Burges et al, 2011, por ejemplo), estos algoritmos tienen la ventaja de relativamente no depender de la configuración de los parámetros, ser resistentes al overtting, y además ser fácilmente paralelizables.

Como en la mayoría de los métodos utilizados luego del trabajo de (Milne y Witten, 2008), la etapa de desambiguación de menciones se realiza implícitamente al hacer la detección de menciones y producir la lista ordenada da de posibles menciones.

Para los experimentos de enlazar a conceptos, se utilizó una copia de Wikipedia de 2010 como corpus, con 3483213 artículos originales y 71204142 enlaces entre artículos. Luego, dos voluntarios anotaron manualmente un conjunto de 562 tuits, con un promedio de 36.5 términos de longitud. Este conjunto de tuits anotados está disponible públicamente.

Otro trabajo parecido es el de (Guo y otros, 2013), el cual sigue el modelo visto en (Meij y otros, 2012) y propone un algoritmo basado en SVMs que optimiza conjuntamente estas dos fases combinando el uso de varios atributos de primer y segundo orden, desambiguando todas las entidades simultáneamente usando un algoritmo de ranking basado en la teoría de grafos, con las aristas definidas por los enlaces de Wikipedia. Este algoritmo considera las relaciones entre entidad y entidad, y no solo entre mención y entidad.

Una propuesta similar es la de (Liu y otros, 2013), quienes proponen un método de inferencia colectiva que simultáneamente desambigua un conjunto de menciones. Su método asume que “menciones similares deben ser enlazadas con entidades similares” y explícitamente modela e integra la similitud entre menciones en el framework propuesto.

En este artículo, se compara la efectividad del algoritmo propuesto en comparación al de Meij, tomando como referencia base Wikify! Utilizando el mismo conjunto de 562 tuits (502 tuits en estos momentos, ya que varios tuits han sido borrados) descrito por Meij, los autores obtuvieron los siguientes resultados:

	Precisión	Recall	F-measure
Wikify!	37.5	42.1	39.6
Meij	73.4	63.2	67.9
Liu	75.2	67.5	71.1

Estos resultados muestran un incremento de la efectividad en comparación con el algoritmo de Meij (que solo utiliza características de similitud entre mención y entidad) al utilizar características globales, relacionadas con la similitud entre entidades y la similitud entre menciones. Otras maneras de mejorar la capacidad de desambiguación propuestas por diferentes autores incluyen expandir el contexto como en (Guo y otros, 2013) y en (Cassidy y otros, 2012), por medio de tuits adicionales.

Los resultados de los experimentos de Guo y demás no son comparables fácilmente con los de otros autores, ya que utilizan un subconjunto de entidades del conjunto presentado en TAC-KBP2009, en vez de Wikipedia, y además su objetivo es relacionar menciones similares entre tuits, más que enlazar todas las menciones candidatas con entidades, por lo que no ofrecen resultados comparables.

En el artículo de Cassidy y demás, se experimenta sobre el conjunto de datos descrito en Meij (502 tuits anotados). Se comparan dos maneras de mejorar la capacidad de desambiguación del algoritmo, una añadiendo la información del autor al tuit, o sea, usando el autor como característica adicional, y la otra consiste en agrupar los tuits por tema, o sea decorarlos con una característica adicional, el tema o área del tuit. Los resultados descritos son los siguientes:

	Precisión	Recall	F-measure
Por autor	62.2	39.2	45.6
Por clúster	63.2	38.0	47.5

Aunque el performance es peor que con el algoritmo de Meij, es un algoritmo mucho más simple, y por tanto más veloz, y solo utilizando información adicional como contexto, ofrece un performance aceptable sobre el conjunto de datos de prueba.

Otras implementaciones y herramientas que ayudan a la tarea del enlazado de tuits con entidades son las siguientes: WikipediaMiner que es una herramienta que permite extraer el contenido semántico de Wikipedia, como su nombre indica, al brindar acceso unificado a la estructura y contenido de Wikipedia y detecta y desambigua temas en Wikipedia cuando son mencionados en documentos; Dexter[8], una plataforma para el enlazado de entidades, que implementa algunos algoritmos populares y provee las herramientas necesarias para desarro-

[8] `http://dexter.isti.cnr.it/`

llar una técnica de enlazado de entidades y TAGME[9], que provee una API para la identificación en tiempo real de menciones y su enlazado a páginas de Wikipedia, desambiguando por medio de un algoritmo de votación.

WikipediaMiner está basada en (Milne y Witten, 2008), y es descrita en (Milne y Witten, 2012). WikipediaMiner incluye scripts en PERL que procesan una copia offline de Wikipedia, extrayendo información como el grafo de enlaces y la jerarquía de categorías. Además tiene la posibilidad de comunicarse con una base de datos en MySQL, que mantiene el contenido indexado, lo que permite un acceso inmediato a la información, y lo más importante, ofrece una API en Java, que modela y abstrae la estructura y el contenido de Wikipedia y también brinda funciones de lenguaje útiles, por ejemplo para comparar términos y conceptos semánticamente.

Dexter provee clases con una clara separación entre las tres frases del enlazado con entidades: la de detección de menciones, la generación de links candidatos y la de desambiguación entre entidades. Dexter consta de varios módulos, siendo el principal Dexter Core. Este módulo contiene el código que manipula la copia de Wikipedia para generar: el repositorio de menciones, el índice de artículos y el grafo de entidades. El repositorio de menciones contiene todas las anclas usadas en Wikipedia para intercomunicar los artículos. Para cada mención, el índice de menciones contiene la probabilidad del enlace y la lista de entidades que pueden ser representadas por la mención. El índice de artículos es un índice con múltiples campos, construido con Lucene[10]. El grafo de entidades persiste las conexiones entre entidades. La clase "ShingleExtractor" produce una lista de posibles menciones a partir de un documento dado (extrae todos los n-gramas posibles, con n de 1 a 6). La clase "Spotter" asocia cada fragmento con una lista de entidades candidatas (si es que existe alguna) usando el repositorio de menciones. Finalmente, el "Tagger" selecciona la mejor entidad para cada mención a partir de la lista producida por el "Spotter". Los métodos implementados en Dexter son los de WikiMiner [11], TAGME y el descrito en (Kulkarni y otros, 2009).

Finalmente, TAGME está basada en (Ferragina y Scaiella, 2010), de los resultados presentados en este trabajo estamos más interesados en la comparación en cuanto a efectividad del algoritmo presentado contra el de Milne sobre textos cortos. El conjunto de datos utilizado, denominado WIKIDISAMB30, consiste de 1.4 millones de pequeños fragmentos de texto extraídos aleatoriamente de Wikipedia. Cada fragmento es de aproximadamente 30 palabras.

El primer experimento compara la etapa de desambiguación entre el algoritmo de TAGME y el de Milne sobre WIKIDISAMB30, dividiendo este conjunto de datos en dos partes: un conjunto de entrenamiento consistente de 400000 enlaces candidatos y el resto de 1 millón. Al hacer varias pasadas y comparar por validación cruzada, obtuvieron:

[9] `http://tagme.di.unipi.it/`
[10] `http://lucene.apache.org/core/`
[11] `http://wikipedia-miner.cms.waikato.ac.nz/`

	Precisión	Recall	F-measure
Milne	92.3	84.6	88.3
TAGME	91.5	90.9	91.2

Se observa que hay una mejoría del Recall de 6.5 %, una ligera disminución de la precisión de 0.8 % y en general una mejora de la F-medida de aproximadamente 3 %.

El segundo conjunto de datos, denominado Wiki-Annot30, consiste de 150000 fragmentos, construido de la siguiente manera; usualmente en Wikipedia los autores solo enlazan la primera ocurrencia de un ancla a en un artículo z, por tanto Wiki-Annot30 es expandido al añadir un enlace para cada ocurrencia de a en z. Después de esta expansión, Wiki-Annot30 contiene casi 1.5 millones de ocurrencias de candidatos, de los cuales el 47 % tiene enlaces. Análogamente, para comparar el proceso entero de enlazado, se divide Wiki-Annot30 en 50000 fragmentos de textos pequeños de entrenamiento, y los 100000 restantes de prueba, obteniendo como resultado:

	Precisión	Recall	F-measure
Milne	69.32	69.52	69.42
TAGME	76.27	76.08	76.17

Entre las razones principales por las que el algoritmo de TAGME presenta resultados superiores al de Milne sobre textos cortos, está que Milne y Witten consideraron características que no son muy efectivas para textos cortos, como el lugar y la frecuencia de las menciones, y también que solo tomaron en cuenta menciones no ambiguas, a diferencia de TAGME que si las considera.

Una vez repasado el estado de la cuestión, se observa que aún persisten problemas abiertos en el enlazado de tuits. A causa del contexto limitado de desambiguación de los tuits, la efectividad del proceso de enlazado puede ser mejorada aún más, y una posible manera de acometer esta tarea, es tratar de explotar las características propias de los tuits, tema central de la propuesta de este trabajo.

Capítulo 3

Propuesta

En este capítulo se presenta la propuesta de un pipeline de enlazado de tuits con entidades. Primeramente, se describen las tecnologías de cf́odigo abierto empleadas, y luego se describe en detalle el enfoque utilizado para el enlazado de tuits con entidades.

3.1. Tecnologías y herramientas utilizadas

3.1.1. TwitIE

La plataforma para NLP de código abierto GATE (Cunningham et al, 2013) incluye el módulo de pipeline de propósito general para extracción de información ANNIE (Cunningham et al.,2002). ANNIE consiste de los siguientes módulos: tokenizador, segmentador de oraciones, etiquetador POS, lista de gazetteers, transductor de estados finitos, calculador de distancia ortográfica y un analizador de coreferencia.

Los componentes de ANNIE pueden ser utilizados individualmente o en conjunto con nuevos módulos para crear nuevas aplicaciones. TwitIE re-usa los módulos de partición de oraciones (que en la mayoría de los casos toma el tuit como una sola oración) y el gazetteer de nombres sin modificarlos, y los demás componentes han sido re-entrenados y adaptados para el género de Twitter.

Las listas gazetteer de ANNIE han sido re-usadas por su generalidad (nombres de países, días de la semana, meses, nombres de personas). Pero el módulo de etiquetado POS viene con listas adaptadas para etiquetar nombres no ambiguos como YouTube, Twitter, Yandex (Derczynski et al, 2013b). El resto de los componentes de TwitIE han sido adaptados, a causa de las características del género, el ruido, la brevedad, lenguaje idiosincrático y el contexto social. TwitIE ha sido diseñado específicamente para lidiar con estos problemas. En la figura se muestra el pipeline completo de TwitIE con sus componentes.

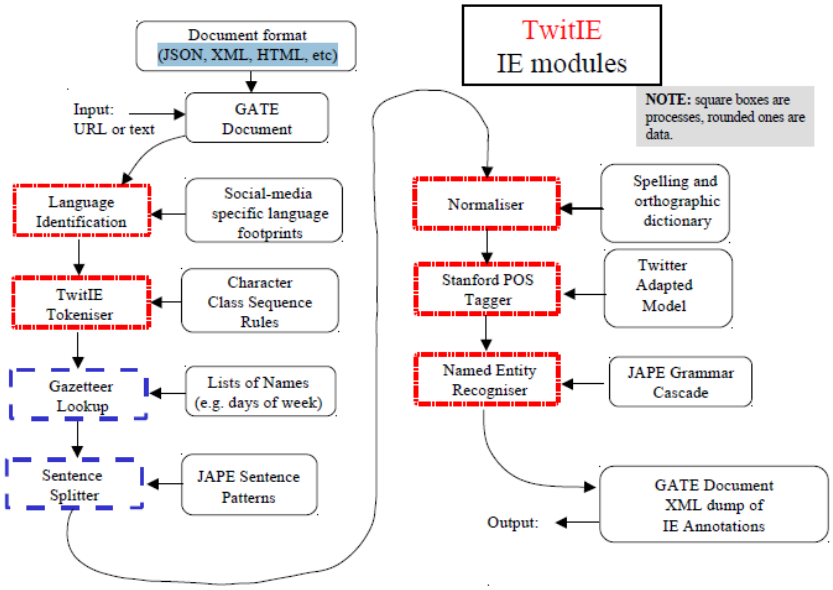


Figura 3.1: Pipeline completo de TwitIE con sus componentes

Los componentes re-usados de ANNIE se muestran enmarcados por líneas discontinuas, y los nuevos por las líneas rojas. El primer paso es la identificación del lenguaje, seguida del tokenizador, luego el gazetteer consiste de listas de ciudades, organizaciones, días de la semana, etc. No solo consiste de entidades, sino también de nombres e indicadores útiles como abreviaturas ampliamente utilizadas (por ejemplo .Ltd, títulos, etc.). Las listas de gazetteer son compiladas en autómatas de estado finito. El segmentador de oraciones de ANNIE se utiliza sin modificación, aunque se pueden procesar todo los tuits como una sola oración, sin más análisis. Luego, están los módulos de normalización, que serán descritos más adelante, el etiquetador POS (no usado en este trabajo) y el reconocedor de entidades nombradas (no usado tampoco).

Tokenización

Los tipos de tokens comúnmente descritos son los números, símbolos (como $, %), signos de puntuaciones y palabras de diferente índole. Tokenizar texto correcto es generalmente una tarea simple y reusable, ya que suele ser independiente del contexto. Sin embargo, estos tokenizadores generales tienen que ser adaptados para trabajar correctamente sobre texto como el de Twitter, para reconocer tokens específicos como URLs, hashtags (#TGIF), menciones de usuarios (@UNED), abreviaturas especiales (como RT, ROFL), y emoticonos. Un estudio de 1.1 millón de tuits estableció que el 26 % de los tuits en inglés contiene al menos una URL, 16.6 % un hashtag, y 54.8 % una mención de

usuario (Carter et al, 2013).

Estos elementos dificultan el trabajo de las herramientas de NLP convencionales (Derczynski et al, 2013a). Por lo tanto, es importante tokenizarlos correctamente.

Específicamente, el tokenizador de TwitIE trata las abreviaturas propias de Twitter (como RT, ROFL) y las URLs como un token cada una. Los hashtags y las menciones de usuarios son dos tokens con una anotación de "HashTag" que cubre a ambos. La normalización y los emoticonos son tratados por módulos opcionales independientes, por lo que la tokenización es rápida y genérica.

El tokenizador extrae los tokens de tipo hashtag, menciones, etc. que estén definidos correctamente, busca en el gazetteer específico a Twitter por entradas definidas como #apple y #sony, y si las encuentra las anota. Sino, al dividir el hashtag en dos tokens, '#' y la palabra, crea una anotación de tipo "Hashtag" con los dos tokens, si el token palabra tiene mayúsculas y minúsculas, aplica una heurística donde asume que la palabra está en CamelCase, y la divide donde quiera que se cambie de minúscula a mayúscula.

Normalización

El normalizador de TwitIE consiste en una combinación de un diccionario corrector ortográfico genérico y uno específico a micro textos. Este último contiene entradas como "2moro" y "brb", similarmente al de (Han et al, 2012). En vez de usar una lista constante de variaciones, es también posible usar una heurística para sugerir la forma correcta. La distancia ortográfica y la fonética pueden ser utilizadas para encontrar candidatas correctas para las palabras identificadas como malformadas. El algoritmo de este componente es el siguiente: dado el término de entrada, ver si tiene una conversión directa en los diccionarios, si hay una entrada, retornar ese candidato; ver si hay algún candidato con distancia ortográfica menor o igual que dos (parámetro ajustable, se usa la distancia de Levenshtein), si existe, devolver el más cercano; ver si hay algún candidato con distancia fonética menor o igual que dos (parámetro ajustable, usando el algoritmo del Doble Metáfono presentado por P. Lawrence en [Lawrence, 2000]); si no ha encontrado ningún candidato, devuelve el término original.

3.1.2. TagDef

Tagdef.com es el mayor diccionario existente de definiciones de hashtags y cuenta con un API que permite acceder a estas definiciones. El contenido es generado por usuarios, por "crowdsourcing", y las posibles definiciones para un hashtag son ordenadas por popularidad por votación. El directorio contiene más de 60.000 definiciones. Un hashtag puede tener ninguna, una o muchas definiciones ordenadas por los votos de los usuarios. Cualquiera puede registrar una definición nueva o votar positiva o negativamente por cierta definición.

El API está abierto para el uso público gratuito, permitiendo a aplicaciones clientes obtener definiciones para hashtags determinados. El formato devuelto

por defecto es XML, para obtener la respuesta del pedido en formato JSON, hay que añadir .json a la *url*. El API contiene dos formas de recurso:

- `http://api.tagdef.com/etiqueta`

  Devuelve una lista de definiciones ordenadas para un hashtag dado. Donde etiqueta es el nombre del hashtag, o sea, sin el prefijo #. Esta url devuelve una lista (puede estar vacía) de hasta cinco definiciones, ordenadas descendentemente por popularidad. Por ejemplo,

`http://api.tagdef.com/ff` para la lista en formato XML

`http://api.tagdef.com/ff.json` la misma lista, pero en formato JSON, por ejemplo para #ff:

```
{ "num_defs": "50",
"defs": [
    {"def": {
        "text": "\#ff is the same as (short for) \#followfriday.",
        "time": "2009-05-08 00:00:00",
        "upvotes": "2058",
        "downvotes": "1441",
        "uri": "http:\/\/tagdef.com\/ff",
        "hashtag": "ff"
       }
            },
         {"def": {
                 "text": "Every friday you can use \#followfriday
                          (\#FF) to suggest people to follow.",
                 "time": "2010-05-28 17:04:37",
                 "upvotes": "989",
                 "downvotes": "517",
                 "uri": "http:\/\/tagdef.com\/ff",
                "hashtag": "ff"
                }
              },

  ]
}
```

Para obtener la definición más votada:

- `http://api.tagdef.com/one.etiqueta`

Devuelve la definición más popular (la más votada) para un hashtag.

`http://api.tagdef.com/one.ff.json` para la entrada en formato JSON

`http://api.tagdef.com/one.ff` la misma entrada en formato XML

si la etiqueta existe en el repositorio de Tagdef, el código HTTP devuelto es 200 OK, con el contenido en el siguiente formato:

```
<?xml version="1.0" encoding="UTF-8"?>
<defs>
    <def>
        <text>
            <![CDATA[\#ff is the same as (short for)
               \#followfriday.]]>
        </text>
        <time>2009-05-08 00:00:00</time>
        <upvotes>2058</upvotes>
        <downvotes>1441</downvotes>
        <uri>http://tagdef.com/ff</uri>
    </def>
</defs>
```

Si la etiqueta no se encuentra, el código HTTP retornado es 404, y el contenido solo consta de la URI a la página con las posibles definiciones de la etiqueta:

```
<?xml version="1.0" encoding="UTF-8"?>
<defs>
    <uri>http://tagdef.com/noexistente</uri>
    <def>
        <uri>http://tagdef.com/noexistente</uri>
    </def>
</defs>
```

Otras capacidades que posee Tagdef, que la distinguen de APIs parecidos (como Tagboard[1] y la extinta Tagalus[2], incluyen:

- el uso de funciones callback en el formato JSON; como el objeto retornado es Javascript, puede ser usado directamente en la función callback definida por el usuario:

`http://api.tagdef.com/one.tagname.json?callback=funcion_callback`

```
funcion_callback(...);
```

donde funcion_callback es el nombre de la función a ejecutar.

[1] `https://tagboard.com/`
[2] `http://www.twtbase.com/tagalus/`

- posibilidad de filtrado por idiomas, añadiendo lang=código-del-lenguaje a la URL:

`http://api.tagdef.com/ff.json?lang=en` para definiciones solo en inglés

`http://api.tagdef.com/ff.json?lang=es` para definiciones solo en español

- en - inglés
- es - español
- pt - portugués
- fr - francés
- de - alemán
- id - indonesio
- it - italiano
- nl - holandés
- int - resto de idiomas

- acceso seguro por https, el API puede ser accedido también a través de `http://api.tagdef.com`

3.1.3. Dexter

Dexter es una plataforma de código abierto para el enlazado de entidades y fue creada con el objetivo de fomentar el desarrollo y la comparación de nuevas técnicas para el enlazado de entidades, dado que la gran mayoría de algoritmos propuestos no ofrecen el código o algún API para uso público. La plataforma implementa algunos algoritmos populares y provee toda las herramientas necesarias para desarrollar un algoritmo de enlazado de entidades.

Esta plataforma es escogida para este trabajo ya que:

- Es de código abierto
- Fácilmente extensible, está diseñada para ser ampliada con nuevos algoritmos
- Tiene una arquitectura modular
- Tiene documentación disponible

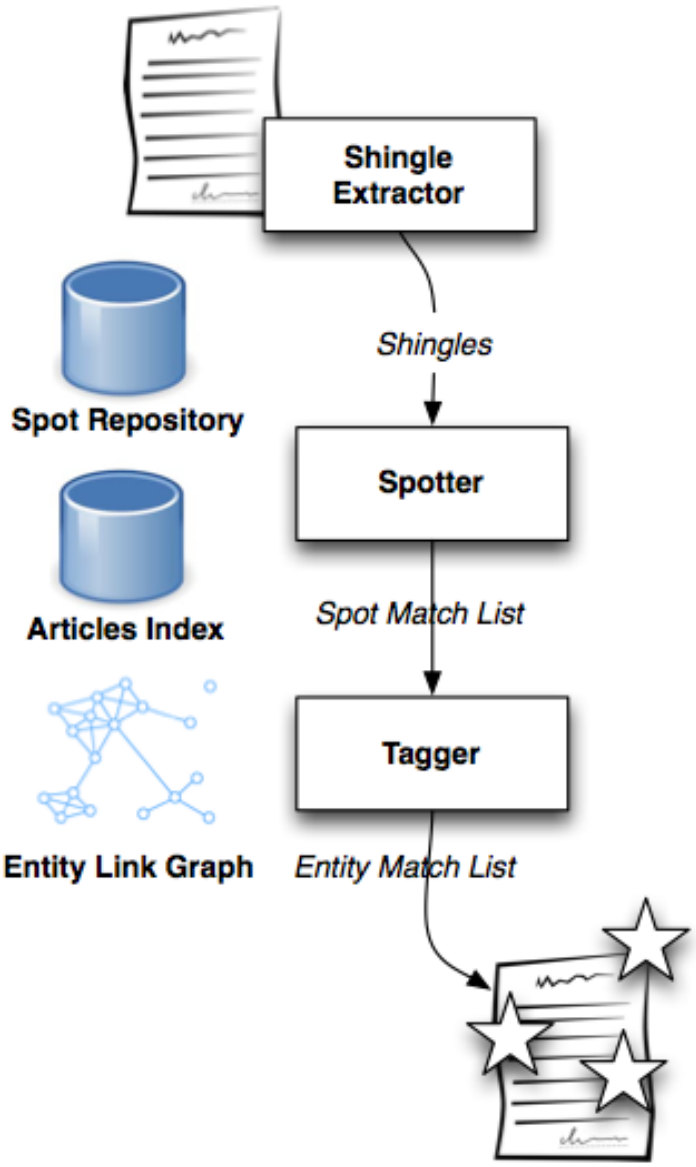


Figura 3.2: Pipeline general de un proceso de enlazado de entidades diseñado con ayuda de Dexter

- Excelente comunicación con uno de los autores de la herramienta (Diego Ceccarelli), lo cual permitió centrarme en los problemas que trata resolver la propuesta y no perder mucho tiempo afinando módulos no relacionados directamente con este trabajo, pero necesarios para la ejecución de la

herramienta.

Dexter es desarrollado en el laboratorio de alto rendimiento computacional (HPC Lab), parte del "Istituto di Scienza e Tecnologie dell'Informazione A. Faedo" ISTI[3], el instituto más grande del Consejo Nacional de la Investigación (CNR) en Italia. El área de interés del grupo de investigación del HPC Lab abarca algoritmos y sistemas de alto rendimiento computacional, diseñados para trabajar con grandes cantidades de datos y aplicados a problemas del ámbito científico, social y financiero.

Dexter puede ser usado de dos maneras diferentes:

- Usando el API como un WebService de tipo REST, después de haber descargado el archivo .jar y sus recursos. La aplicación web tiene dos interfaces, una orientada a desarrolladores, con una lista de los métodos disponibles y su descripción, los cuales se pueden ejecutar in situ desde la misma página:

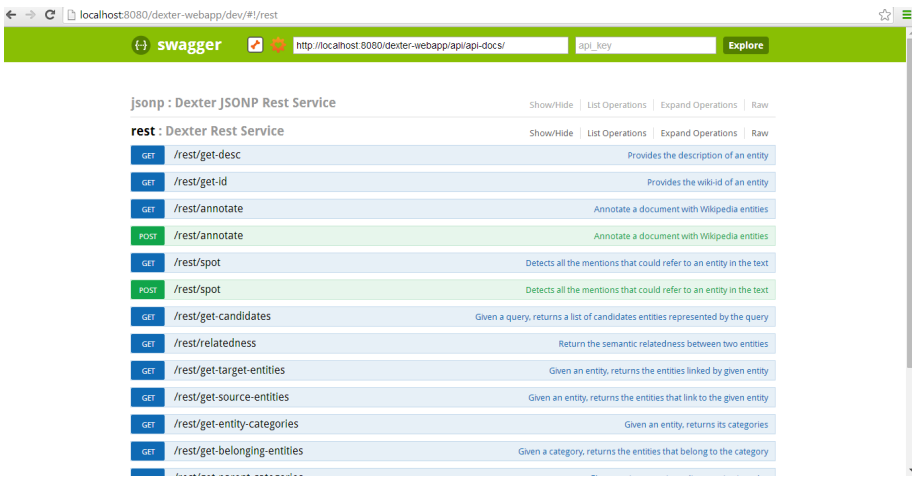


Figura 3.3: Dexter usando el API como un WebService de tipo REST

La segunda interfaz es la de anotación en sí, para ser utilizada directamente por el usuario:

- Usando el cliente en Java

  El modulo más importante de Dexter es dexter-core, que es donde están implementadas las clases principales.

Dexter-Core

[3] `http://www.isti.cnr.it/`

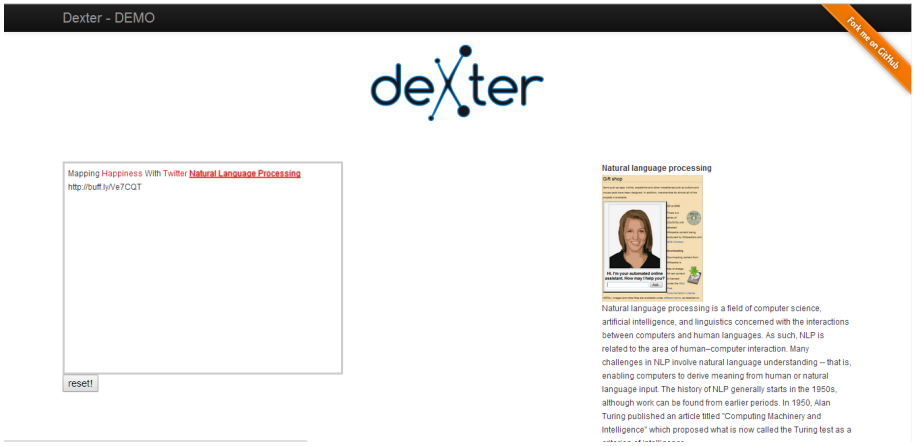


Figura 3.4: Interfaz de Dexter

Este módulo implementa el pipeline para generar el modelo básico de enlazado de entidades a partir de una colección extraída de Wikipedia. También provee las clases necesarias para desarrollar un método de enlazado de entidades adaptable.

Las clases más importantes son:

- La clase **Spot**, la cual representa una mención a una o más entidades candidatas;
- La clase **Entity**, la cual representa una entidad;
- La clase **SpotMatch**, la cual representa una mención particular en el texto dado (o sea, está atada a una palabra con una posición específica en el texto),
- La clase **EntityMatch**, la cual representa un par en específico en el documento de mención entidad candidata.

Además, define otras interfaces importantes para realizar el enlazado:

- La interfaz **Spotter** (detector de menciones) la cual define el método spot, la cual dado un texto devuelve una lista de SpotMatches (ver Spotmatch arriba).
- La interfaz **Disambiguator**, la cual dada una lista de **SpotMatches** devuelve una lista de **EntityMatch**, o sea, dada una lista de menciones con sus respectivas listas de entidades candidatas, devuelve una lista de pares, mención entidad, ya desambiguados.

Es posible diseñar nuevos detectores de menciones o desambiguadores y usarlos en Dexter, poniendo los archivos jar resultantes en la carpeta dexter/libs, o

dentro de la carpeta dexter-webapp/src/main/webapp/WEBINF/lib, y luego seleccionándolos en el archivo de propiedades project.properties, por ejemplo,

```
disambiguator.class=it.cnr.isti.hpc.wikiminer.Wikiminer
```

Por defecto, Dexter viene con un detector de menciones (basado en un diccionario con todos los textos presentes como enlace a todas las páginas de Wikipedia) y un desambiguador simple, el cual implementa el principio de Okkam, resolviendo la ambigedad de una mención escogiendo la entidad candidata con la mayor probabilidad de estar representada por la mención (esta probabilidad es llamada "commonness"), esta probabilidad da una medida de cuan representativa es la mención de la entidad y de cuan comúnmente aparece representando esta entidad y está definida como la razón entre la cantidad de enlaces que apuntan a la entidad en Wikipedia (usando la mención como texto del enlace) y el número total de enlaces que tienen la mención como texto del enlace. Esta medida será descrita en más detalle en la siguiente sección, donde se discute la propuesta de este trabajo.

3.2. Descripción de la propuesta

El enriquecimiento de textos con información relevante es hoy una técnica fundamental para mejorar la calidad de muchas actividades de análisis de textos. En este trabajo específicamente se aborda la tarea del enriquecimiento de textos vinculando conceptos y frases dentro de un texto con datos y recursos disponibles en algún repositorio de información. Estos elementos textuales, denominados menciones, conforman el punto de partida para la búsqueda y selección de recursos y representan los enlaces a la información encontrada.

3.2.1. Notación y terminología

Recordemos las definiciones de entidad y mención utilizadas.

Entidad: Definimos como entidad una página en Wikipedia no ambigua y terminal (o sea, una página que no es una categoría, lista de desambiguación o página de redirección). Nótese que una entidad puede estar representada por enlaces con texto diferente (variaciones).

Mención: Definimos como mención de entidad, o mención, al texto del enlace a la entidad y su contexto, donde su significado está definido inequívocamente por una entidad específica.

A causa de la polisemia y la posibilidad de existencia de diferentes nombres para una misma entidad, dado un fragmento de texto a, se denota como $E(a)$ al conjunto de todas las entidades referidas por a como posible mención, $freq(a)$ al número de veces que a ocurre en Wikipedia (como mención o no); mientras que

$link(a)$ denota al número de veces que a ocurre como mención a una entidad (por lo que $link(a) \leq freq(a)$).

Además se usa $pl(a) = link(a)/freq(a)$ para definir la probabilidad de que una ocurrencia de a haya sido utilizada como mención (o sea, como texto de un enlace a una entidad); y $Pr(e|a)$ denota la probabilidad a priori (también llamada "commonness", (Milne et al, 2008, Ceccarelli et al, 2013) de que una ocurrencia de a sea una mención a una entidad especifica e, donde e pertenece a $E(a)$.

En la variante canónica, este proceso consta de tres fases: (i) detección de posibles menciones en el texto de entrada, con sus respectivos conjuntos de entidades candidatas; (ii) desambiguación de estas menciones, donde cada mención termina enlazada a la entidad más relevante; (iii) poda de menciones, donde se descartan aquellas menciones y sus entidades, donde la anotación no es considerada correcta o consistente con el texto de entrada.

3.2.2. Propuesta para enlazado sobre tuits

La mayoría de los algoritmos tradicionales de enlazado de entidades asumen que el texto de entrada no tiene mucho ruido, es gramáticamente correcto y provee suficiente contexto para lograr una identificación de las entidades. Sin embargo, los tuits son textos cortos, con ruido y con abundantes abreviaturas y otros elementos que no se rigen por las reglas de ortografía, proveyendo un contexto muy limitado para las palabras que contienen.

Una de las maneras de combatir el problema del ruido en los tuits, y la utilizada en este trabajo, es la normalización léxica. Por lo tanto, en la propuesta presentada en este trabajo añadimos al proceso canónico mencionado anteriormente una fase inicial de pre-procesamiento, normalizando el tuit, usando como herramienta de normalización TwitIE, sistema adaptado especialmente para tuits y fácilmente integrable.

Para combatir el problema del contexto limitado en los tuits, es introducida una fase de enriquecimiento del texto del tuit. El objetivo de esta fase es, a partir de recursos externos, añadir información adicional al tuit, para ofrecer un contexto más amplio a la fase de desambiguación posteriormente. Más específicamente, en esta fase se hace uso de dos etiquetas especiales de Twitter, los hashtags (#) y las URLs, basándose en el hecho de que el 26 % de los tuits en inglés contiene al menos una URL y el 16.6 % un hashtag (Carter et al, 2013).

Las fases más importantes de la propuesta son las de detección de entidades, que está orientada a la cobertura, teniendo como objetivo producir una lista de listas ranqueadas de entidades candidatas; y la siguiente fase de desambiguación de entidades, orientada a la precisión, donde se determinan cuales entidades candidatas son escogidas para la anotación. Además, tenemos dos fases intermedias de filtrado, cuyo objetivo es podar menciones incorrectas o de poca significancia semántica, mejorando la precisión del algoritmo.

3.2.3. Arquitectura general de la propuesta

La propuesta presentada en este trabajo tiene una arquitectura de pipeline y cuenta con las fases de normalización → enriquecimiento → detección de menciones → filtrado de menciones → desambiguación de entidades → filtrado de entidades. En la figura 3.5 se muestra cómo funciona el pipeline sobre un tuit:

A continuación se describe cada fase con más detalle

Normalización del tuit

Para realizar la normalización del tuit, se utiliza TwitIE, en específico los siguientes componentes: Twitter Tokenizer, Hashtag Tokenizer, ANNIE Gazzeteer, Twitter Normalizer, Twitter Stanford POS tagger (no se usa directamente, pero su presencia es un requisito para el siguiente componente, el transductor), ANNIE NE Twitter Transducer.

Los tipos de tokens comúnmente descritos son los números, símbolos (como $, %), signos de puntuaciones y palabras de diferente índole. Todos los tokens de tipo palabra, son extraídos y añadidos a un conjunto (ordenado por el offset del token) de tokens. El tokenizador además extrae los tokens de tipo hashtag, menciones de usuario(@) y URLs.

Para los hashtags que estén definidos correctamente, busca en el gazetteer específico a Twitter por entradas pre-definidas y si las encuentra las anota. Sino, al dividir el hashtag en dos tokens, '#' y la palabra, crea una anotación de tipo "Hashtag" con los dos tokens, si el token palabra tiene mayúsculas y minúsculas, aplica una heurística donde asume que la palabra está en CamelCase, y la divide donde quiera que se cambie de minúscula a mayúscula. Por ejemplo, para el hashtag #GodIsGod, se obtienen cinco tokens: uno de tipo puntuación (#), tres de tipo palabra (God, Is, God), y uno de tipo HashTag (GodIsGod).

Nuevamente, se añaden al conjunto de tokens los tokens de tipo palabra, los de tipo HashTag, los de tipo URL y los de tipo mención de usuario. Por cada token del conjunto de tokens extraído, si su tipo es palabra, se normaliza el texto, usando implícitamente el componente Twitter Normalizer.

Enriquecimiento del tuit

El propósito de esta fase es aumentar el contexto limitado que ofrece el texto de un tuit típico. La idea es explotar el alto número de hashtags y URLs que tienen los tuits.

Para cada token de tipo URL, se hace un llamado a la página referida por el enlace, y se aumenta el texto con el título de esta. Nótese que generalmente los URLs en los tuits son reducidos por medio de servicios como bit.ly, para ahorrar caracteres, por lo que el URL en si no tiene ningún sentido semántico.

Para cada token de tipo Hashtag, se hace un llamado al API de Tagdef, el cual devuelve la definición más votada del hashtag, siempre que este esté presente en el diccionario de Tagdef. A diferencia de los URLs, esta información no es determinista, y puede reducir la efectividad del algoritmo, ya que la definición del

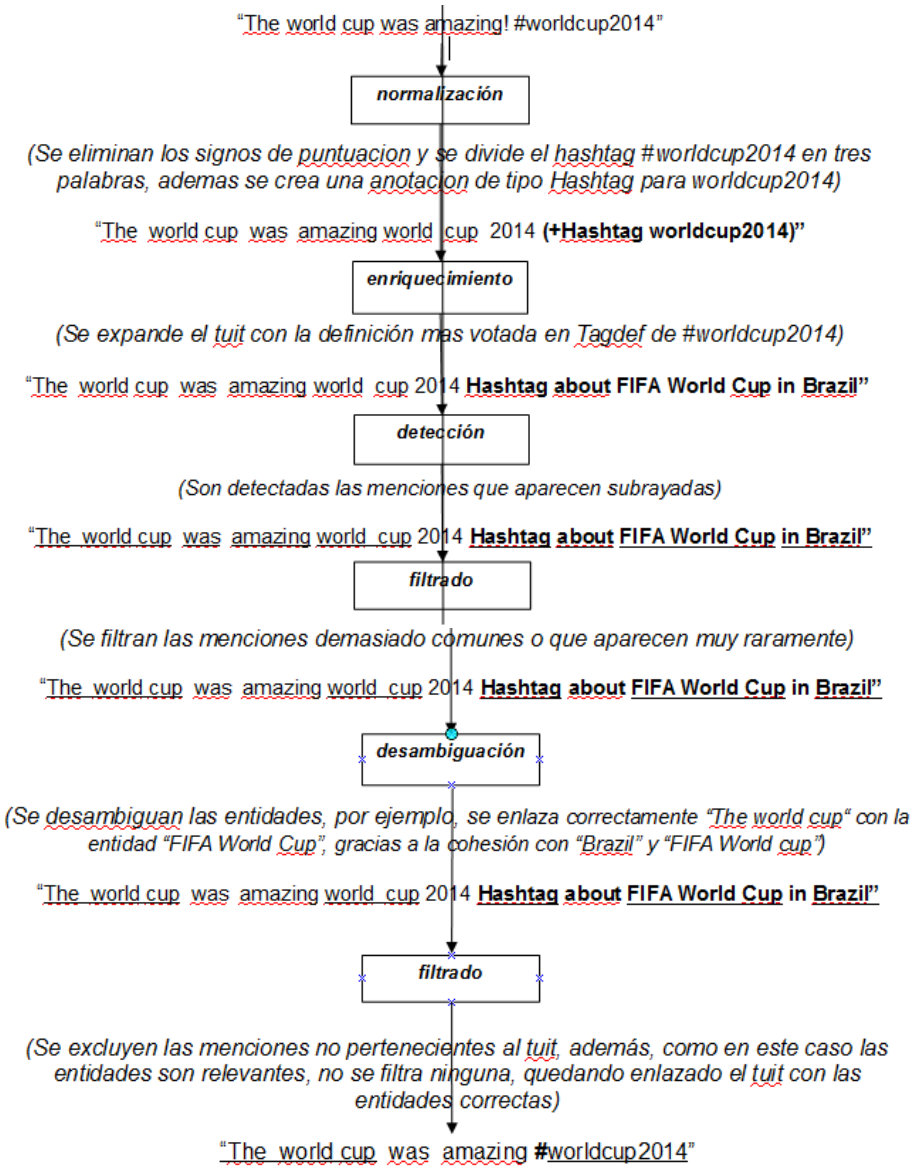


Figura 3.5: Arquitectura de la propuesta

hashtag proviene de factores externos, y aun si la definición correcta está compilada en Tagdef, puede no ser la más votada. Veremos la valía de esta fase en el capítulo de experimentación.

Detección de menciones

El conjunto de posibles menciones que pueden ser reconocidas por el algoritmo se deriva del pre-procesamiento de Wikipedia. Al utilizar Dexter como plataforma base, podemos reutilizar la clase SpotRepository, que representa un modelo extraído de Wikipedia con todos los textos de enlaces a páginas de Wikipedia y los títulos de las páginas, excluyendo los textos compuestos por un solo carácter o solo de números. Las diferentes posibles entidades son construidas al tomar todas las páginas de Wikipedia y descartando luego las páginas de desambiguación, las páginas de lista y las páginas de redirección.

Por cada mención candidata a, el repositorio contiene la probabilidad de esta de ser un enlace ($pl(a)$) y la lista de posibles entidades a las cuales puede referirse.

El diccionario resultante es indexado por Lucene para garantizar búsquedas eficientes. El grafo de entidades almacena las conexiones entre las entidades. El objetivo de esta fase es maximizar la cobertura; a partir del tuit se extraen todos los posibles n-gramas de términos, con n entre uno y seis (valor pre-definido de Dexter, en Meij et al, 2011, utilizan n=3; es extremadamente raro encontrarse una mención a una entidad en un texto tan corto como un tuit en forma de un n-grama de más de seis términos), asociándose a cada fragmento de texto una lista de entidades candidatas (si existe alguna) usando el repositorio de menciones. Cada lista de entidades candidatas está ordenada de acuerdo a la probabilidad $Pr(e|a)$ (también conocida como *commonness*).

Filtrado de menciones

Para hacer el algoritmo más eficiente, se descartan antes de hacer la desambiguación todas las entidades de $E(a)$ con commonness menor que cierto u donde u es un umbral predefinido. Se puede observar, que este parámetro puede servir de control entre cobertura/precisión, si se aumenta, aumenta la precisión, pero disminuye la cobertura, si es disminuido, el efecto es contrario. Los valores utilizados para los parámetros son descritos en la siguiente sección.

Desambiguación de entidades

Las posibles menciones resultantes del paso anterior con sus respectivas entidades candidatas, son procesadas durante la fase de desambiguación. La anotación de un texto a con una entidad e, donde e pertenece a $E(a)$ es denotada por $a \rightarrow e$. A menudo, a tiene varios significados, por lo que $|E(a)|$ ¿1, y se le llama desambiguación al proceso de seleccionar uno de los posibles significados de a, a partir de $E(a)$.

Para cada posible mención, el algoritmo de desambiguación devuelve una entidad con un peso asociado. Este peso resultante puede ser usado para seleccionar las entidades con las que anotar el texto, y para regular la proporción entre cobertura y precisión.

La fase de desambiguación tiene dos pasos. Primero, solo las posibles menciones relevantes deben ser seleccionadas. Segundo, debe ser escogida la mejor entidad candidata por cada mención. Usualmente esto se realiza considerando el contexto circundante a la mención y maximizando alguna medida de interrelación entre las entidades candidatas entre diferentes menciones (Cucerzan et al 2007,

Ferragina et a l 2010, Milne et al 2008a, Shen et al 2012).

Por tanto, la medida de interrelación escogida es uno de los puntos más importantes para la eficacia de un algoritmo de enlazado de entidades.

En concreto, Sea MT el conjunto de todas las posibles menciones que aparecen en un texto T, el algoritmo trata desambiguar cada n-grama a perteneciente a MT, calculando un peso para cada entidad candidata ea, donde ea pertenece a $E(a)$, a través de una relación entre ea y todas las entidades candidatas de los demás n-gramas de MT. Para este propósito, el algoritmo utiliza un esquema de votación (similarmente a TagMe) el cual calcula por cada n-grama b, donde b pertenece a MT {a}, su voto a la anotación $a \rightarrow ea$. Sea $rel(e1, e2)$ una medida de interrelación entre dos entidades $e1$ y $e2$, ya que b puede tener muchas entidades candidatas (o sea, $|E(b)| \geq 1$), este voto es calculado como el promedio de $rel(eb, ea)$ para cada entidad candidata eb de b y la entidad candidata en concreto de a (ea). Además, como no todas las entidades candidatas de b tienen la misma significancia estadística, la contribución de cada eb es ponderada multiplicándola por su commoness ($Pr(eb|b)$). El peso final dado a la anotación $a \rightarrow ea$ es calculado como la suma de los votos obtenidos por cada posible mención b de MT.

En este trabajo se utiliza la medida de interrelación entre entidades empleada por Guo en su trabajo influyente "To Link or Not to Link? A Study on End-to-End Tweet Entity Linking" (Guo et al 2013). Esta medida está basada sobre la medida empleada en TagMe, con ciertas modificaciones, empleando la distancia de Jaccard entre los conjuntos de enlaces entrantes, y su efectividad ha sido probada en (Guo et al 2013, Piccino et al, 2014), en este último artículo se prueba su efectividad comparándola con la medida original utilizada en TagMe.

El cálculo de la interrelación entre dos entidades tiene dos pasos. Primero se estima aproximadamente la entidad más probable por cada candidato dadas todas entidades candidatas en el tuit. Luego, el peso resultante es calculado en base a las entidades candidatas más probables calculadas en el primer paso.

Más formalmente, en el primer paso, se calcula la medida de relevancia por cada par mención entidad:

$$Rel(ea, a|MT) = \frac{\sum_{a' \neq a} a' \sum_{ea' \in E(a')} ea' P(ea'|a') Jac(ea, ea')}{|MT|}$$

Entonces, el peso de cohesión por entidad que será utilizado para escoger las anotaciones, es calculado por la siguiente formula:

$$Pcoh(ea, a|MT) = \frac{\sum_{a' \neq a} a' P(emax(a')|a') Jac(ea, emax(a))}{|MT|}$$

donde $emax(x)$ es la entidad con mayor $Rel(ex, x|t)$ de todas las entidades en $E(x)$.

Nótese que si una mención b no tiene ambigedad, su medida de commonness $Pr(eb|b) = 1$ y $|E(b)| = 1$, por lo que se explota la no ambigüedad de las menciones no ambiguas (como en Milne et al, 2008b).

Ahora, si b tiene polisemia, solo las entidades eb relacionadas con ea van a afectar $Rel(ea, a|MT)$ a causa del uso de la medida de relación entre las entidades. Esta es la diferencia principal sobre la medida propuesta en (Milne et al, 2008a), que está basada en menciones no ambiguas solamente (las cuales pueden no estar presentes).

Como ejemplo de desambiguación veamos el siguiente tuit del dataset de prueba:

"Sneijder update: Meeting is close with Sneijder representatives, Inter and United oficials in the room, will keep you posted how it goes"

United: (Manchester United F.C., equipo de fútbol inglés), Pcoh: 0.005, prior: 0.016

United: (Sheffield United F.C., equipo de fútbol inglés), Pcoh:0.001, prior:0.012

United: (D.C. United, equipo de fútbol estadounidense), Pcoh:0.0005, prior:0.65

La fase de desambiguación enlaza correctamente "United" con la entidad "Manchester United F. C.", apoyándose en las otras menciones del tuit ("Inter", equipo de fútbol italiano, y "Sneijder", jugador de fútbol holandés), ya que Sneijder iba a ser jugador del Man. U. y por lo tanto está mucho más correlacionada como mención. Nótese que la probabilidad inicial o commonness, de "United" como el equipo de fútbol americano es mucho mayor.

Filtrado de entidades

El conjunto de anotaciones candidatas $A(MT)$ producido por la fase de desambiguación debe ser podado para descartar posibles menciones sin mucho contenido semántico. Estas menciones son detectadas por una función que solo depende de dos factores: la probabilidad $pl(a)$ de a ser una mención y la coherencia de su anotación candidata $a \rightarrow ea$ con respecto a las anotaciones candidatas de las demás menciones en $A(MT)$. La probabilidad $pl(a)$ es un atributo simple y eficiente para detectar menciones relevantes (Milne et al, 2008b). El algoritmo produce un peso $(a \rightarrow e)$ para la anotación candidata $a \rightarrow e$, el cual es a su vez comparado con un umbral u, de manera que si el peso es menor que el umbral, se desecha la anotación $a \rightarrow e$.

Entonces, para desambiguar, se seleccionan las anotaciones (pares mención - entidad) con peso mayor que un cierto umbral $\acute{\varepsilon}$, y de estas, la de mayor commonness. Este parámetro puede ser utilizado para afinar la desambiguación: un valor mayor favorecerá a los temas más comunes, mientras que uno menor hará que la desambiguación dependa más del contexto. Es empleado de la siguiente manera, dado el peso de cohesión máximo de entre las entidades de $E(a)$, se calcula el peso de cohesión mínimo para una entidad candidata necesaria para ser seleccionada como:

$minScore = maxScore - maxScore * \acute{\varepsilon}$

3.3. Parámetros utilizados

Habiendo hecho una revisión del estado del arte y basándose en la distribución de enlaces en el modelo de Wikipedia utilizado, se han prefijado los siguientes valores iniciales para los parámetros usados:

- umbral de commonness: 0.005, este parámetro permite filtrar las menciones no representativas, mejorando la eficacia de la desambiguación. Un valor más pequeño puede mejorar la cobertura, si se aumenta, puede mejorar la precisión. Por ejemplo, el texto "Michael Collins" puede referirse a más de 20 entidades, sin embargo, las cuatro más comunes son las usadas en casi el 100 % de las ocurrencias (Ceccarelli et al, 2013). Este parámetro se usa en las fases de filtrado y desambiguación.
- umbral de probabilidad de ser un enlace: 0.02 → al igual que el parámetro anterior, permite filtrar las menciones sin mucha carga semántica. Se usa como parámetro en el filtrado de menciones. Este parámetro se usa en la fase de filtrado inicial.
- epsilon: 0.3, este parámetro puede ser útil cuando se anota texto fragmentado, como los tuits, donde a causa del contexto limitado es mejor favorecer a las entidades más comunes. Este parámetro se usa en la fase de desambiguación

Capítulo 4

Descripción de la evaluación

En este capítulo se describe el marco de evaluación. Primeramente se presentan las dos colecciones de prueba utilizadas en la experimentación, a continuación se describe el algoritmo de enlazado de tuits con entidades tomado como baseline y finalmente son descritas brevemente las medidas de evaluación utilizadas en la experimentación.

4.1. Descripción de la colección de evaluación

La base para la colección de evaluación utilizada en este trabajo es la empleada en la competición "Named Entity Extraction and Linking Challenge (NEEL)"de la edición de 2014 del workshop "Making Sense of Microposts", en el marco de la conferencia WWW (World Wide Web Conference 2014), facilitada por los organizadores bajo licencia a los participantes [link], consistente en la extracción y enlazado de entidades a partir de microtextos.

La colección está formada por 3500 tuits extraídos de una colección mucho más grande de 18 millones de tuits, los cuales abarcan el periodo desde el 15 de julio de 2011 al 15 de agosto de 2011 (31 días). En los tuits aparecen eventos tales como la muerte de Amy Winehouse, los motines en Londres y el acto de terrorismo en Oslo. Los tuits que contienen eventos generalmente contienen entidades, pero la colección también contiene tuits que no cubren eventos y/o contienen entidades (un 30 % de los tuits no contiene entidades).

Para el enlazado, se utiliza como repositorio de entidades la versión 3.9 de DBPedia en inglés, dado que cada entrada de este catálogo tiene una entrada equivalente en Wikipedia (ya que es compilado a partir de esta), podemos utilizar esta colección en nuestro trabajo y comparar las entidades extraídas de Wikipedia, que serán las mismas de DBPedia.

4.1.1. Conceptos básicos

Son considerados como entidades entes que pueden ser caracterizados por una instancia de una clase de una taxonomía (ver apéndice II); el objeto o conjunto de objetos no tienen por qué tener necesariamente una existencia material (competición NEEL[1]).

Una entidad puede ser referenciada por una palabra en un tuit si se cumplen las siguientes reglas (las entidades consideradas son las que se encuentran entre corchetes):

- Pertenece a una de las categorías especificadas en la taxonomía que aparece en el apéndice II.
- Puede ser desambiguada por una URI de DBpedia, teniendo como contexto el tuit. Por tanto, todas las entidades NIL (o sea, entidades sin una URI de desambiguaci ón) no son tomadas en cuenta.
- Subyuga a otras entidades. Quiere decir, que una mención a entidad, compuesta por dos o más entidades, es considerada una sola entidad si puede ser desambiguada por una URI de DBpedia. Por tanto las menciones más largas tendrán preferencia sobre menciones de una sola palabra y/o cortas, por ejemplo:
  - [Natural History Museum at Tring]
  - [News International chairman James Murdoch]'s evidence to MPs on phone hacking
  - [Sony]'s [Android Honeycomb] Tablet

En este último caso, ya que no hay una entrada en DBpedia para [[Sony]'s [Android Honeycomb]], se divide en dos entidades, [Sony] y [Android Honeycomb].

4.1.2. Caso especial. Expresiones numéricas

En la anotación original solo fueron consideradas las expresiones numéricas que cumplen las siguientes reglas:

- Se refieren a números enteros.
- Expresiones de enteros separados por un guion (por ejemplo 0-1).

Excluyendo:

- Expresiones numéricas acompañadas de un símbolo de moneda (como 20$).

[1] `http://www.scc.lancs.ac.uk/microposts2014/challenge/`

- Expresiones con decimales.
- Casos con la palabra "billion", la cual no tiene entrada en DBPedia.
- Expresiones separadas por "/" o ":".
- Expresiones compuestas por un número y una palabra (como 48h).

4.1.3. Criterios de reanotación

Para este trabajo se ha extraído, a partir de la colección original, un subconjunto de 200 tuits, los cuales fueron reanotados, dadas las muchas imprecisiones que tenía la colección original. Los principales errores de dicha colección son:

- Entidades que no estaban anotadas, como en:

  "[US] to aid famine-struck [Somalia]: The [US] says it will send aid to parts of [Somalia] controlled by [al-Shabab] if i... `http://bbc.in/ph0TP3`", falta la referencia a `http://dbpedia.org/resource/Somalia`,

  o en, "Reports of disturbances in the following areas: #Wathamstow #PalmersGreen #Enfield #PondersEnd #Earlsfield and #Westfield. #LondonRiots", donde falta la referencia a `http://dbpedia.org/resource/Palmers_Green`,

- Entidades anotadas con una entidad errónea, tal como:

  "RT @STATS_MLB #[WhiteSox]'s [Dunn] has 6 hits in 13 at-bats at [Progressive Field]. He has 17 hits in 138 at-bats at [U.S. Cellular Field]. #[Indians]",

  donde la entidad correcta es `http://dbpedia.org/resource/Cleveland_Indians` en vez de `http://dbpedia.org/resource/Indianapolis_Indians`).

En total a partir del conjunto original de 200 tuits fueron reanotadas 4 entidades que estaban incorrectamente anotadas y añadidas 26 nuevas entidades.

Tal como en la anotación original, se ignoraron las siguientes: entidades duplicadas, menciones a entidades ambiguas, y entidades no presentes en Wikipedia.

4.1.4. Etiquetas especiales propias de Twitter

Las anotaciones son en inglés y la distribución de etiquetas especiales como hashtags y menciones de usuarios es la siguiente.

Para el conjunto de evaluación original:

- 28 % de los tuits de la colección contienen al menos un hashtag.

- 26 % de los tuits de la colección contienen al menos un enlace a una página web.
- 7 % de los tuits de la colección contienen al menos una etiqueta de tipo RT.
- 29 % de los tuits de la colección contienen al menos una mención de usuario (@usuario).

Para el subconjunto reanotado:

- 32 % de los tuits de la colección contienen al menos un hashtag.
- 26 % de los tuits de la colección contienen al menos un enlace a una página web.
- 10 % de los tuits de la colección contienen al menos una etiqueta de tipo RT.
- 31 % de los tuits de la colección contienen al menos una mención de usuario (@usuario).

Compárese con la distribución global en Twitter, según el estudio de 1.1 millón de tuits realizado por Carter (Carter et al, 2013) donde estableció que el 26 % de los tuits en inglés contiene al menos una URL, 16.6 % un hashtag, y 54.8 % una mención de usuario.

Los casos particulares de hashtags y menciones de usuarios también pueden ser referenciados por entidades, como en:

"#[Obama] is proud to support the Respect for Marriage Act" "#[Barack Obama] is proud to support the Respect for Marriage Act" "@[BarackObama] is proud to support the Respect for Marriage Act"

Otro caso particular son los alias, cuando una palabra se refiere a otra entidad, para estos casos se utiliza la coreferencia con otras entidades en el contexto del tuit (excepto los pronombres personales como ellos, ella, que no se consideran como entidades):

"#[Panda] with 3 straight hits to give #[SFGiants] 61 lead in 12th"

- "#[Panda] > `http://dbpedia.org/page/Pablo_Sandoval`"
- "#[SFGiants] > `http://dbpedia.org/page/San_Francisco_Giants`"

"Knox mother hits out at 'errors': The mother of [Amanda Knox], jailed for murdering her [British flatmate] in [Italy], `http://bit.ly/nWebV1`"

> [British flatmate] > `http://dbpedia.org/resource/Murder_of_Meredith_Kercher#Meredith_Kercher`

Cabe destacar, que hay un gran grado de subjetividad al comparar colecciones anotadas para la tarea de enlazado de entidades, como observan (Guo et al 2013) al analizar la colección utilizada por (Meij et al, 2012) en la cual aparecen muchas entidades que ellos no consideran como entidades. En general muchas palabras en inglés tienen su propia entrada en Wikipedia, pero la mayoría de las palabras no son tan importantes como para ser consideradas entidades.

4.1.5. Estructura de la colección

La estructura de la colección original es la siguiente:

tuit_id

mención_entidad_1

uri_entidad_1

...

mención_entidad_n

uri_entidad_n

donde los pares (mención, uri) son opcionales. La colección contiene el id del tuit original, en vez del texto, a causa de la política de privacidad de Twitter, que no permite distribuir los tuits. Pero dado el identificador del tuit se puede acceder al tuit original, siempre y cuando este no haya sido borrado por su autor. Un ejemplo de instancia es la siguiente:

94153790718099400 -> #Visa, #Mastercard, #AMEX - and #PayPal at POS: PayPal will be in trial with a major U.S. retailer later this year (per eBay chief Donahoe). ->

Visa -> `http://dbpedia.org/resource/Visa_Inc.` ->

AMEX -> `http://dbpedia.org/resource/American_Stock_Exchange` ->

PayPal -> `http://dbpedia.org/resource/PayPal` ->

U.S. -> `http://dbpedia.org/resource/United_States` ->

eBay -> `http://dbpedia.org/resource/EBay` ->

chief -> `http://dbpedia.org/resource/Chief_officer` ->

Donahoe -> `http://dbpedia.org/resource/John_Donahoe`

4.2. Baseline: Algoritmo TagMe (implementación incluida en la plataforma Dexter)

Como baseline para los experimentos sobre la efectividad del enlazado de entidades, se usa la implementación del algoritmo TagMe incluida en la plataforma Dexter. Este algoritmo es escogido por ser el más eficaz del estado de la cuestión para enlazado de entidades sobre textos cortos, al menos de los disponibles públicamente. Sobre este algoritmo está basada la fase de desambiguación de la propuesta presentada en este trabajo, siendo uno de los objetivos mejorar su eficacia.

A continuación se describirá su funcionamiento. Sea MT el conjunto de todas las posibles menciones que aparecen en un texto T, el algoritmo trata desambiguar cada n-grama a perteneciente a MT, calculando un peso para cada entidad candidata ea, donde $ea \in E(a)$, a través de una relación entre ea y todas las entidades candidatas de los demás n-gramas de MT. Para este propósito, el algoritmo utiliza un esquema de votación el cual calcula por cada n-grama b, donde $b \in MT \setminus \{a\}$, su voto a la anotación $a \rightarrow ea$.

Sea $rel(e1, e2)$ una medida de interrelación entre dos entidades $e1$ y $e2$, ya que b puede tener muchas entidades candidatas (o sea, $|E(b)| \geq 1$), este voto es calculado como el promedio de $rel(eb, ea)$ para cada entidad candidata eb de b y la entidad candidata en concreto de $a(ea)$. Además, como no todas las entidades candidatas de b tienen la misma significancia estadística, la contribución de cada eb es ponderada multiplicándola por su commoness ($Pr(eb|b)$). Por tanto la fórmula es:

$$voteb(ea) = \frac{\sum_{eb \in E(b)} ebrel(eb,ea) x Pr(eb|b)}{|E(b)|}$$

La medida de interrelación $rel(eb, ea)$ es la propuesta en (Milne et al, 2008a), explotando la estructura de grafo de Wikipedia, pero modificada para incluir las entidades ambiguas también:

$$rel(eb, ea) = \frac{1 - \log max(|in(a), in(b)| - \log min(|in(a) \cap in(b)|}{\log |W| - \log min(|in(a), in(b)|}$$

donde W es el conjunto de todas las entidades presentes en el modelo extraído de Wikipedia, mientras que $in(a)$ e $in(b)$ son los conjuntos de las menciones que se refieren a a y a b, respectivamente. Cuando:

$|in(a) \cap in(b)| = 0, rel(ea, eb) = 0.$

Además, $rel(ea, eb)$ alcanza su máximo de 1 cuando:

$in(a) \cap in(b) = in(a) = in(b)$, y por lo tanto todos los artículos que citan a a, citan también a b, y vice versa.

Solo las entidades eb relacionadas con ea afectan $voteb(ea)$ a causa del uso de la medida de interrelación $rel(eb, ea)$. El peso final dado a la anotación $a \rightarrow ea$ es calculado como la suma de los votos obtenidos por cada posible mención b de MT.

Entonces, para desambiguar se selecciona la mejor anotación $a \rightarrow ea$, utilizando un algoritmo de ranking con umbral.

Filtrado de menciones y entidades

Al ser ejecutado el baseline sobre la misma plataforma Dexter, las fases de filtrado son las mismas. Para el filtrado de menciones la misma función que solo depende de dos factores: la probabilidad $pl(a)$ de a ser una mención y la coherencia de su anotación candidata $a \rightarrow ea$ con respecto a las anotaciones candidatas de las demás menciones en $A(MT)$.

Para el filtrado de menciones, similarmente, se seleccionan las anotaciones (pares mención - entidad) con peso mayor que un cierto umbral $\acute{\varepsilon}$, y de estas, la de mayor commonness.

Los valores iniciales usados para los parámetros son los estándares para el algoritmo, tales como se muestra en (Ceccarelli et al, 2013) e indicados en el apartado 3.3 del presente trabajo.

4.3. Medidas de evaluación

Para evaluar la eficacia de un sistema de enlazado de entidades, se considera la habilidad del mismo para detectar y enlazar una entidad a partir de un tuit. Por cada tuit t en un conjunto de tuits T, el sistema provee un conjunto de pares de la forma: texto de la mención, entidad (con el identificador a la entidad en el catálogo de Wikipedia). La evaluación consiste de comparar los pares producidos por la propuesta y las diferentes modificaciones, contra los anotados manualmente en el dataset.

Para evaluar el proceso de anotación, se utilizan las medidas estándar de cobertura de información de precisión y cobertura de entidades, siguiendo las pautas de la competición NEEL[2], se utilizan medidas de precisión (Pre) y cobertura (Rec) sobre el conjunto de menciones que aparecen exactamente en el dataset de verificación, en el mismo orden de este. Además, se utiliza la medida $F1$, la cual al combinar las medidas de precisión y cobertura, sirve como indicador veraz de la eficacia del algoritmo.

Precisión (Pre). Es la fracción de entidades relevantes obtenidas por el algoritmo de enlazado respecto al total de entidades del tuit. Dado un tuit t, se denomina precisión al número de entidades relevantes anotadas sobre T, dividido por el número total de entidades anotadas sobre t. Sean, una colección de tuits T, $A(t)$ el conjunto de entidades obtenido por el algoritmo de anotación para un tuit t y $E(t)$ el conjunto de entidades relevantes de t, la precisión se define como:

$$Pre = \frac{\sum_{t \in T} t|A(t) \cap E(t)|/|A(t)|}{|T|}$$

Cobertura (Recall, Rec). Se puede describir la cobertura como la capacidad del

[2] `http://www.scc.lancs.ac.uk/microposts2014/challenge/`

algoritmo de enlazado de encontrar el máximo número de entidades relevantes en el tuit. Dado un tuit t, se denomina cobertura al número de entidades relevantes anotadas sobre t, dividido por el número total de entidades relevantes de t. Similarmente, sean, una colección de tuits T, $A(t)$ el conjunto de entidades obtenido por el algoritmo de anotación para un tuit t y $E(t)$ el conjunto de entidades relevantes de t, la cobertura se define como:

$Recall = \frac{\sum_{t \in T} t|A(t) \cap E(t)|/|E(t)|}{|T|}$

F1. La medida $F1$ combina las medidas de precisión y cobertura, y se puede interpretar como el promedio ponderado de la precisión y la cobertura. Esta medida es calculada como el doble del producto de la precisión y la cobertura dividido entre la suma de la precisión y la cobertura:

$F1 = \frac{2 * Pre * Rec}{Pre + Rec}$

Para evaluar la correspondencia de los pares mención-entidad producidos por el sistema con los del dataset anotado de prueba, según las medidas de precisión y cobertura, se valora la correspondencia exacta entre los pares, teniendo en cuenta el orden exacto de los pares. Sean, $S(m, e)$ el conjunto de pares extraídos por el sistema, $D(m, e)$ el conjunto de pares anotados manualmente en el dataset, se definen los conjuntos de pares falsos positivos FP (pares detectados por el sistema, pero no presentes en el dataset anotado), positivos verdaderos TP (pares detectados y presentes en el dataset) y falsos negativos FN (pares del dataset no detectados) como:

$TP = \{(m, e) \in S(m, e) \cap D(m, e)\}$

$FP = \{(m, e) \in S(m, e)^{(m,e)! \in D(m,e)}\}$

$FN = \{(m, e) \in D(m, e)^{(m,e)! \in S(m,e)}\}$

A partir de estas definiciones, una manera equivalente de definir las medidas de precisión, cobertura y F1 es la siguiente:

$Pre = \frac{|TP|}{|TP \cup FP|}$

$Rec = \frac{|TP|}{|TP \cup FN|}$

$F1 = \frac{2 * Pre * Rec}{Pre + Rec}$

Una vez definidas las medidas de evaluación, podemos definir un marco comparativo entre los resultados de los experimentos sobre las colecciones de prueba (dataset original del NEEL y dataset corregido).

Capítulo 5

Experimentación

En este capítulo se describe la experimentación realizada en este trabajo. En primer lugar se describe el método empleado en la experimentación. En un segundo apartado se describen los modelos que se quieren evaluar para cuantificar su valía para el proceso de enlazado. Luego se hace una comparación de los mejores modelos obtenidos con el modelo base, ya que uno de los objetivos de la propuesta es mejorar su eficacia. Por último, se realiza el análisis de los resultados obtenidos en los experimentos y las conclusiones asociadas.

5.1. Diseño de la experimentación

En este apartado se describe el proceso utilizado para evaluar los diferentes modelos en la experimentación. Para evaluar la eficacia del sistema de enlazado de entidades, se va a evaluar la habilidad del mismo para detectar y enlazar una entidad a partir de una colección de tuits preanotados en cuanto a precisión, cobertura y medida F1, tal como se describe en la sección 4.3, tomando como base ambas colecciones de tuits de prueba descritas en la sección 4.2; primero se realizarán los experimentos sobre la colección reanotada de 200 tuits, y luego sobre la colección más amplia original de la competición NEEL de la edición de 2014 del workshop "Making Sense of Microposts".

Los experimentos evalúan las fases de detección de entidades, que está orientada a la cobertura, teniendo como objetivo producir una lista de listas ranqueadas de entidades candidatas; y la siguiente fase de desambiguación de entidades, orientada a la precisión, donde se determinan cuales entidades candidatas son escogidas para la anotación. En especial se evalúa el efecto de las modificaciones propuestas en cada modelo sobre la efectividad de detección y desambiguación de entidades. Además, en todos los experimentos son aplicadas las fases intermedias de filtrado, cuyo objetivo es podar menciones incorrectas o de poca carga semántica, mejorando la precisión del algoritmo.

5.2. Descripción de los modelos a evaluar

En este apartado se describen los modelos con las modificaciones propuestas que se quieren evaluar para cuantificar su valía para el proceso de enlazado, por separado, y en conjunto con los demás modelos.

Normalización de los tuits.

Este modelo consiste en aplicar solo la fase inicial de pre-procesamiento, normalizando el tuit, usando como herramienta de normalización específica para tuits TwitIE, sin la expansión de URLS ni hashtags. Luego para la desambiguación se utiliza el algoritmo propuesto, con la medida de similitud entre entidades y cálculo de la medida de cohesión entre estas con la modificación propuesta por Guo et al.

Expansión de hashtags presentes en tuits.

Este modelo consiste en aplicar solo la expansión de hashtags. O sea, para cada tuit, se extraen los hashtags a partir del texto sin normalizar, y luego, con ayuda del API de Tagdef, se expande cada hashtag con la definición más votada según aparece en el sitio de Tagdef. Luego para la desambiguación se utiliza el algoritmo propuesto, con la medida de cohesión de (Guo et al, 2013).

Expansión de URLs presentes en tuits.

Este modelo consiste en aplicar solo la expansión de URLs. O sea, para cada tuit, se extraen las URLs a partir del texto sin normalizar, y luego, por cada una, se hace un llamado a la página referida por el enlace, y se aumenta el texto con el título de esta. Luego para la desambiguación se utiliza el algoritmo propuesto, con la medida de cohesión de (Guo et al, 2013).

5.3. Comparación de los resultados obtenidos en la experimentación

En este apartado se presentan y analizan los resultados obtenidos en la experimentación llevada a cabo a lo largo de este trabajo de investigación. Este análisis se estructura en torno a las características del modelo consideradas y definidas en la propuesta:

5. Normalización de los tuits.
6. Expansión de hashtags presentes en tuits.
7. Expansión de URLs presentes en tuits.
8. Uso de la medida de similitud entre entidades y cálculo de la medida de cohesión entre estas, utilizando el algoritmo diseñado por Guo.

Para cada uno de los modelos propuestos, en primer lugar se muestran los resultados obtenidos con y sin la modificación del mismo, comparándose con cada uno de los demás modelos en cuanto a precisión, cobertura y medida F1 obtenida sobre el conjunto de evaluación.

En segundo lugar, se muestran y discuten los resultados obtenidos al realizar la misma experimentación sobre el conjunto de evaluación empleado en la competición NEEL de la edición de 2014 del workshop "Making Sense of Microposts".

A continuación, se muestran y discuten los resultados obtenidos al comparar el mejor modelo obtenido para cada conjunto de evaluación, contra el modelo base, descrito en la sección 4.2. Recordemos que el modelo escogido como baseline para los experimentos sobre la efectividad del enlazado de entidades es la implementación del algoritmo TagMe incluida en la plataforma Dexter.

Experimentos

Partiendo desde los valores iniciales para los parámetros usados, se evalúa un rango para cada parámetro, y las combinaciones entre estos, estimando y fijando el valor óptimo de umbral de probabilidad para la colección de prueba.

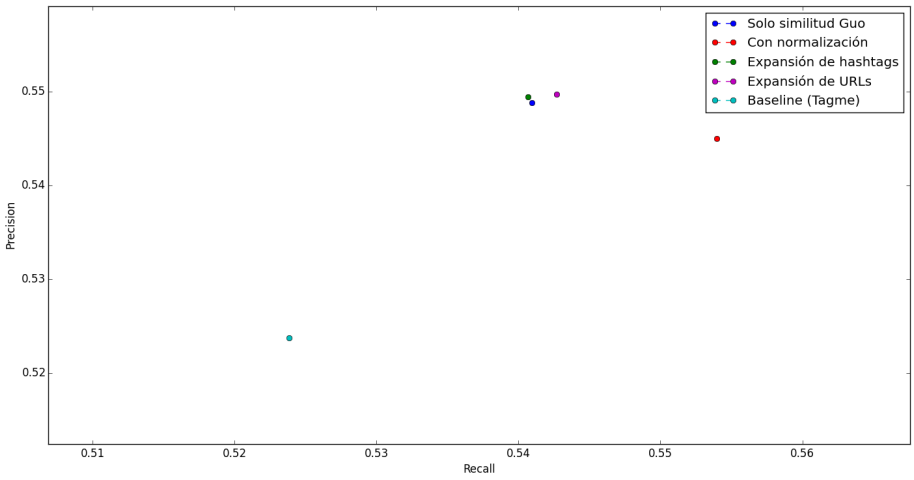
Para el umbral de probabilidad se probaron valores entre 0.001 y 0.9 (ver resultados en el apéndice III), siendo los valores óptimos: 0.03 para la colección de 200 tuits y 0.3 para la colección NEEL; para valores menores la precisión y la cobertura alcanzados sobre la colección de evaluación son menores, y al aumentar más el valor, aumenta la precisión, pero la cobertura comienza a disminuir abruptamente, disminuyendo también la medida F1, o sea, se quedan muchas entidades sin anotar.

Para los parámetros de *commonness* y *epsilon* se prueba un rango de valores. El rango de evaluación para el parámetro de *commonness* es de [0.005, 0.1, 0.3 y 0.5] para ambas colecciones. Se toma un valor pequeño inicialmente (0.005) para intentar lograr una cobertura mayor, pero el máximo de precisión/cobertura se alcanza con un valor de 0.03; este es el valor utilizado en la comparación de los modelos. El rango de evaluación para el parámetro de *epsilon* es de [0.3, 0.5, 0.7 y 0.9] para ambas colecciones. El máximo de precisión/cobertura se alcanza con un valor de 0.7; este es el valor utilizado en la comparación de los modelos.

5.3.1. Experimentos sobre la colección reanotada de 200 tuits

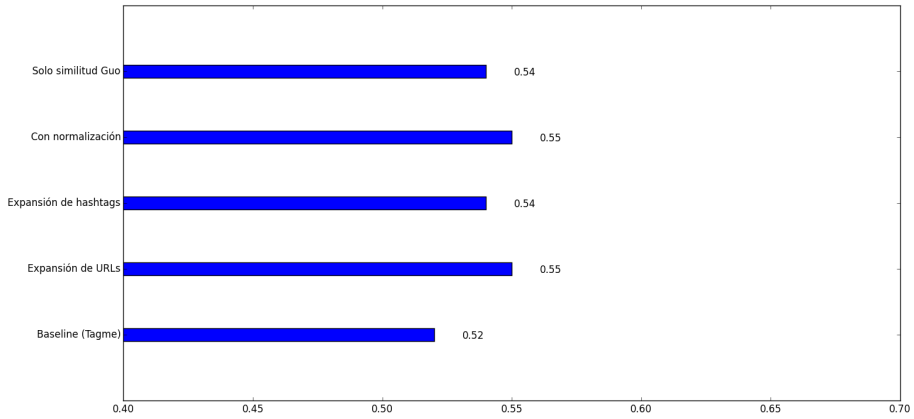
A continuación se presenta una gráfica comparando los diferentes modelos por separado en cuanto a precisión y cobertura, el valor de *commonness* escogido es de 0.03 y el de *epsilon* de 0.7 (los resultados de la evaluación de todo el rango y sus combinaciones se pueden ver en el apéndice IV):

En la gráfica se aprecia como al aplicar la normalización a los tuits, aumenta la cobertura, o sea, se identifica un número mayor de entidades. Se observa que el uso de la similitud de Guo aumenta la eficacia del algoritmo, mejorando la



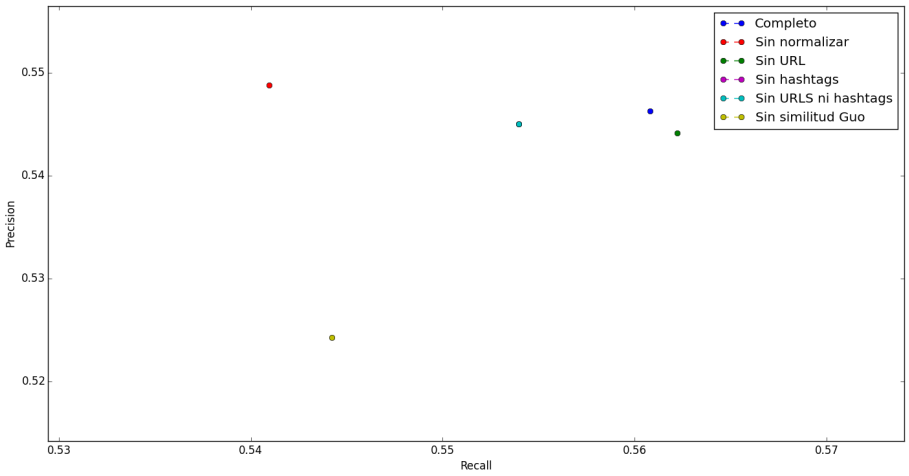
precisión y cobertura a la vez, asi como el enriquecer el contenido del tuit por medio de la expansión de las URLs, aunque en mucha menor medida.

La expansión de los hashtags no parece aportar un beneficio apreciable.



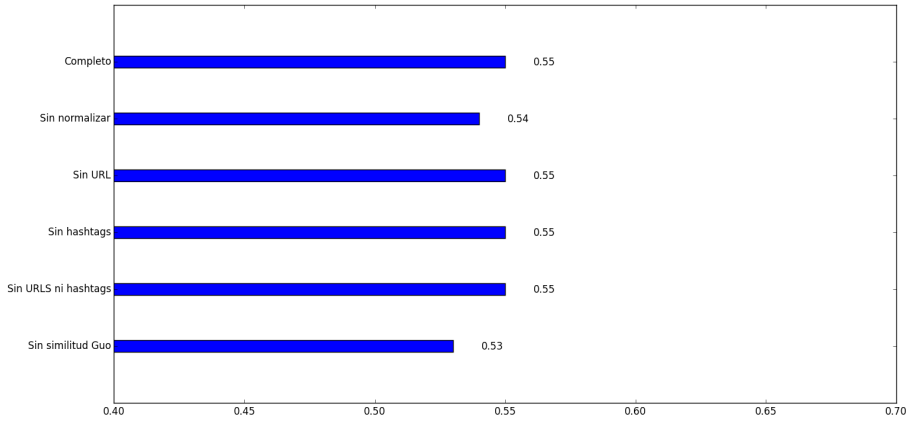
Esta gráfica muestra el valor alcanzado de la medida F1 para los mismos modelos de la figura anterior, confirmando la mejoría en términos de F1 al usar la medida de similitud de Guo, y una ligera mejoría de la efectividad del algoritmo al utilizar la normalización de tuits o la expansión de las URLs, manteniéndose la misma F1 al expandir los hashtags.

En la siguiente gráfica se muestra el efecto de suprimir cada modificación a partir de la propuesta "completa" (con normalización, expansión de hashtags y URLs y usando la similitud de Guo). El último modelo (sin similitud de Guo), utiliza la medida de similitud y el algoritmo de cálculo de la cohesión de Tagme, además de las fases de normalización y expansión de los tuits.



Se observa que al no normalizar los tuits, se pierde en cobertura. Se corrobora también, que al utilizar la medida y algoritmo de desambiguación de Tagme original, hay una pérdida de efectividad, en cuanto a precisión y cobertura. Curiosamente, al no usar la expansión de URLs, pero si la de hashtags y normalización, hay una muy pequeña mejoría en cuanto a cobertura. Al no utilizar la expansión de hashtags se observa una pequeña disminución de la cobertura (el punto que representa el modelo sin expansión de hashtags se encuentra solapado por el de sin URLs ni hashtags).

A continuación se ve la gráfica correspondiente, pero mostrando la medida F1 para los mismos modelos:



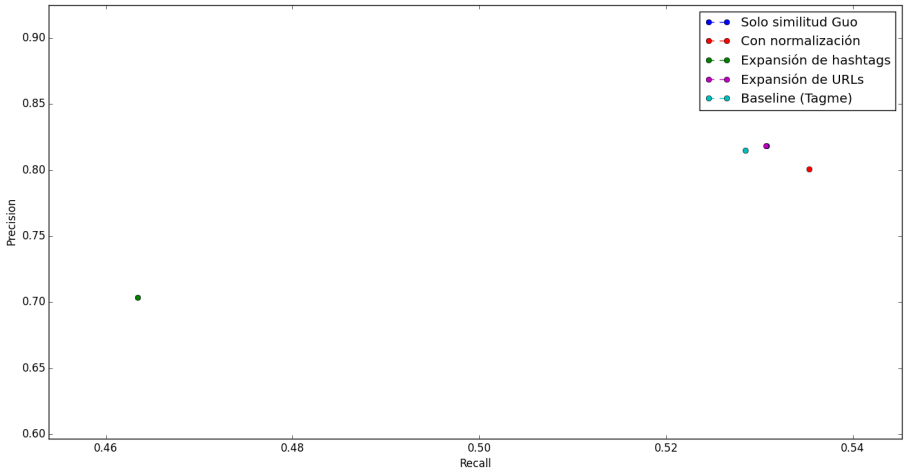
Esta gráfica con los valores de F1 solo corrobora las observaciones anteriores, siendo poco apreciables las expansiones de hashtags y URLs.

En el apéndice IV se pueden consultar los valores de precisión, cobertura y F1

alcanzados por cada modelo por separado sobre esta colección de evaluación, en forma tabular.

5.3.2. Experimentos sobre la colección NEEL

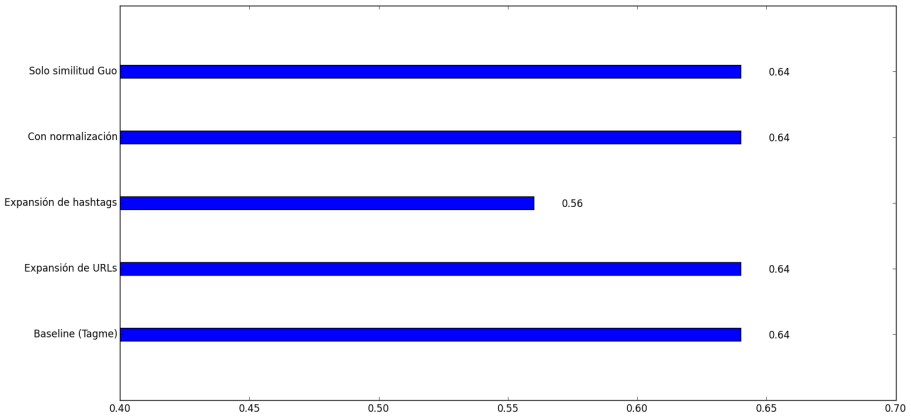
Similarmente a la sección anterior, a continuación se presenta una gráfica comparando los diferentes modelos por separado en cuanto a precisión y cobertura, el valor de *commonness* escogido es de 0.03 y el de *epsilon* de 0.7 (los resultados de la evaluación de todo el rango y sus combinaciones se pueden ver en el apéndice IV):



En la gráfica se aprecia que el modelo con solo la expansión de los hashtags tiene un rendimiento inferior a los demás modelos, lo cual pone en duda su utilidad. El punto que representa el modelo de solo similitud de Guo se encuentra solapado casi completamente por el de expansión de URLs. Se aprecia como al aplicar la normalización a los tuits, al igual que sobre la otra colección de evaluación, aumenta la cobertura ligeramente. Se observa que el uso de la similitud de Guo aumenta ligeramente la eficacia del algoritmo, mejorando la precisión y cobertura a la vez. El enriquecer el contenido del tuit por medio de la expansión de las URLs no parece aportar un beneficio apreciable y la expansión de los hashtags hace que disminuya la efectividad del modelo.

A continuación se ve la gráfica correspondiente, pero mostrando la medida F1 para los mismos modelos. Se puede observar que la medida F1 alcanzada por los modelos es similar, excepto para el modelo de expansión de hashtags, el cual tiene un rendimiento inferior.

Los resultados alcanzados al eliminar una por una las modificaciones a partir de la propuesta completa son similares, corroborando los resultados anteriores (ver apéndice IV).



5.4. Conclusiones

En este apartado se presentan las conclusiones obtenidas de la evaluación de los diferentes modelos de enlazado de entidades sobre tuits, así como de la comparación con el baseline, usando las dos colecciones de evaluación presentadas en este trabajo, una vez analizados los resultados de los experimentos de cada uno de ellos.

Los mejores modelos en cuanto a efectividad son los que utilizan la medida de similitud de Guo, obteniendo un mejor rendimiento en comparación al baseline. Esta diferencia es más visible sobre la colección reanotada de 200 tuits ya que la densidad de entidades por tuit es mayor con respecto a la colección NEEL (3.9 entidades por tuit como promedio en la colección reanotada y 1.6 tuits en la colección NEEL), por lo que el poder de desambiguación del algoritmo juega un papel más importante.

El uso de la normalización de tuits no disminuye el poder de desambiguación de los modelos, pero la muy ligera mejora en cobertura no merita que esta fase sea utilizada. Similarmente, la expansión de URLs no mejora el rendimiento del algoritmo apreciablemente.

Respecto a la expansión de hashtags, esta no parece ser una modificación positiva, su inclusión afectó negativamente el rendimiento del algoritmo sobre la colección de evaluación de la competición NEEL, y no mejoró el rendimiento de la propuesta sobre la colección de evaluación de 200 tuits.

Esto se debe a que, a diferencia de la expansión de URLs, donde el título de la página está determinado unívocamente, la definición más votada de los hashtags puede no coincidir con la usada implícitamente en el tuit, en particular, en la colección NEEL, hay muchos tuits con hashtags a signos del Zodiaco (#Taurus, #Leo, etc), estas definiciones todas han sido expandidas erróneamente, por ejemplo, #Leo “Information about the activities of a law enforcement organization.”.

Otra observación importante, es el uso del umbral de probabilidad de ser un enlace, que permite filtrar menciones no relevantes, vemos que al aumentar considerablemente el tamaño de la colección de evaluación, aumentar este umbral mejora considerablemente la precisión del algoritmo, eliminando menciones no importantes, aunque se descarten ciertas menciones no muy comunes, como observación, hay ciertas menciones que no son incluidas en el modelo de Dexter (o sea, imposibles de detectar usando esta plataforma), por tener ínfima probabilidad de aparecer, un ejemplo que aparee en la colección de evaluación es el número 1700 (1700_(number)).

También se observa, como al aumentar el contexto del tuit, por medio de las expansiones, por ejemplo, se hace necesario disminuir el valor de *epsilon*, reduciendo ligeramente la importancia de la *commonness*, y dándole más relevancia al algoritmo de desambiguación, aunque tratándose de tuits, textos de poca longitud, sigue siendo necesario un valor de *epsilonalto*.

Resumiendo, podemos concluir:

- Beneficio del uso de la medida de similitud entre entidades y cálculo de la medida de cohesión entre estas, utilizando el algoritmo diseñado por (Guo et al, 2013).
- Poca utilidad para el proceso de enlazado de la normalización de tuits y la expansión de URLs.
- Importancia de la búsqueda de valores óptimos para los parámetros sobre la colección a utilizar, particularmente *epsilon* depende del contexto de desambiguación.
- Es mejor no usar la expansión de hashtags.
- El modelo propuesto es capaz de mejorar el rendimiento del baseline (Tagme)

Capítulo 6

Conclusiones y trabajo futuro

En este capítulo se presentan las principales conclusiones derivadas de este trabajo de investigación, y a continuación se describen posibles trabajos futuros inspirados por la investigación realizada.

6.1. Conclusiones generales

En este trabajo se ha realizado un estudio del problema de enlazado de entidades en textos, y en especial, en el contexto de los tuits. Son analizados los problemas y características que hacen que el enlazado de entidades sobre tuits se diferencie del enlazado de entidades sobre textos canónico, dado el contexto limitado de desambiguación, el uso de lenguaje informal, etc. Sin embargo, dado el dinamismo del estado de la cuestión en el área, se infiere la creciente importancia de Twitter como fuente de información.

Además se ha acarreado un estudio del estado de la cuestión del enlazado de entidades sobre tuits y sobre la aplicación de técnicas de NLP específicamente sobre tuits, particularmente la normalización léxica de estos.

Es propuesto un modelo de enlazado de tuits, consistente en varias modificaciones a un algoritmo de desambiguación por votación base (implementación de Tagme). Para su implementación, me he ayudado de tecnologías de código abierto, como son TwitIE y Dexter, permitiendo el reúso y la extensibilidad del modelo, en vista a posibles modificaciones futuras. Es analizado también el uso de Wikipedia como un repositorio de información semi-estructurado.

Este modelo es evaluado experimentalmente, además se evalúa la efectividad de cada característica del modelo por separado, y se hace una comparación del modelo resultante versus el modelo base, evidenciándose una ligera mejoría en

cuanto a rendimiento frente a este.

Al realizar los diferentes experimentos, fue evidenciada la escasez de colecciones de evaluación para enlazado de entidades sobre tuits, o sea de tuits anotados, disponibles públicamente (siendo otra dificultad el hecho de que no hay consenso en las diferentes anotaciones, aunque se están dando pasos en esa dirección para hacer disponibles bases de evaluación para algoritmos de desambiguación y enlazado de tuits, por ejemplo en (Rizzo et al, 2014)

Para los experimentos se utilizó uno de los pocos datasets disponibles públicamente, y para los cuales hay resultados de varios sistemas de anotación; además se re-anotó un subconjunto de este, corrigiéndose varios errores siguiendo la misma metodología original de diseño del mismo. Este dataset se ampliará y será hecho disponible públicamente para otros investigadores.

De las modificaciones propuestas la más efectiva en cuanto a efectividad, fue utilizar las medidas de similitud y cohesión propuestas inicialmente en (Guo et al, 2013). Las otras modificaciones evaluadas fueron las de enriquecer el contexto de los tuits para ayudar a una desambiguación más efectiva entre entidades, por medio de los caracteres especiales presentes en los tuits, se mostró que la expansión de URLs puede ser empleada, y sin embargo la de hashtags no. La otra modificación analizada es la de normalizar lexicalmente los tuits, como paso previo al proceso de anotación, se discuten los pros y los contras, y según los experimentos se muestra que no brinda un aporte tangible a la efectividad final del modelo.

6.2. Trabajo futuro

Queda pendiente realizar un test de significancia estadística para comprobar si las diferencias obtenidas en cuanto a medida-F entre los distintos modelos son significativas o no.

Una posible modificación que no fue posible evaluar consiste en hacer el valor del parámetro epsilon variable en dependencia del número de entidades disponibles para la desambiguación, si el tuit tiene bastantes entidades, entonces reducir epsilon, y viceversa. O sea, epsilon varía de tuit en tuit, en dependencia del contexto de desambiguación disponible, y no es prefijado inicialmente.

Otro posible atributo a considerar, es la información temporal de los tuits; cuando ocurre un evento, comienzan a surgir seguidamente tuits relacionados, en dependencia de la importancia del evento, habrá más cantidad de estos, estos tuits suelen compartir los temas, y por tanto muchas veces las entidades para una misma mención, por lo que se podría implementar una cache temporal que guarde pares mención - entidad, los cuales expiran al pasar cierto tiempo.

Otra posible vertiente a explotar es la utilización de modelos estadísticos del lenguaje, para modelar la posible entidad dada una mención, a partir del dataset recientemente publicado por Google (ver

http://semanticweb.com/wikilinks-corpus-what-will-you-do-with-40-million-disambiguated-entity-mentions-across-10-million-plus-web-pages,

http://googleresearch.blogspot.co.uk/2013/03/learning-from-big-data-40-million.html,

http://www.iesl.cs.umass.edu/data/wiki-links)

el cual está compuesto por 40,323,863 menciones con 2,933,659 entidades.

También, muy recientemente, los autores de Tagme han puesto a disposición de los investigadores el código de la versión más reciente de Tagme, para obtenerlo es necesario contactar directamente a los autores. Habiendo obtenido una copia personalmente, sería interesante comparar su efectividad sobre las colecciones de evaluación empleadas en la experimentación de este trabajo.

Apéndice A

Instalación y uso de Dexter

Para usar la herramienta, es necesario descargarla e instalarla como se describe. Usando Maven[1], ya que es un proyecto ensamblado con esta herramienta.

Una vez instalado en el repositorio local, se puede aadir como dependencia al proyecto cliente:

```
<dependency>
        <groupId>it.cnr.isti.hpc</groupId>
        <artifactId>dexter-webapp</artifactId>
        <version>1.0.0</version>
</dependency>
```

Con esto, es posible llamar el API desde nuestro proyecto usando la clase **DexterRestClient** como en el siguiente ejemplo:

[1] `http://maven.apache.org`

```
DexterRestClient client = new DexterRestClient(
          "http://dexterdemo.isti.cnr.it:8080/rest");
AnnotatedDocument ad = client
.annotate("Dexter is an American television drama series which
debuted on Showtime on October 1, 2006. The series centers
on Dexter Morgan (Michael C. Hall), a blood spatter pattern
analyst for the fictional Miami Metro Police Department (based
on the real life Miami-Dade Police Department) who also leads
a secret life as a serial killer. Set in Miami, the show's first
season was largely based on the novel Darkly Dreaming Dexter,
the first of the Dexter series novels by Jeff Lindsay. It was
adapted for television by screenwriter James Manos, Jr.,
who wrote the first episode. ");

System.out.println(ad);
SpottedDocument sd = client
                .spot("Dexter is an American television drama
series which debuted on Showtime on October 1, 2006.
The series centers on Dexter Morgan (Michael C. Hall),
a blood spatter pattern analyst for the fictional Miami
Metro Police Department (based on the real life Miami-
Dade Police Department) who also leads a secret life
as a serial killer. Set in Miami, the show's first season
was largely based on the novel Darkly Dreaming Dexter,
the first of the Dexter series novels by Jeff Lindsay.
It was adapted for television by screenwriter James
Manos, Jr., who wrote the first episode. ");

System.out.println(sd);
ArticleDescription desc = client.getDesc(5981816);
System.out.println(desc);
```

Si tenemos la herramienta ejecutándose localmente podemos acceder a esta de la siguiente manera:

```
DexterRestClient client = new DexterRestClient(
               "http://localhost:8080/rest");
```

Nota: ejemplo tomado de la página web de Dexter[2].

[2] http://dexter.isti.cnr.it/

Apéndice B

Taxonomía de las entidades consideradas en las colecciones de evaluación

```
Amount
Animal
    Bird
    Insect
Event
    MilitaryConflict
    PoliticalEvent
    SportEvent
    WeatherEvent
    MeetingEvent
    BreakingNews
Function
    Job
Location
    AdministrativeRegion
    Airport
    Bridge
    Canal
    City
    Continent
    Country
    Hospital
    Island
    Museum
    Lake
    Lighthouse
    Mountain
    Park
    Restaurant
    River
    Road
    ShoppingMall
```

```
    Stadium
    Station
    Valley
Organization
    Airline
    Band
    Broadcast
    Company
    EducationalInstitution
    Legislature
    NonProfitOrganisation
    RadioStation
    SoccerClub
    SportsLeague
    SportsTeam
    TVStation
    University
    PoliticalOrganisation
Person
    Ambassador
    Architect
    Artist
    Astronaut
    Athlete
    Celebrity
    ComicsCharacter
    Criminal
    FictionalCharacter
    Mayor
    MusicalArtist
    Politician
    SoccerPlayer
    TennisPlayer
Product
    Aircraft
    Album
    Automobile
    Book
    Drug
    EmailAddress
    Magazine
    Movie
    Newspaper
    OperatingSystem
    ProgrammingLanguage
    RadioProgram
    SchoolNewspaper
    Software
    Song
    Spacecraft
    URL
    VideoGame
    Weapon
    Website
```

```
Time
    Holiday
Cardinal Direction
Language
Nationality
Religion
Season
AstronomicalObject
     Planet
     Natural Satellite
EthnicGroup
Weather
Sport Name
AstrologicalSig
```

Apéndice C

Evaluación de un rango de valores para el umbral de probabilidad de ser enlace

A continuación se presentan en forma tabular los resultados del cálculo de precisión/cobertura para varios valores del umbral de probabilidad de ser enlace (*linkprobability*) en un rango de [0.01 0.7] para la colección reanotada de 200 tuits.

Link probability 0.01 **commonness/epsilon** 0.03	**0.7** pre: 0.4552 rec: 0.5951 F1: 0.5158
Link probability 0.03 **commonness/epsilon** 0.03	**0.7** pre: 0.5463 rec: 0.5608 F1: 0.5534
Link probability 0.05 **commonness/epsilon** 0.03	**0.7** pre: 0.6102 rec: 0.4846 F1: 0.5402
Link probability 0.07 **commonness/epsilon** 0.03	**0.7** pre: 0.6102 rec: 0.4846 F1: 0.5402
Link probability 0.09 **commonness/epsilon** 0.03	**0.7** pre: 0.6343 rec: 0.4755 F1: 0.5436
Link probability 0.1 **commonness/epsilon** 0.03	**0.7** pre: 0.646 rec: 0.4695 F1: 0.5438
Link probability 0.3 **commonness/epsilon** 0.03	**0.7** pre: 0.78 rec: 0.3832 F1: 0.5139
Link probability 0.5 **commonness/epsilon** 0.03	**0.7** pre: 0.8086 rec: 0.2846 F1: 0.421
Link probability 0.7 **commonness/epsilon** 0.03	**0.7** pre: 0.8933 rec: 0.2039 F1: 0.332

A continuación se presentan en forma tabular los resultados del cálculo de precisión/cobertura para varios valores del umbral de probabilidad de ser enlace (*linkprobability*) en un rango de [0.01 0.7] para la colección NEEL.

Link probability 0.01
commonness/epsilon **0.7**
0.03 pre: 0.4552
rec: 0.5951
F1: 0.5158

Link probability 0.03
commonness/epsilon **0.7**
0.03 pre: 0.5463
rec: 0.5608
F1: 0.5534

Link probability 0.05
commonness/epsilon **0.7**
0.03 pre: 0.5893
rec: 0.518
F1: 0.5514

Link probability 0.07
commonness/epsilon **0.7**
0.03 pre: 0.6102
rec: 0.4846
F1: 0.5402

Link probability 0.09
commonness/epsilon **0.7**
0.03 pre: 0.6343
rec: 0.4755
F1: 0.5436

Link probability 0.1
commonness/epsilon **0.7**
0.03 pre: 0.646
rec: 0.4695
F1: 0.5438

Link probability 0.3
commonness/epsilon **0.7**
0.03 pre: 0.78
rec: 0.3832
F1: 0.5139

Link probability 0.5
commonness/epsilon **0.7**
0.03 pre: 0.8086
rec: 0.2846
F1: 0.421

Link probability 0.7
commonness/epsilon **0.7**
0.03 pre: 0.8933
rec: 0.2039
F1: 0.332

Apéndice D

Resultados de la experimentación en forma tabular

En este apéndice se pueden consultar los valores de precisión, cobertura y F1 alcanzados por cada modelo por separado sobre esta colección de evaluación, en forma tabular.

Colección de 200 tuits

Solo similitud de Guo, modelo usado como comparativa en las demás gráficas (valores en negritas en las demás tablas).Se observa que el valor máximo de F1 es alcanzado con commonness = 0.03 y epsilon=0.9, donde se maximiza la precisión/recall. Se toma como partida un valor pequeño de commonness (0.005) para tratar de aumentar el recall, pero se aprecia que el rendimiento del algoritmo usando este valor es siempre menor que con un umbral de commonness mayor. También se realizaron experimentos no mostrados aquí, para mayores mayores del umbral de commonness, donde se muestra que para valores mayores de 0.03 (0.08, 0.1) el rendimiento del algoritmo disminuye. Se maximiza el recall con un valor alto de epsilon, o sea, que se favorecen las entidades más comunes, del subconjunto más votado por arriba del umbral de aceptación.

commonness/epsilon	**0.1**	**0.3**	**0.7**	**0.9**
0.005	pre: 0.5252	pre: 0.5331	pre: 0.5489	pre: 0.5456
	rec: 0.5157	rec: 0.5242	rec: 0.5408	rec: 0.5402
	F1: 0.5204	F1: 0.5286	F1: 0.5448	F1: 0.5428
0.01	pre: 0.528	pre: 0.5354	pre: 0.5474	pre: 0.5464
	rec: 0.522	rec: 0.5316	rec: 0.5409	rec: 0.5427
	F1: 0.525	F1: 0.5335	F1: 0.5442	F1: 0.5446
0.03	pre: 0.5418	pre: 0.545	pre: 0.5488	pre: 0.549
	rec: 0.5345	rec: 0.5388	rec: 0.541	rec: 0.5444
	F1: 0.5382	F1: 0.5419	F1: 0.5448	F1: 0.5467
0.05	pre: 0.5438	pre: 0.547	pre: 0.5456	pre: 0.5458
	rec: 0.5366	rec: 0.5409	rec: 0.5393	rec: 0.5427
	F1: 0.5402	F1: 0.5439	F1: 0.5424	F1: 0.5443

Normalización de los tuits. Similarmente, se observa que el valor máximo de F1 es alcanzado con commonness = 0.03 y epsilon=0.9, donde se maximiza la precisión/recall. Se tomó como partida un valor pequeño de commonness (0.005) para tratar de aumentar el recall, pero se aprecia que el rendimiento del algoritmo usando este valor es siempre menor que con un umbral de commonness mayor. Se aprecia la mejoría contra el modelo base. La normalización de los tuits no expande el contexto, lo cual haría más efectivo un valor de epsilon menor que el máximo usado.

commonness/epsilon	**0.1**	**0.3**	**0.7**	**0.9**
0.005	pre: 0.5241	pre: 0.5331	pre: 0.5417	pre: 0.5396
	rec: 0.5302	rec: 0.5397	rec: 0.5502	rec: 0.5509
	F1: 0.5271	F1: 0.5364	F1: 0.5459	F1: 0.5452
	pre: 0.5252	**pre: 0.5331**	**pre: 0.5489**	**pre: 0.5456**
	rec: 0.5157	**rec: 0.5242**	**rec: 0.5408**	**rec: 0.5402**
	F1: 0.5204	**F1: 0.5286**	**F1: 0.5448**	**F1: 0.5428**
0.01	pre: 0.5288	pre: 0.537	pre: 0.5436	pre: 0.542
	rec: 0.5369	rec: 0.5475	rec: 0.5543	rec: 0.5543
	F1: 0.5328	F1: 0.5422	F1: 0.5489	F1: 0.5481
	pre: 0.528	**pre: 0.5354**	**pre: 0.5474**	**pre: 0.5464**
	rec: 0.522	**rec: 0.5316**	**rec: 0.5409**	**rec: 0.5427**
	F1: 0.525	**F1: 0.5335**	**F1: 0.5442**	**F1: 0.5446**
0.03	pre: 0.5377	pre: 0.5421	pre: 0.545	pre: 0.5447
	rec: 0.5459	rec: 0.5512	rec: 0.554	rec: 0.556
	F1: 0.5418	F1: 0.5466	F1: 0.5495	F1: 0.5503
	pre: 0.5418	**pre: 0.545**	**pre: 0.5488**	**pre: 0.549**
	rec: 0.5345	**rec: 0.5388**	**rec: 0.541**	**rec: 0.5444**
	F1: 0.5382	**F1: 0.5419**	**F1: 0.5448**	**F1: 0.5467**
0.05	pre: 0.5419	pre: 0.545	pre: 0.5444	pre: 0.5424
	rec: 0.5505	rec: 0.5545	rec: 0.5548	rec: 0.5556
	F1: 0.5462	F1: 0.5497	F1: 0.5496	F1: 0.5489
	pre: 0.5438	**pre: 0.547**	**pre: 0.5456**	**pre: 0.5458**
	rec: 0.5366	**rec: 0.5409**	**rec: 0.5393**	**rec: 0.5427**
	F1: 0.5402	**F1: 0.5439**	**F1: 0.5424**	**F1: 0.5443**

Expansión de hashtags presentes en tuits. Similarmente, se observa que el valor máximo de F1 es alcanzado con commonness = 0.03 y epsilon=0.9, donde se maximiza la precisión/recall. Se tomó como partida un valor pequeño de commonness (0.005) para tratar de aumentar el recall, pero se aprecia que el rendimiento del algoritmo usando este valor es siempre menor que con un umbral de commonness mayor. No se aprecia mejoría contra el modelo base, sino al revés, una muy ligera disminución del rendimiento.

commonness/epsilon	**0.1**	**0.3**	**0.7**	**0.9**
0.005	pre: 0.524	pre: 0.531	pre: 0.5477	pre: 0.545
	rec: 0.5131	rec: 0.5205	rec: 0.5376	rec: 0.5386
	F1: 0.5185	F1: 0.5257	F1: 0.5426	F1: 0.5418
	pre: 0.5252	**pre: 0.5331**	**pre: 0.5489**	**pre: 0.5456**
	rec: 0.5157	**rec: 0.5242**	**rec: 0.5408**	**rec: 0.5402**
	F1: 0.5204	**F1: 0.5286**	**F1: 0.5448**	**F1: 0.5428**
0.01	pre: 0.5281	pre: 0.5342	pre: 0.5481	pre: 0.5468
	rec: 0.5211	rec: 0.5296	rec: 0.5406	rec: 0.5422
	F1: 0.5246	F1: 0.5319	F1: 0.5443	F1: 0.5445
	pre: 0.528	**pre: 0.5354**	**pre: 0.5474**	**pre: 0.5464**
	rec: 0.522	**rec: 0.5316**	**rec: 0.5409**	**rec: 0.5427**
	F1: 0.525	**F1: 0.5335**	**F1: 0.5442**	**F1: 0.5446**
0.03	pre: 0.5409	pre: 0.5444	pre: 0.5495	pre: 0.5494
	rec: 0.5311	rec: 0.5368	rec: 0.5407	rec: 0.5439
	F1: 0.536	F1: 0.5406	F1: 0.545	F1: 0.5466
	pre: 0.5418	**pre: 0.545**	**pre: 0.5488**	**pre: 0.549**
	rec: 0.5345	**rec: 0.5388**	**rec: 0.541**	**rec: 0.5444**
	F1: 0.5382	**F1: 0.5419**	**F1: 0.5448**	**F1: 0.5467**
0.05	pre: 0.5429	pre: 0.5454	pre: 0.5463	pre: 0.5463
	rec: 0.5332	rec: 0.5377	rec: 0.539	rec: 0.5422
	F1: 0.538	F1: 0.5415	F1: 0.5426	F1: 0.5442
	pre: 0.5438	**pre: 0.547**	**pre: 0.5456**	**pre: 0.5458**
	rec: 0.5366	**rec: 0.5409**	**rec: 0.5393**	**rec: 0.5427**
	F1: 0.5402	**F1: 0.5439**	**F1: 0.5424**	**F1: 0.5443**

Expansión de URLs presentes en tuits. Similarmente, se observa que el valor máximo de F1 es alcanzado con commonness = 0.03 y epsilon=0.9, donde se maximiza la precisión/recall. Se tomó como partida un valor pequeño de commonness (0.005) para tratar de aumentar el recall, pero se aprecia que el rendimiento del algoritmo usando este valor es siempre menor que con un umbral de commonness mayor. Se aprecia la mejoría ligera contra el modelo base. La expansión de las URLs no es suficiente para expandir el contexto en la mayoría de los casos, lo cual haría más efectivo un valor de epsilon menor que el máximo usado.

commonness/epsilon	**0.1**	**0.3**	**0.7**	**0.9**
0.005	pre: 0.5284	pre: 0.5323	pre: 0.5501	pre: 0.5481
	rec: 0.5193	rec: 0.5232	rec: 0.5427	rec: 0.5437
	F1: 0.5238	F1: 0.5277	F1: 0.5464	F1: 0.5459
	pre: 0.5252	**pre: 0.5331**	**pre: 0.5489**	**pre: 0.5456**
	rec: 0.5157	**rec: 0.5242**	**rec: 0.5408**	**rec: 0.5402**
	F1: 0.5204	**F1: 0.5286**	**F1: 0.5448**	**F1: 0.5428**
0.01	pre: 0.5283	pre: 0.533	pre: 0.5463	pre: 0.5472
	rec: 0.5217	rec: 0.5279	rec: 0.5393	rec: 0.5436
	F1: 0.525	F1: 0.5304	F1: 0.5428	F1: 0.5454
	pre: 0.528	**pre: 0.5354**	**pre: 0.5474**	**pre: 0.5464**
	rec: 0.522	**rec: 0.5316**	**rec: 0.5409**	**rec: 0.5427**
	F1: 0.525	**F1: 0.5335**	**F1: 0.5442**	**F1: 0.5446**
0.03	pre: 0.5442	pre: 0.5446	pre: 0.5497	pre: 0.5498
	rec: 0.538	rec: 0.5389	rec: 0.5427	rec: 0.5452
	F1: 0.541	F1: 0.5417	F1: 0.5462	F1: 0.5475
	pre: 0.5418	**pre: 0.545**	**pre: 0.5488**	**pre: 0.549**
	rec: 0.5345	**rec: 0.5388**	**rec: 0.541**	**rec: 0.5444**
	F1: 0.5382	**F1: 0.5419**	**F1: 0.5448**	**F1: 0.5467**
0.05	pre: 0.5462	pre: 0.5466	pre: 0.5465	pre: 0.5467
	rec: 0.54	rec: 0.541	rec: 0.541	rec: 0.5436
	F1: 0.5431	F1: 0.5438	F1: 0.5438	F1: 0.5451
	pre: 0.5438	**pre: 0.547**	**pre: 0.5456**	**pre: 0.5458**
	rec: 0.5366	**rec: 0.5409**	**rec: 0.5393**	**rec: 0.5427**
	F1: 0.5402	**F1: 0.5439**	**F1: 0.5424**	**F1: 0.5443**

Uso de la medida de similitud original de Tagme. Se observa que el valor máximo de F1 es alcanzado con commonness = 0.005 y epsilon=0.9, donde se maximiza la precisión/recall. Se tomó como partida un valor pequeño de commonness (0.005) para tratar de aumentar el recall, y con este valor se alcanza el máximo de F1. Se aprecia la disminución de rendimiento en comparación contra el modelo base que emplea la similitud de Guo. El algoritmo es también favorecido por un valor grande de epsilon, favoreciendo las entidades más comunes.

commonness/epsilon	**0.1**	**0.3**	**0.7**	**0.9**
0.005	pre: 0.4141	pre: 0.4459	pre: 0.5186	pre: 0.5268
	rec: 0.4074	rec: 0.4395	rec: 0.5228	rec: 0.5267
	F1: 0.4107	F1: 0.4427	F1: 0.5207	F1: 0.5267
	pre: 0.5252	**pre: 0.5331**	**pre: 0.5489**	**pre: 0.5456**
	rec: 0.5157	**rec: 0.5242**	**rec: 0.5408**	**rec: 0.5402**
	F1: 0.5204	**F1: 0.5286**	**F1: 0.5448**	**F1: 0.5428**
0.01	pre: 0.418	pre: 0.4668	pre: 0.5241	pre: 0.5223
	rec: 0.4147	rec: 0.4618	rec: 0.5286	rec: 0.5232
	F1: 0.4163	F1: 0.4643	F1: 0.5263	F1: 0.5228
	pre: 0.528	**pre: 0.5354**	**pre: 0.5474**	**pre: 0.5464**
	rec: 0.522	**rec: 0.5316**	**rec: 0.5409**	**rec: 0.5427**
	F1: 0.525	**F1: 0.5335**	**F1: 0.5442**	**F1: 0.5446**
0.03	pre: 0.4467	pre: 0.4946	pre: 0.5238	pre: 0.5186
	rec: 0.4472	rec: 0.4883	rec: 0.5238	rec: 0.5165
	F1: 0.447	F1: 0.4914	F1: 0.5238	F1: 0.5175
	pre: 0.5418	**pre: 0.545**	**pre: 0.5488**	**pre: 0.549**
	rec: 0.5345	**rec: 0.5388**	**rec: 0.541**	**rec: 0.5444**
	F1: 0.5382	**F1: 0.5419**	**F1: 0.5448**	**F1: 0.5467**
0.05	pre: 0.4756	pre: 0.506	pre: 0.5233	pre: 0.5186
	rec: 0.4754	rec: 0.5011	rec: 0.5237	rec: 0.5165
	F1: 0.4755	F1: 0.5035	F1: 0.5235	F1: 0.5175
	pre: 0.5438	**pre: 0.547**	**pre: 0.5456**	**pre: 0.5458**
	rec: 0.5366	**rec: 0.5409**	**rec: 0.5393**	**rec: 0.5427**
	F1: 0.5402	**F1: 0.5439**	**F1: 0.5424**	**F1: 0.5443**

A continuación se muestran los valores de precisión, recall y F1 alcanzados al suprimir cada modificación a partir de la propuesta "completa" (con normalización, expansión de hashtags y URLs y usando la similitud de Guo).

Propuesta completa (valores con fondo gris en las demás tablas). Se observa que el valor máximo de F1 es alcanzado con commonness = 0.03 y epsilon=0.7, donde se maximiza la precisión/recall. Se toma como partida un valor pequeño de commonness (0.005) para tratar de aumentar el recall, pero se aprecia que el rendimiento del algoritmo usando este valor es siempre menor que con un umbral de commonness mayor. El valor de epsilon optimal es un poco menor que en las tablas de los modelos por separado, con la normalización, más las expansiones de hashtags y URLs, se amplía ligeramente el contexto del tuit, justificando un valor menor de epsilon, el cual favorece la capacidad de desambiguación del algoritmo contra la commonness de las entidades.

commonness/epsilon	**0.1**	**0.3**	**0.7**	**0.9**
0.005	pre: 0.56	pre: 0.5638	pre: 0.5739	pre: 0.5757
	rec: 0.5133	rec: 0.5154	rec: 0.5252	rec: 0.5299
	F1: 0.5357	F1: 0.5385	F1: 0.5485	F1: 0.5518
0.01	pre: 0.5567	pre: 0.5596	pre: 0.5706	pre: 0.5718
	rec: 0.5128	rec: 0.5156	rec: 0.5255	rec: 0.5284
	F1: 0.5339	F1: 0.5367	F1: 0.5471	F1: 0.5493
0.03	pre: 0.5741	pre: 0.5725	pre: 0.5785	pre: 0.5753
	rec: 0.5304	rec: 0.5294	rec: 0.5353	rec: 0.534
	F1: 0.5514	F1: 0.5501	F1: 0.5561	F1: 0.5539
0.05	pre: 0.5754	pre: 0.5737	pre: 0.576	pre: 0.5728
	rec: 0.5325	rec: 0.5315	rec: 0.5348	rec: 0.5335
	F1: 0.5531	F1: 0.5518	F1: 0.5547	F1: 0.5525

Sin normalización de los tuits. Se observa que el valor máximo de F1 es alcanzado con commonness = 0.03 y epsilon=0.9, donde se maximiza la precisión/recall. Al eliminar la expansión de URLs, disminuye ligeramente el contexto, por lo que se favorece un valor mayor de epsilon. Al eliminar esta modificación, el rendimiento decae ligeramente.

commonness/epsilon	**0.1**	**0.3**	**0.7**	**0.9**
0.005	pre: 0.5628	pre: 0.57	pre: 0.5843	pre: 0.586
	rec: 0.4944	rec: 0.5009	rec: 0.5121	rec: 0.5165
	F1: 0.5264	F1: 0.5332	F1: 0.5458	F1: 0.5491
	pre: 0.56	**pre: 0.5638**	**pre: 0.5739**	**pre: 0.5757**
	rec: 0.5133	**rec: 0.5154**	**rec: 0.5252**	**rec: 0.5299**
	F1: 0.5357	**F1: 0.5385**	**F1: 0.5485**	**F1: 0.5518**
0.01	pre: 0.5678	pre: 0.5716	pre: 0.5828	pre: 0.586
	rec: 0.5005	rec: 0.5041	rec: 0.5122	rec: 0.5181
	F1: 0.532	F1: 0.5357	F1: 0.5452	F1: 0.55
	pre: 0.5567	**pre: 0.5596**	**pre: 0.5706**	**pre: 0.5718**
	rec: 0.5128	**rec: 0.5156**	**rec: 0.5255**	**rec: 0.5284**
	F1: 0.5339	**F1: 0.5367**	**F1: 0.5471**	**F1: 0.5493**
0.03	pre: 0.5808	pre: 0.5832	pre: 0.5873	pre: 0.588
	rec: 0.5137	rec: 0.5163	rec: 0.5185	rec: 0.522
	F1: 0.5452	F1: 0.5477	F1: 0.5508	F1: 0.553
	pre: 0.5741	**pre: 0.5725**	**pre: 0.5785**	**pre: 0.5753**
	rec: 0.5304	**rec: 0.5294**	**rec: 0.5353**	**rec: 0.534**
	F1: 0.5514	**F1: 0.5501**	**F1: 0.5561**	**F1: 0.5539**
0.05	pre: 0.5839	pre: 0.5863	pre: 0.585	pre: 0.5856
	rec: 0.5171	rec: 0.5197	rec: 0.5181	rec: 0.5215
	F1: 0.5485	F1: 0.551	F1: 0.5495	F1: 0.5517
	pre: 0.5754	**pre: 0.5737**	**pre: 0.576**	**pre: 0.5728**
	rec: 0.5325	**rec: 0.5315**	**rec: 0.5348**	**rec: 0.5335**
	F1: 0.5531	**F1: 0.5518**	**F1: 0.5547**	**F1: 0.5525**

Sin expansión de URLs presentes en tuits. Se observa que el valor máximo de F1 es alcanzado con commonness = 0.03 y epsilon=0.7, donde se maximiza

la precisión/recall. Acorde a los resultados, esta modificación no enriquece el contexto suficientemente como para influir sobre el valor óptimo de epsilon. Al eliminar esta modificación, el rendimiento mejora ligeramente.

commonness/epsilon	**0.1**	**0.3**	**0.7**	**0.9**
0.005	pre: 0.5595	pre: 0.564	pre: 0.5739	pre: 0.5725
	rec: 0.5187	rec: 0.523	rec: 0.5312	rec: 0.5315
	F1: 0.5383	F1: 0.5428	F1: 0.5517	F1: 0.5512
	pre: 0.56	**pre: 0.5638**	**pre: 0.5739**	**pre: 0.5757**
	rec: 0.5133	**rec: 0.5154**	**rec: 0.5252**	**rec: 0.5299**
	F1: 0.5357	**F1: 0.5385**	**F1: 0.5485**	**F1: 0.5518**
0.01	pre: 0.5622	pre: 0.5655	pre: 0.5725	pre: 0.5732
	rec: 0.5217	rec: 0.5254	rec: 0.5313	rec: 0.5339
	F1: 0.5412	F1: 0.5448	F1: 0.5511	F1: 0.5528
	pre: 0.5567	**pre: 0.5596**	**pre: 0.5706**	**pre: 0.5718**
	rec: 0.5128	**rec: 0.5156**	**rec: 0.5255**	**rec: 0.5284**
	F1: 0.5339	**F1: 0.5367**	**F1: 0.5471**	**F1: 0.5493**
0.03	pre: 0.5732	pre: 0.5742	pre: 0.5775	pre: 0.5757
	rec: 0.5339	rec: 0.5356	rec: 0.5383	rec: 0.5387
	F1: 0.5529	F1: 0.5542	F1: 0.5573	F1: 0.5566
	pre: 0.5741	**pre: 0.5725**	**pre: 0.5785**	**pre: 0.5753**
	rec: 0.5304	**rec: 0.5294**	**rec: 0.5353**	**rec: 0.534**
	F1: 0.5514	**F1: 0.5501**	**F1: 0.5561**	**F1: 0.5539**
0.05	pre: 0.5745	pre: 0.5755	pre: 0.575	pre: 0.5732
	rec: 0.536	rec: 0.5377	rec: 0.5379	rec: 0.5383
	F1: 0.5546	F1: 0.5559	F1: 0.5559	F1: 0.5552
	pre: 0.5754	**pre: 0.5737**	**pre: 0.576**	**pre: 0.5728**
	rec: 0.5325	**rec: 0.5315**	**rec: 0.5348**	**rec: 0.5335**
	F1: 0.5531	**F1: 0.5518**	**F1: 0.5547**	**F1: 0.5525**

Sin expansión de hashtags presentes en tuits. Se observa que el valor máximo de F1 es alcanzado con commonness = 0.03 y epsilon=0.7, donde se maximiza la precisión/recall. Acorde a los resultados, esta modificación por sí sola no enriquece el contexto suficientemente como para influir sobre el valor óptimo de epsilon. Al eliminar esta modificación, el rendimiento disminuye ligeramente.

commonness/epsilon	**0.1**	**0.3**	**0.7**	**0.9**
0.005	pre: 0.5542	pre: 0.5617	pre: 0.571	pre: 0.5736
	rec: 0.5061	rec: 0.5126	rec: 0.52	rec: 0.5248
	F1: 0.5291	F1: 0.5361	F1: 0.5443	F1: 0.5481
	pre: 0.56	**pre: 0.5638**	**pre: 0.5739**	**pre: 0.5757**
	rec: 0.5133	**rec: 0.5154**	**rec: 0.5252**	**rec: 0.5299**
	F1: 0.5357	**F1: 0.5385**	**F1: 0.5485**	**F1: 0.5518**
0.01	pre: 0.5617	pre: 0.5663	pre: 0.5738	pre: 0.5764
	rec: 0.5128	rec: 0.5175	rec: 0.5244	rec: 0.5284
	F1: 0.5362	F1: 0.5408	F1: 0.548	F1: 0.5514
	pre: 0.5567	**pre: 0.5596**	**pre: 0.5706**	**pre: 0.5718**
	rec: 0.5128	**rec: 0.5156**	**rec: 0.5255**	**rec: 0.5284**
	F1: 0.5339	**F1: 0.5367**	**F1: 0.5471**	**F1: 0.5493**
0.03	pre: 0.5735	pre: 0.577	pre: 0.5804	pre: 0.5787
	rec: 0.5251	rec: 0.5288	rec: 0.5315	rec: 0.5323
	F1: 0.5483	F1: 0.5519	F1: 0.5549	F1: 0.5545
	pre: 0.5741	**pre: 0.5725**	**pre: 0.5785**	**pre: 0.5753**
	rec: 0.5304	**rec: 0.5294**	**rec: 0.5353**	**rec: 0.534**
	F1: 0.5514	**F1: 0.5501**	**F1: 0.5561**	**F1: 0.5539**
0.05	pre: 0.576	pre: 0.5783	pre: 0.5779	pre: 0.5762
	rec: 0.5285	rec: 0.5308	rec: 0.5311	rec: 0.5319
	F1: 0.5512	F1: 0.5536	F1: 0.5535	F1: 0.5531
	pre: 0.5754	**pre: 0.5737**	**pre: 0.576**	**pre: 0.5728**
	rec: 0.5325	**rec: 0.5315**	**rec: 0.5348**	**rec: 0.5335**
	F1: 0.5531	**F1: 0.5518**	**F1: 0.5547**	**F1: 0.5525**

Sin expansión de hashtags ni URLs presentes en tuits. Se observa que el valor máximo de F1 es alcanzado con commonness = 0.03 y epsilon=0.7, donde se maximiza la precisión/recall. Acorde a los resultados, esta modificación por sí sola no enriquece el contexto suficientemente como para influir sobre el valor óptimo de epsilon. Al eliminar esta modificación, el rendimiento disminuye ligeramente.

commonness/epsilon	**0.1**	**0.3**	**0.7**	**0.9**
0.005	pre: 0.5542	pre: 0.5617	pre: 0.571	pre: 0.5736
	rec: 0.5061	rec: 0.5126	rec: 0.52	rec: 0.5248
	F1: 0.5291	F1: 0.5361	F1: 0.5443	F1: 0.5481
	pre: 0.56	**pre: 0.5638**	**pre: 0.5739**	**pre: 0.5757**
	rec: 0.5133	**rec: 0.5154**	**rec: 0.5252**	**rec: 0.5299**
	F1: 0.5357	**F1: 0.5385**	**F1: 0.5485**	**F1: 0.5518**
0.01	pre: 0.5617	pre: 0.5663	pre: 0.5738	pre: 0.5764
	rec: 0.5128	rec: 0.5175	rec: 0.5244	rec: 0.5284
	F1: 0.5362	F1: 0.5408	F1: 0.548	F1: 0.5514
	pre: 0.5567	**pre: 0.5596**	**pre: 0.5706**	**pre: 0.5718**
	rec: 0.5128	**rec: 0.5156**	**rec: 0.5255**	**rec: 0.5284**
	F1: 0.5339	**F1: 0.5367**	**F1: 0.5471**	**F1: 0.5493**
0.03	pre: 0.5735	pre: 0.577	pre: 0.5804	pre: 0.5787
	rec: 0.5251	rec: 0.5288	rec: 0.5315	rec: 0.5323
	F1: 0.5483	F1: 0.5519	F1: 0.5549	F1: 0.5545
	pre: 0.5741	**pre: 0.5725**	**pre: 0.5785**	**pre: 0.5753**
	rec: 0.5304	**rec: 0.5294**	**rec: 0.5353**	**rec: 0.534**
	F1: 0.5514	**F1: 0.5501**	**F1: 0.5561**	**F1: 0.5539**
0.05	pre: 0.576	pre: 0.5783	pre: 0.5779	pre: 0.5762
	rec: 0.5285	rec: 0.5308	rec: 0.5311	rec: 0.5319
	F1: 0.5512	F1: 0.5536	F1: 0.5535	F1: 0.5531
	pre: 0.5754	**pre: 0.5737**	**pre: 0.576**	**pre: 0.5728**
	rec: 0.5325	**rec: 0.5315**	**rec: 0.5348**	**rec: 0.5335**
	F1: 0.5531	**F1: 0.5518**	**F1: 0.5547**	**F1: 0.5525**

Sin usar la similitud de Guo, o sea, usando la medida de similitud original de Tagme. Se observa que el valor máximo de F1 es alcanzado con commonness = 0.005 y epsilon=0.9, donde se maximiza la precisión/recall. Al eliminar esta modificación, el rendimiento disminuye considerablemente.

commonness/epsilon	**0.1**	**0.3**	**0.7**	**0.9**
0.005	pre: 0.3988	pre: 0.4544	pre: 0.5482	pre: 0.5576
	rec: 0.3676	rec: 0.4152	rec: 0.5115	rec: 0.5215
	F1: 0.3825	F1: 0.4339	F1: 0.5292	F1: 0.5389
	pre: 0.56	**pre: 0.5638**	**pre: 0.5739**	**pre: 0.5757**
	rec: 0.5133	**rec: 0.5154**	**rec: 0.5252**	**rec: 0.5299**
	F1: 0.5357	**F1: 0.5385**	**F1: 0.5485**	**F1: 0.5518**
0.01	pre: 0.3993	pre: 0.4706	pre: 0.5539	pre: 0.5543
	rec: 0.3694	rec: 0.4341	rec: 0.5153	rec: 0.5188
	F1: 0.3838	F1: 0.4516	F1: 0.5339	F1: 0.536
	pre: 0.5567	**pre: 0.5596**	**pre: 0.5706**	**pre: 0.5718**
	rec: 0.5128	**rec: 0.5156**	**rec: 0.5255**	**rec: 0.5284**
	F1: 0.5339	**F1: 0.5367**	**F1: 0.5471**	**F1: 0.5493**
0.03	pre: 0.4509	pre: 0.512	pre: 0.5545	pre: 0.5483
	rec: 0.4217	rec: 0.4747	rec: 0.5198	rec: 0.5115
	F1: 0.4358	F1: 0.4926	F1: 0.5366	F1: 0.5293
	pre: 0.5741	**pre: 0.5725**	**pre: 0.5785**	**pre: 0.5753**
	rec: 0.5304	**rec: 0.5294**	**rec: 0.5353**	**rec: 0.534**
	F1: 0.5514	**F1: 0.5501**	**F1: 0.5561**	**F1: 0.5539**
0.05	pre: 0.4931	pre: 0.5338	pre: 0.5554	pre: 0.5483
	rec: 0.462	rec: 0.4951	rec: 0.5196	rec: 0.5115
	F1: 0.477	F1: 0.5137	F1: 0.5369	F1: 0.5293
	pre: 0.5754	**pre: 0.5737**	**pre: 0.576**	**pre: 0.5728**
	rec: 0.5325	**rec: 0.5315**	**rec: 0.5348**	**rec: 0.5335**
	F1: 0.5531	**F1: 0.5518**	**F1: 0.5547**	**F1: 0.5525**

Colección NEEL

Solo similitud de Guo, modelo usado como comparativa en las demás gráficas (valores con fondo gris en las demás tablas).Se observa que el valor máximo de F1 es alcanzado con commonness = 0.03 y epsilon=0.7, donde se maximiza la precisión/recall. Obsérvese también el aumento de la precisión con respecto a los experimentos sobre la colección de 200 tuits, esto se debe en parte al aumento del umbral de probabilidad de ser enlace a 0.3, sin perder en recall, y ademas a que en esta colección de evaluación hay menos entidades por tuit (además habiendo un 30 % de tuits sin entidades).

Se toma como partida un valor pequeño de commonness (0.005) para tratar de aumentar el recall, pero se aprecia que el rendimiento del algoritmo usando este valor es siempre menor que con un umbral de commonness mayor (fueron probados exhaustivamente varios valores entre 0.001 y 0.1). Se maximiza el recall con un valor alto de epsilon, o sea, que se favorecen las entidades más comunes del subconjunto más votado por encima del umbral de aceptación.

commonness/epsilon	**0.1**	**0.3**	**0.7**	**0.9**
0.01	pre: 0.8143	pre: 0.8153	pre: 0.8173	pre: 0.8166
	rec: 0.5277	rec: 0.5285	rec: 0.5302	rec: 0.5295
	F1: 0.6404	F1: 0.6413	F1: 0.6431	F1: 0.6425
0.03	pre: 0.8173	pre: 0.8177	pre: 0.8183	pre: 0.8175
	rec: 0.53	rec: 0.5302	rec: 0.5307	rec: 0.5301
	F1: 0.643	F1: 0.6433	F1: 0.6439	F1: 0.6432
0.05	pre: 0.8171	pre: 0.8174	pre: 0.8179	pre: 0.8174
	rec: 0.53	rec: 0.5301	rec: 0.5305	rec: 0.53
	F1: 0.6429	F1: 0.6431	F1: 0.6436	F1: 0.643
0.08	pre: 0.8169	pre: 0.8172	pre: 0.8178	pre: 0.8168
	rec: 0.5298	rec: 0.5299	rec: 0.5304	rec: 0.5296
	F1: 0.6427	F1: 0.6429	F1: 0.6435	F1: 0.6426

Normalización de los tuits. Similarmente, se observa que el valor máximo de F1 es alcanzado con commonness = 0.03 y epsilon=0.7, donde se maximiza la precisión/recall. Se tomó como partida un valor pequeño de commonness (0.01) para tratar de aumentar el recall, pero se aprecia que el rendimiento del algoritmo usando este valor es siempre menor que con un umbral de commonness mayor. Se aprecia una ligera mejoría contra el modelo base en cuanto a recall, pero disminuye la precisión, haciendo que la F1 sea menor en comparación. La normalización de los tuits no expande el contexto, lo cual haría más efectivo un valor de epsilon menor que el usado.

commonness/epsilon	**0.1**	**0.3**	**0.7**	**0.9**
0.01	pre: 0.7981	pre: 0.7991	pre: 0.7996	pre: 0.7992
	rec: 0.5354	rec: 0.5359	rec: 0.5346	rec: 0.5341
	F1: 0.6409	F1: 0.6416	F1: 0.6408	F1: 0.6403
	pre: 0.8143	**pre: 0.8153**	**pre: 0.8173**	**pre: 0.8166**
	rec: 0.5277	**rec: 0.5285**	**rec: 0.5302**	**rec: 0.5295**
	F1: 0.6404	**F1: 0.6413**	**F1: 0.6431**	**F1: 0.6425**
0.03	pre: 0.801	pre: 0.8015	pre: 0.8009	pre: 0.8002
	rec: 0.5375	rec: 0.5374	rec: 0.5353	rec: 0.5347
	F1: 0.6433	F1: 0.6434	F1: 0.6417	F1: 0.641
	pre: 0.8173	**pre: 0.8177**	**pre: 0.8183**	**pre: 0.8175**
	rec: 0.53	**rec: 0.5302**	**rec: 0.5307**	**rec: 0.5301**
	F1: 0.643	**F1: 0.6433**	**F1: 0.6439**	**F1: 0.6432**
0.05	pre: 0.8007	pre: 0.801	pre: 0.8002	pre: 0.7998
	rec: 0.5374	rec: 0.5372	rec: 0.535	rec: 0.5343
	F1: 0.6432	F1: 0.6431	F1: 0.6413	F1: 0.6406
	pre: 0.8171	**pre: 0.8174**	**pre: 0.8179**	**pre: 0.8174**
	rec: 0.53	**rec: 0.5301**	**rec: 0.5305**	**rec: 0.53**
	F1: 0.6429	**F1: 0.6431**	**F1: 0.6436**	**F1: 0.643**
0.08	pre: 0.8007	pre: 0.801	pre: 0.8002	pre: 0.7991
	rec: 0.5373	rec: 0.5372	rec: 0.5349	rec: 0.5339
	F1: 0.6431	F1: 0.6431	F1: 0.6412	F1: 0.6401
	pre: 0.8169	**pre: 0.8172**	**pre: 0.8178**	**pre: 0.8168**
	rec: 0.5298	**rec: 0.5299**	**rec: 0.5304**	**rec: 0.5296**
	F1: 0.6427	**F1: 0.6429**	**F1: 0.6435**	**F1: 0.6426**

Expansión de hashtags presentes en tuits. Con este modelo se observa una disminución del rendimiento. Los valores obtenidos de precisión/recall/F1 son casi idénticos para distintos valores de commonness y epsilon, y son menores con respecto al modelo base.

commonness/epsilon	**0.1**	**0.3**	**0.7**	**0.9**
0.01	pre: 0.5341	pre: 0.5341	pre: 0.5342	pre: 0.5304
	rec: 0.3504	rec: 0.3504	rec: 0.3505	rec: 0.3465
	F1: 0.4232	F1: 0.4232	F1: 0.4233	F1: 0.4192
	pre: 0.8143	**pre: 0.8153**	**pre: 0.8173**	**pre: 0.8166**
	rec: 0.5277	**rec: 0.5285**	**rec: 0.5302**	**rec: 0.5295**
	F1: 0.6404	**F1: 0.6413**	**F1: 0.6431**	**F1: 0.6425**
0.03	pre: 0.5341	pre: 0.5341	pre: 0.5342	pre: 0.5304
	rec: 0.3504	rec: 0.3504	rec: 0.3505	rec: 0.3465
	F1: 0.4232	F1: 0.4232	F1: 0.4233	F1: 0.4192
	pre: 0.8173	**pre: 0.8177**	**pre: 0.8183**	**pre: 0.8175**
	rec: 0.53	**rec: 0.5302**	**rec: 0.5307**	**rec: 0.5301**
	F1: 0.643	**F1: 0.6433**	**F1: 0.6439**	**F1: 0.6432**
0.05	pre: 0.5341	pre: 0.5341	pre: 0.5342	pre: 0.5304
	rec: 0.3504	rec: 0.3504	rec: 0.3505	rec: 0.3465
	F1: 0.4232	F1: 0.4232	F1: 0.4233	F1: 0.4192
	pre: 0.8171	**pre: 0.8174**	**pre: 0.8179**	**pre: 0.8174**
	rec: 0.53	**rec: 0.5301**	**rec: 0.5305**	**rec: 0.53**
	F1: 0.6429	**F1: 0.6431**	**F1: 0.6436**	**F1: 0.643**
0.08	pre: 0.5341	pre: 0.5341	pre: 0.5342	pre: 0.5304
	rec: 0.3504	rec: 0.3504	rec: 0.3505	rec: 0.3465
	F1: 0.4232	F1: 0.4232	F1: 0.4233	F1: 0.4192
	pre: 0.8169	**pre: 0.8172**	**pre: 0.8178**	**pre: 0.8168**
	rec: 0.5298	**rec: 0.5299**	**rec: 0.5304**	**rec: 0.5296**
	F1: 0.6427	**F1: 0.6429**	**F1: 0.6435**	**F1: 0.6426**

Expansión de URLs presentes en tuits. Se observa que el valor máximo de F1 es alcanzado con commonness = 0.03 y epsilon=0.7, donde se maximiza la precisión/recall. Se tomó como partida un valor pequeño de commonness (0.01) para tratar de aumentar el recall, pero se aprecia que el rendimiento del algoritmo usando este valor es siempre menor que con un umbral de commonness mayor. Se aprecia una mejoría ligera contra el modelo base. La expansión de las URLs no es suficiente para expandir el contexto en la mayoría de los casos, lo cual haría más efectivo un valor de epsilon menor que el máximo usado.

commonness/epsilon	**0.1**	**0.3**	**0.7**	**0.9**
0.01	pre: 0.8148	pre: 0.8156	pre: 0.818	pre: 0.8172
	rec: 0.5282	rec: 0.5286	rec: 0.5302	rec: 0.5296
	F1: 0.6409	F1: 0.6415	F1: 0.6434	F1: 0.6427
	pre: 0.8143	**pre: 0.8153**	**pre: 0.8173**	**pre: 0.8166**
	rec: 0.5277	**rec: 0.5285**	**rec: 0.5302**	**rec: 0.5295**
	F1: 0.6404	**F1: 0.6413**	**F1: 0.6431**	**F1: 0.6425**
0.03	pre: 0.8172	pre: 0.8176	pre: 0.8186	pre: 0.8177
	rec: 0.5299	rec: 0.5301	rec: 0.5307	rec: 0.5301
	F1: 0.6429	F1: 0.6432	F1: 0.6439	F1: 0.6432
	pre: 0.8173	**pre: 0.8177**	**pre: 0.8183**	**pre: 0.8175**
	rec: 0.53	**rec: 0.5302**	**rec: 0.5307**	**rec: 0.5301**
	F1: 0.643	**F1: 0.6433**	**F1: 0.6439**	**F1: 0.6432**
0.05	pre: 0.8175	pre: 0.8178	pre: 0.8182	pre: 0.8176
	rec: 0.5301	rec: 0.5303	rec: 0.5305	rec: 0.53
	F1: 0.6432	F1: 0.6434	F1: 0.6437	F1: 0.6431
	pre: 0.8171	**pre: 0.8174**	**pre: 0.8179**	**pre: 0.8174**
	rec: 0.53	**rec: 0.5301**	**rec: 0.5305**	**rec: 0.53**
	F1: 0.6429	**F1: 0.6431**	**F1: 0.6436**	**F1: 0.643**
0.08	pre: 0.8173	pre: 0.8176	pre: 0.8181	pre: 0.8171
	rec: 0.53	rec: 0.5302	rec: 0.5306	rec: 0.5297
	F1: 0.6431	F1: 0.6433	F1: 0.6437	F1: 0.6427
	pre: 0.8169	**pre: 0.8172**	**pre: 0.8178**	**pre: 0.8168**
	rec: 0.5298	**rec: 0.5299**	**rec: 0.5304**	**rec: 0.5296**
	F1: 0.6427	**F1: 0.6429**	**F1: 0.6435**	**F1: 0.6426**

Uso de la medida de similitud original de Tagme. Se observa que el valor máximo de F1 es alcanzado con commonness = 0.05 y epsilon=0.1, donde se maximiza la precisión/recall. Se aprecia la disminución de rendimiento en comparación contra el modelo base que emplea la similitud de Guo. El algoritmo es favorecido por un valor pequeño de epsilon, favoreciendo las entidades escogidas por el algoritmo de desambiguación sobre las entidades más comunes. Se utiliza también un umbral más alto de commonness, aunque estos cambios no son suficientes para mejorar sustancialmente la efectividad del modelo.

commonness/epsilon	**0.1**	**0.3**	**0.7**	**0.9**
0.005	pre: 0.5173	pre: 0.5141	pre: 0.5135	pre: 0.5105
	rec: 0.354	rec: 0.3504	rec: 0.3495	rec: 0.3461
	F1: 0.4204	F1: 0.4168	F1: 0.4159	F1: 0.4125
	pre: 0.8133	**pre: 0.8147**	**pre: 0.8169**	**pre: 0.8164**
	rec: 0.5268	**rec: 0.5279**	**rec: 0.5296**	**rec: 0.5295**
	F1: 0.6394	**F1: 0.6407**	**F1: 0.6426**	**F1: 0.6424**
0.01	pre: 0.517	pre: 0.5137	pre: 0.5137	pre: 0.5105
	rec: 0.3539	rec: 0.3498	rec: 0.3496	rec: 0.3461
	F1: 0.4201	F1: 0.4162	F1: 0.4161	F1: 0.4125
	pre: 0.8143	**pre: 0.8153**	**pre: 0.8173**	**pre: 0.8166**
	rec: 0.5277	**rec: 0.5285**	**rec: 0.5302**	**rec: 0.5295**
	F1: 0.6404	**F1: 0.6413**	**F1: 0.6431**	**F1: 0.6425**
0.03	pre: 0.5191	pre: 0.5181	pre: 0.5137	pre: 0.5105
	rec: 0.3561	rec: 0.3548	rec: 0.3496	rec: 0.3461
	F1: 0.4224	F1: 0.4212	F1: 0.4161	F1: 0.4125
	pre: 0.8173	**pre: 0.8177**	**pre: 0.8183**	**pre: 0.8175**
	rec: 0.53	**rec: 0.5302**	**rec: 0.5307**	**rec: 0.5301**
	F1: 0.643	**F1: 0.6433**	**F1: 0.6439**	**F1: 0.6432**
0.05	pre: 0.5199	pre: 0.5179	pre: 0.5137	pre: 0.5105
	rec: 0.3571	rec: 0.3547	rec: 0.3496	rec: 0.3461
	F1: 0.4234	F1: 0.421	F1: 0.4161	F1: 0.4125
	pre: 0.8171	**pre: 0.8174**	**pre: 0.8179**	**pre: 0.8174**
	rec: 0.53	**rec: 0.5301**	**rec: 0.5305**	**rec: 0.53**
	F1: 0.6429	**F1: 0.6431**	**F1: 0.6436**	**F1: 0.643**

Bibliografía

[DBPEDIA] http://dbpedia.org/

[WORDNET] http://wordnet.princeton.edu/

[CYC] http://www.cyc.com/

[OPENCYC] http://www.cyc.com/platform/opencyc

[TOUTANOVA 2003] K. Toutanova, D. Klein, C. Manning, and Y. Singer. 2003. Feature-Rich Part-of-Speech Tagging with a Cyclic Dependency Network (2003 In Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology, pp. 173 - 180. ACL.

[GIMPEL 2011] Gimpel 2011 K. Gimpel, N. Schneider, B. O'Connor, D. Das, D. Mills,J. Eisenstein, M. Heilman, D. Yogatama, J. Flanigan, and N. A. Smith. 2011. Part-of-speech tagging for Twitter: Annotation, features, and experiments. In Proc. of ACL.

[RITTER 2011] Ritter 2011 Named Entity Recognition in Tweets: An Experimental Study, by A. Ritter, S. Clark, Mausam, O. Etzioni. In Empirical Methods in Natural Language Processing, 2011.

[FINKEL 2005] Finkel, J. R., T. Grenager & C. Manning (2005). Incorporating non local information into information extraction systems by Gibbs sampling. In Proc. of ACL-05, pp. 363 - 370.

[DU BOIS 2007] Du Bois., John W. 2007. The stance triangle. In Robert Engelbretson, editor, Stancetaking in discourse,Amsterdam/Philadelphia.

[AW ZHANG 2006] Aw, Zhang, et al. 2006. A phrase-based statistical model for SMS text normalization.

[LOPEZ] V. Lpez-Ludea, R. San-Segundo, J. M. Montero, R. Barra-Chicote, J. Lorenzo Architecture for Text Normalization using Statistical Machine Translation techniques. Universidad Politcnica de Madrid. Spain.

[KAUFMANN 2010] Joseph Kaufmann and Jugal Kalita. 2010. Syntactic nor malization of Twitter messages. In International Con ference on Natural Language Processing, Kharagpur India.

[SHANNON 1948] Shannon, Claude E. .ᴬ Mathematical Theory of Communication". Bell System Technical Journal 27 (3): 379 - 423.

[MONOJIT 2007] Monojit Choudhury, Rahul Saraf, Vijit Jain, Animesh Mukherjee, Sudeshna Sarkar, and Anupam Basu. 2007. Investigation and modeling of the structure of texting language. International Journal on Document Analysis and Recognition, 10(3):157 - 174.

[GOUWS 1990] Stephan Gouws, Dirk Hovy, and Donald Metzler. 2011 Unsupervised mining of lexical variants from noisy text. In Proceedings of the First workshop on Unsu pervised Learning in NLP, pages 82-90, Edinburgh, Scotland, UK

[HAN 2011] Han, Bo and Timothy Baldwin (2011) Lexical Normalisation of Short Text Messages: Makn Sens a #twitter, In Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (ACL HLT 2011), Portland, USA, pp. 368-378

[FREEBASE] `https://www.freebase.com/`

[HAN 2012] Bo Han, Paul Cook, and Timothy Baldwin. 2012. Au tomatically constructing a normalisation dictionary for microblogs. In Proceedings of the 2012 Joint for microblogs. pages 421-432. ACL

[OLUTOBI 2012] Olutobi Owoputi , Chris Dyer , Kevin Gimpel , Nathan Schneider , Noah A. Smith Improved part-of-speech tagging for online conversational text with word clusters (2013) In Proceedings of NAACL

[GATE] `https://gate.ac.uk/`

[EISENSTEIN 2013] Eisenstein J.. 2013b. What to do about bad language on the internet. In Proceedings of NAACL, pages 359-369.

[RADA 2007] Rada Mihalcea, Andras Csomai: 2007. Wikify!: linking documents to encyclopedic knowledge

[MILNET 2008] Milne, D.; and Witten, I.H. 2008. An effective, low-cost measure of semantic relatedness obtained from Wikipedia links.

[MILNET 2008a] Milne, D.; and Witten, I.H. 2008. Learning to link with Wikipedia.

[SAYALI 2009] Sayali Kulkarni, Amit Singh, Ganesh Ramakrishnan, and Soumen Chakrabarti. 2009. Collective annotation of wikipedia entities in web text.

[RATINOV 2011] L Ratinov, D Roth, D Downey, M Anderson. 2011. Local and global algorithms for disambiguation to Wikipedia.

[JIANG 2003] Peng Jiang, Huiman Hou, Lijiang Chen, Shimin Chen, Conglei Yao, Chengkai Li, and Min Wang. 2013. Wiki3C: exploiting wikipedia for context-aware concept categorization.

[OPENCALAIS] OpenCalais http://www.opencalais.com/

[MEJI 2013] E. Meij. 2013. http://ejmeij.github.io/entity-linking-and-retrieval-tutorial/

[WEERKAMO2012] Meij E, Weerkamp W, de Rijke M. 2012. Adding semantics to microblog posts.

[BURGES 2011] C. J. C. Burges, K. M. Svore, P. N. Bennett, A. Pastusiak, and Q. Wu. 2011. Learning to rank using an ensemble of lambda-gradient models.

[SGUO 2013] S Guo, Ming-Wei Chang, Emre Kcman. 2013. To Link or Not to Link? A Study on End-to-End Tweet Entity Linking.

[LIU 2013] Xiaohua Liu, Yitong Li, Haocheng Wu, Ming Zhou, Furu Wei and Yi Lu. 2013. Entity Linking for tweets.

[YGUO 2013] Yuhang Guo; Bing Qin; Ting Liu; Sheng Li. 2013. Microblog Entity Linking by Leveraging Extra Posts.

[CASSIDY 2013] Taylor Cassidy, Heng Ji, Lev-Arie Ratinov, Arkaitz Zubiaga, Hongzhao Huang. 2013. Analysis and Enhancement of Wikification for Microblogs with Context Expansion.

[WIKIPEDIAMINER] WikipediaMiner http://wikipedia-miner.cms.waikato.ac.nz/

[DEXTER] Dexter: an Open Source Framework for Entity Linking http://dexter.isti.cnr.it/

[TAGME] TAGME http://tagme.di.unipi.it/

[MILNE3] Milne, D. 2012. An Open-Source Toolkit for Mining Wikipedia.

[FERRAGINA2010] Ferragina P., Scaiella U. 2010. Fast and accurate annotation of short texts with Wikipedia pages.

[CUNNINGHAM 2013] H. Cunningham, V. Tablan, A. Roberts, K. Bontcheva (2013) Getting More Out of Biomedical Documents with GATE's Full Lifecycle Open Source Text Analytics. PLoS Comput Biol

[CUNNINGHAM 2002] H. Cunningham, D. Maynard, K. Bontcheva, V. Tablan. GATE: A Framework and Graphical Development Environment for Robust NLP Tools and Applications. Proceedings of the 40th Anniversary Meeting of the Association for Computational Linguistics (ACL'02). Philadelphia, July 2002.

[BONTCHEVA 2013] K. Bontcheva, L. Derczynski, A. Funk, M.A. Greenwood, D. Maynard, N. Aswani. TwitIE: An Open-Source Information Extraction Pipeline for Microblog Text. Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP 2013)

[CARTER 2013] Simon Carter , Wouter Weerkamp , Manos Tsagkias, Microblog language identification: overcoming the limitations of short, unedited and idiomatic text, Language Resources and Evaluation, v.47 n.1, p.195-215, March 2013

[DERCZYNSKI 2013] L. Derczynski, A. Ritter, S. Clark, K. Bontcheva. Twitter Part-of-Speech Tagging for All: Overcoming Sparse and Noisy Data. Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP 2013)

[LAWRENCE 2000] The Double Metaphone Search Algorithm, By Lawrence Phillips, June 1, 2000, Dr Dobb's

[CECCARELLI 2013] Dexter: an Open Source Framework for Entity Linking D. Ceccarelli, C. Lucchese,S. Orlando, R. Perego, S. TraniSubmitted to the Sixth International Workshop on Exploiting Semantic Annotations in Information Retrieval (ESAIR), San Francisco, 2013 [PDF]

[MAVEN] `http://maven.apache.org/`

[CECCARELLI 2013a] D. Ceccarelli, C. Lucchese, S. Orlando, R. Perego, and S. Trani. Learning relatedness measures for entity linking. In Proceedings of CIKM, 2013

[TWITIE] `https://gate.ac.uk/wiki/twitie.html`

[TAGDEF] `http://api.tagdef.com`

[LUCENE] `http://lucene.apache.org/core/`

[SHEN 2012] W. Shen, J. Wang, P. Luo, and M. Wang. Linden: linking named entities with knowledge base via semantic knowledge. In Proceedings of WWW, 2012

[PICCINNO 2014] F. Piccinno, P. Ferragina. From TagME to WAT: a new entity annotator. In Entity Annotation and Disambiguation Challange (ERD): Long track, ACM SIGIR Forum, 2014.

[RIZZO 2014] Giuseppe Rizzo, Marieke van Erp and Raphael Troncy 2014 Benchmarking the Extraction and Disambiguation of Named Entities on the Semantic Web