Detección de plagio intrínseco basada en histogramas

Dario G. Funez y Marcelo L. Errecalde

Laboratorio de Investigación y Desarrollo en Inteligencia Computacional (LIDIC)
Facultad de Cs. Físico, Matemáticas y Naturales, Universidad Nacional de San Luis
e-mail: {dgfunez, merreca}@unsl.edu.ar

Resumen La detección de plagio intrínseco obtiene las secciones de un texto que se sospecha no fueron escritas por el autor del mismo, en base a las variaciones estilográficas observadas en el mismo. En este trabajo, se analiza la factibilidad del uso de histogramas globales para modelar el estilo de escritura de un autor y la detección de outliers, una subtarea de este tipo de detección, que identifica los cambios de estilos en el histograma. El algoritmo de detección fue testeado en el corpus de la Competencia de Detección de Plagio *PAN-PC-2011*, obteniendo un desempeño aceptable en comparación con los otros detectores participantes de la competencia.

Palabras Claves: detección de plagio intrínseco, histogramas.

1. Introducción

Todo trabajo tiene un derecho de autor intelectual y si se utiliza el mismo debe ser reconocida su autoría. Toda utilización del mismo sin consentimiento del autor se denomina *plagio* [11]. El plagio de texto se lleva a cabo cuando un texto está compuesto de fragmentos, cuya propiedad intelectual no le corresponde al autor que se atribuye su autoría. Por ejemplo, un caso de plagio muy frecuente es aquel en que se utilizan fragmentos que son copias exactas de otro texto, siendo éste el más simple de detectar.

Es ampliamente reconocida la importancia de suministrar herramientas informáticas que permitan a los usuarios la *detección automática* de plagio [1]. Esta detección se dificulta cuando, a diferencia de la copia exacta del fragmento original, se modifica el mismo cambiando el orden de las palabras, reemplazando las palabras por sinónimos, sustituyendo oraciones largas por cortas etc. En este contexto, la detección de plagio se puede clasificar en términos generales como *extrínseca* o *intrínseca* [11]. En la primera, se debe proporcionar una colección de documentos de referencia además del texto a analizar y se provee como resultado las secciones en donde se produjo el plagio y la sección que se utilizó en el archivo de referencia. La detección intrínseca en cambio, no utiliza un conjunto de archivos de referencia y realiza un análisis de estilo en el texto para detectar

las variaciones estilográficas. La salida del detector en este caso, son los fragmentos de texto que estilísticamente se desvían del modelo observado a lo largo del texto.

La detección de plagio intrínseco es una tarea relativamente nueva e importante que ha captado la atención de distintos grupos de investigación, como ha quedado demostrado en el último Congreso Internacional de Plagio PAN-PC-2011. En esta competencia, la propuesta presentada en [9] ganó la competencia del 2011 con un detector que utiliza desviación de parámetros de segmentos de texto con respecto al estilo de escritura del documento completo. La comparación entre los segmentos y el texto completo se realiza utilizando vectores de frecuencias de palabras y se calcula un valor de referencia del estilo del autor que se obtiene con el promedio de todas las diferencias de los segmentos y el texto completo. Los segmentos se clasifican como plagiados si distan de este valor.

Kestemont por su parte [6], obtuvo el 2º puesto en la misma competencia con un modelo estilográfico que comprende la obtención de frecuencias de tri-gramas predefinidos más frecuentes utilizado por los autores en general. La detección de outliers se consigue con una matriz de distancia que almacena la distancia entre cada ventana con todas las demás.

Los autores de [4] por su parte, en base a los buenos resultados obtenidos con el uso de histogramas en atribución de autoría, sugieren su utilización para modelar estilos de escritura en detección de plagio. Para obtener los histogramas, utilizan el framework *lowbow* [7], el cual ha sido empleado en diferentes aplicaciones como la visualización de documentos, la segmentación de películas en capítulos y escenas y la segmentación discursiva entre otros.

En este trabajo, analizamos la factibilidad de la representación con histogramas utilizada en [4], en la detección de plagio intrínseco. De acuerdo a nuestro conocimiento, esta técnica no ha sido utilizada previamente para la modelización de estilos de escritura en tareas de detección de plagio intrínseco. El modelo estilográfico en este caso, difiere significativamente de otros modelos utilizados previamente, ya que cuenta con mucha más información, no solamente respecto a puntos aislados, sino respecto a la gráfica completa que representa el histograma. En los métodos de detección de outliers más conocidos, sólo se suele disponer de un conjunto de puntos aislados y se eligen los puntos distinguibles del resto.

Un detector intrínseco, en general, debe definir las siguientes tres etapas: la descomposición del texto, la construcción del modelo estilográfico y la detección de outliers [11]. En nuestro caso, las dos primeras etapas las realiza el framework lowbow, al que se le suministra la cantidad de muestras determinadas por un algoritmo de segmentación de texto; luego, la detección de outliers elige los puntos extremos de la gráfica del histograma. Como se verá en la Sección 6, los resultados obtenidos en el corpus PAN-PC-2011 ofrecen evidencia que el uso de histogramas es una buena opción para el modelado estilográfico.

El resto del trabajo se compone de las siguientes secciones. En la Sección 2 se describen aspectos generales de la detección intrínseca. La Sección 3 explica el algoritmo de segmentación de texto empleado en este trabajo. En la Sección 4 se describe el framework lowbow. En la Sección 5 se presentan los detalles de

implementación del detector propuesto. La Sección 6 describe los experimentos realizados y los resultados obtenidos. Finalmente, en la Sección 7 se ofrecen las conclusiones y el trabajo futuro.

2. Detección de Plagio Intrínseco

El problema de la detección de plagio intrínseco puede enunciarse de la siguiente manera: dado un texto t de un único autor, identificar las secciones de t que han sido redactadas por otros autores [11]. Se asume que el texto fuente fue escrito por un único autor, ya que si se considerara que varios autores colaborativamente escribieron el texto, la complejidad de la tarea de detección se incrementaría.

La detección de plagio intrínseco ha ganado una relevancia creciente, ya que no necesita disponer del conjunto de archivos de donde posiblemente se extrajo la información. Este conjunto, a veces es imposible de obtener dado que existe información que no está disponible en la Web o en otros medios digitales y por lo tanto no se puede llevar a cabo la comparación con dichos archivos. La desventaja que tiene este tipo de detección, es que no puede asegurar con precisión la prueba de plagio, al no tener el documento fuente.

El análisis intrínseco se fundamenta en el hecho de que cada autor tiene un estilo de escritura propio que se mantiene a lo largo de todo el texto [2]. Este estilo de escritura del autor necesita estar representado por un *modelo* que suele estar compuesto de *medidas estilográficas* que caracterizan el estilo de escritura individual del autor [11]. La detección intrínseca utiliza las variaciones significativas del modelo para obtener las secciones plagiadas. En este problema, la única clase que se conoce es la asociada con el estilo de escritura del autor, por lo que se lo considera como un problema de una *única clase* (*one class clasification*) [12] y los restantes estilos de escritura son denominados *outliers*. En [11] por su parte, se divide la tarea de detección intrínseca en las siguientes 3 subtareas: a) *estrategia de descomposición*, b) *construcción del modelo de estilo* y c) *identificación de outliers*, que se describen a continuación.

2.1. Estrategia de descomposición

El documento completo necesita ser dividido para obtener información del estilo de escritura en diferentes puntos del texto. La estrategia elegida en este caso se debe seleccionar con mucho cuidado, ya que el desempeño del detector dependerá fuertemente de esta etapa. Las estrategias más simples pueden ser más sencillas de implementar y más rápidas en ejecución, pero no siempre obtienen una buena performance, como es el caso de dividir el texto en bloques del mismo tamaño. La división del texto en límites *estructurales* (como por ejemplo *párrafos* o *sentencias*) puede ser una mejor opción, debido a que las secciones plagiadas suelen ser párrafos completos. Una alternativa que ha logrado resultados interesantes consiste en dividir el texto en segmentos coersivos utilizando un algoritmo de *segmentación de texto*, como hemos propuesto previamente en [5].

2.2. Construcción del modelo de estilo

Para detectar variaciones en el estilo de escritura se necesita un modelo con información estilográfica del autor del texto. Un escritor, inconcientemente mantiene en todo sus escritos un mismo modelo, por lo que una variación significativa en el mismo hace sospechar que se esta usando otro estilo de escritura y que posiblemente un autor no referenciado sea quien lo escribió. El modelo estilográfico más común, se construye con un conjunto de medidas estilográficas, como por ejemplo *índices de legibilidad*, frecuencias de clases de palabras como *adjetivos* y *sustantivos*, la *riqueza de vocabulario*, etc [11]. Sin embargo, una opción interesante, y la propuesta en este trabajo, consiste en representar un estilo de escritura mediante histogramas globales de palabras, la cual ha demostrado ser efectiva en tareas de atribución de autoría [4].

2.3. Identificación de outliers

Con la información de la etapa anterior, se caracteriza el estilo de escritura del redactor en diferentes puntos del texto. En un texto escrito por un único autor, el estilo de escritura debe permanecer con alteraciones mínimas en todo el texto completo. Su modelo es la única información con que se cuenta y de esta manera se define un problema de clasificación de una única clase [12], ya que no es posible caracterizar el estilo de escritura de todos los posibles autores. A este tipo de problemas también se los denomina como *detección de outliers* ya que se deben elegir aquellos puntos en el texto en donde el estilo de escritura es significativamente diferente a las demás muestras extraídas del texto [11].

Los métodos de detección de outliers se pueden clasificar en tres categorías:

- *Métodos de densidad*: aproximan la función de densidad de probabilidad de la clase objetivo. Se considera a los outliers uniformente distribuidos y la regla de Bayes se puede utilizar para diferenciar objetos outliers de los objetivos. Estos métodos proveen buenos resultados cuando el tamaño de la muestra es grande.
- *Métodos de límite*: tratan de delimitar una región utilizando distancias entre los elementos objetivos. Los outliers son aquellos objetos que no están comprendidos en esa región.
- *Métodos de reconstrucción*: necesitan conocimiento previo sobre la generación de los elementos objetivos. Los outliers son aquellos objetos que son difíciles de reconstruir.

En la siguiente subsección, se definen las medidas de evaluación generalmente empleadas para cuantificar el desempeño de un detector de plagio.

2.4. Medidas de evaluación

Para evaluar el comportamiento de un algoritmo de detección de plagio, se deben computar la *precisión* (en inglés *precision*), la *cobertura* (*recall*) y la

granularidad de las detecciones realizadas [8]. En la definición de estas medidas seguiremos las siguientes convenciones de notación: 1) s representa una sección plagiada del conjunto S de todas las secciones plagiadas, 2) r denota una sección detectada del conjunto R de detecciones, 3) S_R son las secciones plagiadas que han sido detectadas, 4) $\mid s_i \mid$ y $\mid r_i \mid$ denotan el tamaño (en cantidad de caracteres) de la sección correspondiente y $\mid S \mid$ y $\mid R \mid$ denotan la cardinalidad de los conjuntos respectivos. Finalmente, $\alpha(s_i)$ es la cantidad de caracteres detectados de s_i, $\beta(r_i)$ es la cantidad de caracteres plagiados de r_i y $\gamma(s_i)$ es la cantidad de caracteres plagiados detectados de s_i. En base a estos valores, la precisión, cobertura, granularidad y evaluación global (*overall*), se definen de la siguiente manera[1]:

$$recall = 1/\mid S \mid \sum_{i=1}^{|S|} \alpha(s_i)/\mid s_i \mid \tag{1}$$

$$precision = 1/\mid R \mid \sum_{i=1}^{|R|} \beta(r_i)/\mid r_i \mid \tag{2}$$

$$granularidad = 1/\mid S_R \mid \sum_{i=1}^{|S_R|} \gamma(s_i) \tag{3}$$

$$overall = F/(\log_2(1 + granularidad)) \tag{4}$$

Estas medidas se interpretan de la siguiente manera. La precisión cuantifica el porcentaje de detecciones correctas, el recall el porcentaje de plagio detectado, una granularidad cercana a 1 significa que el algoritmo detectará cada plagio a lo sumo una vez. En todos los casos, valores cercanos a 1 indican que el algoritmo de detección tiene un buen desempeño.

3. Segmentación de texto

La segmentación de texto divide un texto en unidades con el mismo tópico [3]. La implementación de Freddy Choi es realizada en dos fases sobre el texto completo. En la primer etapa las *stops words* (artículos, preposiciones, conectores etc.) son removidas ya que no aportan información relevante del texto. La raíz de cada palabra se obtiene mediante un algoritmo de *stemming* y se almacena su frecuencia en el texto en un vector. Cada oración tiene asociada un vector y la frecuencia de la palabra j en la oración i se denota como $f_{i,j}$.

La matriz S resultante de aplicar la similitud coseno a cada par de vectores es llamada *matriz de similitud* [3]. Dado que no es sencillo determinar los límites de los segmentos directamente sobre S, esta matriz es sometida a un proceso de *ranking* que obtiene una nueva matriz S' a partir de S, denominada *matriz de rango*. Cada elemento (valor) de la matriz S' resulta de desplazar una *máscara* (matriz cuadrada) sobre S. Cada valor r en S' se determina en base al conjunto

[1] En la evaluación global, F refiere a la tradicional medida F, la media harmónica de precisión y recall: $F = 2 \times (precision \times recall)/(precision + recall)$

de valores que cubre la máscara en S (N) y al valor central de la máscara en S (c). La fórmula para obtener r es: $r = lvalue/|N|$, donde $lvalue$ es el número de elementos en N con menor similitud que c.

La última etapa del algoritmo de segmentación, utiliza los valores obtenidos en S' y aplica un método de clustering divisivo, basado en el algoritmo de maximización de Reynar [10] para detectar los límites de los segmentos. Este algoritmo se basa en el concepto de *densidad interna* donde, dado un segmento delimitado por las sentencias i y j (inclusive); si $s_{i,j}$ es la suma de los valores rango de las sentencias en el segmento y $a_{i,j}$ es el área interna que abarca el segmento dada por la fórmula: $a_{i,j} = (j - i + 1)^2$. Si $B = b_1 \dots b_m$ es una lista de m segmentos coherentes y s_k y a_k denotan la suma de valores rango y área respectivamente, correspondiente al segmento k en B, la densidad interna de B se define como:

$$D = \sum_{i=1}^{m} s_k / \sum_{i=1}^{m} a_k \tag{5}$$

El proceso comienza inicializando B con un único segmento que representa todo el documento. Cada paso del algoritmo separa uno de los segmentos en B y el punto de corte se elige de tal manera que maximiza D. La cantidad de segmentos m se determina de forma automática y queda establecida cuando el gradiente tiene variaciones inusuales. Si $D^{(n)}$, es la densidad interna de n segmentos, el gradiente se define como: $\delta D^{(n)} = D^{(n)} - D^{(n-1)}$.

Para un documento con b límites potenciales, si u,v denotan la media y varianza de $\delta D^{(n)}$ con $n \in 2, \dots, b + 1$, el m queda definido al aplicar el threshold $u + l\sqrt{v}$ a δd. A menudo, un valor de $l = 1,5$ es utilizado en la práctica.

4. Lowbow

Lowbow es un framework en Matlab que provee una representación secuencial, diferenciable y contínua de un documento [7]. Ha sido utilizado con éxito en problemas de atribución de autoría, donde se emplearon histogramas locales de n-gramas [4]. Lowbow requiere como entrada: a) el archivo a analizar, b) la cantidad de fragmentos o muestras en que lowbow va a dividir el texto y c) un conjunto de *kernels* cuyos valores oscilan entre 0 y 1.[2] El framework permite obtener, dado un archivo con el texto, histogramas con diferentes kernels. Un kernel con valor cercano a 0 preserva la información secuencial y ésta se va perdiendo a medida que el valor se incrementa. También se le puede proporcionar el vocabulario, pero éste es un parámetro opcional, calculándolo el framework en forma automática a partir del texto, en caso de no ser especificado. Lowbow devuelve como resultado, archivos con distinto tipo de información relacionada a los histogramas, como por ejemplo:

- Cantidad de muestras y los kernels (archivo *lowbow.info*). Se utiliza para visualizar los histogramas.

[2] Razones de espacio impiden una explicación detallada de los kernels, pero el lector interesado puede consultar [7] para mayores detalles.

- Información sobre el análisis de componentes principales (archivos **.proj*).
- Información sobre velocidad, curvatura, vector tangente, etc (archivos **.tan*).

La representación con histogramas provee de información secuencial, que es importante para modelar el estilo de escritura de un autor, y que será utilizada en el detector que se describe en la próxima sección.

5. Descripción del detector

En la Figura 1 se muestran todos los pasos involucrados en nuestra propuesta de detección de plagio intrínseca.

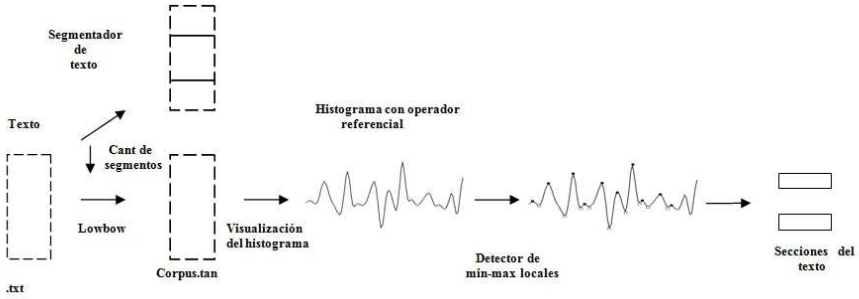

Figura 1. Arquitectura del detector.

Para utilizar lowbow, se le debe proporcionar la cantidad de muestras en el texto. Este valor se obtiene al aplicar al documento completo que se desea chequear, el algoritmo de segmentación de texto explicado en la Sección 3, ya que éste permite determinar la cantidad de segmentos presentes en el mismo. Se realizaron pruebas con diferentes técnicas de descomposición de texto, como dividir el texto en segmentos de tamaño uniforme y en párrafos pero se obtuvo un desempeño inferior. Por lo tanto, y al igual que en [5] donde ya habíamos obtenido buenos resultados con la segmentación de texto, se aplicó el algoritmo de segmentación de Freddy-Choi que forma parte de la librería Morphadorner.[3]

A continuación, se ejecuta lowbow con los siguientes parámetros: a) la cantidad de muestras (determinada mediante el procedimiento antes explicado) y b) el kernel, el cual fué determinado experimentalmente con un valor de 0.03. El lowbow convencional produce con los parámetros anteriores distintos archivos de salida, que corresponden a distintos histogramas. De todos los posibles histogramas se seleccionó aquel que mejor representó el modelo estilográfico, en un conjunto de archivos de referencia.

[3] *Morphadorner* es una librería java de acceso libre para PLN suministrada por la Universidad de Northwestern.

La Figura 2 muestra un histograma obtenido por lowbow el cual, como se puede observar, se presenta como una función matemática contínua.

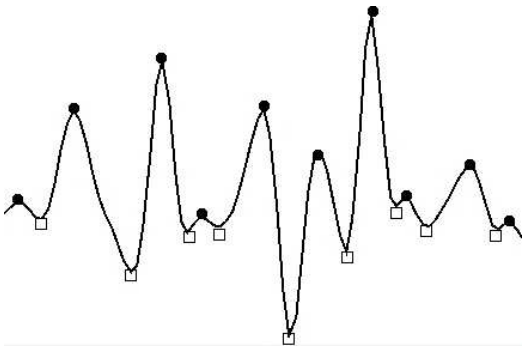

Figura 2. Histograma de palabras.

La detección de outliers identifica aquellas secciones del texto en el cual se producen variaciones significativas en la gráfica del histograma. Para tal fin, se empleó el script *Peakdetect* en Matlab, el cual detecta los picos que se encuentren en una gráfica, es decir, los mínimos y máximos locales, como se muestran en la Figura 2. Todos los fragmentos del texto que muestran estos comportamientos anómalos son sospechosos de plagio.

Una vez que se cuenta con las secciones provistas por la detección de outliers, se debe verificar si existen secciones adyacentes para unirlas en una única sección, y así obtener una mejor granularidad. La salida del detector es un archivo *.xml* con la información de las secciones del documento que se sospecha de plagio. Cada sección consta de la información donde se posiciona en el texto, es decir, el desplazamiento desde el comienzo del texto en cantidad de caracteres y su tamaño. A continuación se exponen las diferencias del detector propuesto con el presentado en [5]:

- La segmentación de texto se utiliza para obtener la cantidad de segmentos en el texto requerida por lowbow, pero no se utilizan las secciones provistas por el algoritmo.
- El modelo estilográfico del texto completo se representa con un único histograma, mientras que en [5] se calcula un conjunto de medidas estilográficas cuidadosamente seleccionadas en cada segmento del documento.
- La detección de outliers en este trabajo es muy innovadora, ya que detecta los puntos outliers en el texto cuando se producen alteraciones en la gráfica del histograma. En el anterior, la detección de outliers se realiza utilizando el método de detección basado en la Meda.

6. Experimentos

Para evaluar el comportamiento del detector propuesto, se utilizó el corpus PAN-PC-2011 suministrado para la competencia [8], utilizándose el script Python *perfmeasure.py*, para computar la precisión, el recall, la granularidad y el *Plagdet score* u overall. El corpus esta compuesto por una colección de 4.753 archivos de diferentes tamaños y con distinta cantidad de secciones plagiadas. En la siguiente tabla, se muestran los resultados totales de aplicar el detector a los archivos del corpus de la competencia.

plag-det	Recall	Precisión	Granularidad
0.088486	0.167477	0.063944	1.064707

Tabla 1. Resultados totales obtenidos con el corpus PAN-PC-2011.

A continuación, se muestra en la Tabla 2 los resultados comparativos de todos los competidores del PAN2011 en la detección de plagio intrínseca.

Puesto	plag-det	Recall	Precisión	Granularidad
1	0.3254817	0.3397965	0.3123243	1.0000000
2	0.1679779	0.4279112	0.1075817	1.0329386
3	0.0841286	0.1277831	0.0664302	1.0549085
4	0.0693820	0.1080543	0.0783903	1.4787234

Tabla 2. Resultados de todos los participantes de la competencia PAN-PC-2011

Como se puede observar, nuestro detector se posicionaría en el tercer puesto, con un mejor *recall* que el obtenido por el tercer lugar. Sin embargo, una de las deficiencias observadas en el detector propuesto, es que en varios casos devuelve secciones plagiadas cuando el documento no tiene plagio (falsos positivos). Esta falencia disminuye el desempeño global del detector, ya que el corpus está compuesto por un gran porcentaje de archivos sin plagio.

7. Conclusiones y Trabajo Futuro

A partir de la experimentación realizada, se ha obtenido evidencia firme de la factibilidad de la modelización del estilo de escritura mediante histogramas en la detección de plagio intrínseco. Si bien los resultados presentados son aún preliminares y pueden ser mejorados en el futuro, el enfoque propuesto en este trabajo ha demostrado ser en este momento, competitivo con respecto a otros algoritmos representativos del estado del arte en la detección de plagio intrínseco.

El algoritmo propuesto tiene la ventaja que es muy rápido, ya que la obtención del modelo se reduce a recuperar un histograma de todo el documento. La detección de outliers también es muy simple y eficiente y contribuye satisfactoriamente al desempeño global del detector.

Como trabajos a futuro, una posibilidad es utilizar como vocabulario en lowbow, los *tri-gramas* en lugar de las palabras completas, una variante que ya

ha producido buenos resultados en atribución de autoría [4]. También se planea, modificar el código de lowbow para que utilice información sobre las secciones de texto provistas por el algoritmo de segmentación de texto. Además, y como objetivo inmediato, el trabajo estará dirigido a mejorar el desempeño del detector en el tratamiento de los documentos que no tengan plagio, mediante un análisis más detallado de las variaciones de los picos de los histogramas.

Referencias

1. Enrique Vallés Balaguer. Empresa 2.0: Detección de plagio y análisis de opiniones. Master's thesis, Universidad Politécnica de Valencia, 2011.
2. Chien-Ying Chen, Jen-Yuan Yeh, and Hao-Ren Ke. Plagiarism detection using rouge and wordnet. *CoRR*, abs/1003.4065, 2010.
3. Freddy Y.Y. Choi. Advances in domain independent linear text segmentation. In *ANLP*, pages 26–33. The first conference on North American chapter of the Association for Computational Linguistics, Morgan Kaufmann, 2000.
4. Hugo Jair Escalante, Thamar Solorio, and Manuel Montes y Gómez. Local histograms of character n-grams for authorship attribution. In *ACL*, pages 288–298. The Association for Computer Linguistics, 2011.
5. Dario G. Funez and Marcelo L. Errecalde. Un ambiente de ejecución para la detección de plagio intrínseco usando la segmentación de texto. *Cacic 2011*, 2011.
6. Mike Kestemont, Kim Luyckx, and Walter Daelemans. Intrinsic plagiarism detection using character trigram distance scores. In *CLEF (Notebook Papers/Labs/Workshop)*, 2011.
7. Guy Lebanon, Yi Mao, and Joshua Dillon. The locally weighted bag of words framework for document representation. *Journal of Machine Learning Research 8 (2007) 2405-2441*, 2007.
8. Alberto Barrón-Cedeño Benno Stein Martin Potthast, Andreas Eiselt and Paolo Rosso. Overview of the 3rd international competition on plagiarism detection. *Notebook Papers of CLEF 2011 Labs and Workshops*, 2011.
9. Gabriel Oberreuter, Gaston LHuillier, Sebastián A. Ríos, and Juan D. Velásquez. Approaches for intrinsic and external plagiarism. In *CLEF (Notebook Papers/Labs/Workshop)*, 2011.
10. Jeffrey C. Reynar and Adwait Ratnaparkhi. A maximum entropy approach to identifying sentence boundaries. In *Proceedings of 5th Conference of Applied NLP*, pages 16–19, 1997.
11. Benno Stein, Nedim Lipka, and Peter Prettenhofer. Intrinsic plagiarism analysis. *Language Resources and Evaluation*, 45(1):63–82, 2011.
12. David M. J. Tax. *One-class classification; Concept-learning in the absence of counter- examples*. PhD thesis, Delft University of Technology, 2001.