Detección automática de repeticiones parciales en repositorios digitales

Trabajo práctico final de la materia
Bibliotecas y Repositorios Digitales
Mayo 2020

Docente: Dra. Marisa R. de Giusti

Nahuel González

Laboratorio de Sistemas de Información Avanzados
Facultad de Ingeniería, Universidad de Buenos Aires
Ciudad Autónoma de Buenos Aires, Argentina
ngonzalez@lsia.fi.uba.ar

DRAFT

***Resumen*—A medida que crece el tamaño de los repositorios digitales y otros tipos de archivo digital, ya sea gestionados por instituciones públicas o empresas privadas en el transcurso de su funcionamiento diario, la posibilidad de verificar la ausencia de repeticiones en el contenido en forma manual y exhaustiva deja de ser posible. Las repeticiones puede ser maliciosas, como en la copia y el plagio, o involuntarias; en ambos casos, tanto la persecución de la calidad del contenido como la minimización del esfuerzo de conservación invitan a su detección temprana por medio de herramientas automáticas. El caso más simple de repetición lo constituye una repetición total; más sutiles son los casos de repetición parcial, o no aparentes como dos archivos de audio en distinta calidad. Sin embargo, el primero dista de ser trivial si el tamaño del repositorio no es pequeño. No sólo los manuscritos son pasibles de repetición total y parcial, o diversos grados de similaridad, sino también muchos otros tipos de objetos digitales como el código fuente. Asimismo, dos colecciones pueden contener archivos a los fines prácticos completamente idénticos pero cuyos bitstreams difieren notoriamente. En este trabajo se reseñarán las técnicas utilizadas con éxito para la detección de repeticiones totales y parciales en repositorios digitales, con énfasis en el análisis de textos, y se discutirán los resultados de aplicar GERMÁN, COMPLETAR sobre el repositorio del SEDICI.**

I. Introducción

A medida que crece el tamaño de los repositorios digitales y otros tipos de archivo digital, ya sea gestionados por instituciones públicas o empresas privadas en el transcurso de su funcionamiento diario, la posibilidad de verificar la ausencia de repeticiones en el contenido en forma manual y exhaustiva deja de ser posible. Las repeticiones puede ser maliciosas, como en la copia y el plagio, o involuntarias; en ambos casos, tanto la persecución de la calidad del contenido como la minimización del esfuerzo de conservación invitan a su detección temprana por medio de herramientas automáticas.

El caso más simple de repetición lo constituye una repetición total; un archivo que se encuentra en dos o más colecciones. Más sutiles son los casos de repetición parcial, o no aparentes como dos archivos de audio en distinta calidad. Sin embargo, el primero dista de ser trivial si el tamaño del repositorio no es pequeño. El desafío que presenta consiste en que durante el flujo normal de trabajo humano el conocimiento sobre qué documentos se encuentran relacionados se pierde a menudo [1]. Lo que es más, si se trata de colecciones de millones de documentos (muchos de los cuáles no contienen metadatos o los contienen en forma incompleta) que son periódicamente unificadas o divididas, el costo computacional y a veces el tiempo requerido para la identificación de repeticiones no es despreciable y se debe recurrir a técnicas escalables que consideran fragmentos de archivos embebidos en grafos bipartitos [1] o árboles de sufijos [2].

La detección de repeticiones parciales es relevante para la detección temprana del plagio. Una política institucional que exija autoarchivo de las publicaciones producidas por sus investigadores en combinación con herramientas y procesos de detección de repeticiones parciales en textos y otros objetos digitales puede prevenir en ocasiones el plagio malicioso o el no intencional [3]. Considérese, por ejemplo, la infame instancia de plagio flagrante en la introducción de [4]:

"La arquitectura dirigida por modelos (MDA) y las

ontologías son dos de los recursos cada vez más populares dentro de la comunidad informática para el desarrollo de sistemas de software. La MDA presenta un marco de trabajo para crear aplicaciones informáticas. Paralelamente, las ontologías devienen en recursos cuya función es facilitar la interacción entre herramientas de software heterogéneas."

que es un mera paráfrasis, indisimulada y carente de referencias al artículo original [5] donde podemos leer:

"La arquitectura dirigida por modelos (MDA) y las ontologías constituyen dos de los recursos más populares dentro de la comunidad informática actual para el desarrollo de sistemas de información. MDA presenta un marco de trabajo para crear soluciones informáticas. A su vez, las Ontologías son recursos para facilitar la interoperabilidad entre herramientas de software heterogéneas."

Configura una jocosa ironía académica el hecho de que precisamente las arquitecturas dirigidas por modelos y las ontologías hayan encontrado extendida utilización para la detección de plagios como el que antecede. Por ejemplo, la identificación de copias en segmentos de tesis puede ser llevada a cabo, con cierto éxito, mediante el entrenamiento de ontologías adecuadas al campo de estudio [6]. Algunas de estas técnicas son lo suficientemente poderosas como para no limitarse a la detección de copia o paráfrasis sin cita adecuada, como en [7], sino también para detectar la más difusa apropiación de ideas [8]. Una interesante reseña de las técnicas existentes hasta hace algunos años puede encontrarse en [9]. Incluso técnicas de vanguardia como las redes neuronales han encontrado aplicación a este problema [10].

Aunque a primera vista parezcan similares, los problemas de detección de duplicaciones internas en repositorios digitales y de descubrimiento de plagio no deben ser tratados en la misma forma. No sólo en el objetivo, donde en el primer caso nos encontramos ante la necesidad de mejorar la calidad del contenido y minimizar los esfuerzos de conservación evitando redundancias, mientras que en el segundo buscamos evitar la inclusión del material que infrige la ética de publicación. La diferencia fundamental es que en el primer caso realizamos comparaciones estrictamente internas dentro de un único repositorio (posiblemente federado, como se describe en la sección siguiente) y esperamos quizás encontrar múltiples versiones de un documento con la misma autoría, o incluso párrafos reutilizados. En el segundo, en contraste, se requieren datos externos de otros repositorios y bases de datos, y se requieren en un volumen que posiblemente supere en al menos un orden de magnitud al encontrado en el repositorio considerado. Si bien las técnicas de detección no difieren sustancialmente, en este trabajo nos concentraremos en la detección de repeticiones o similaridades dentro de un único repositorio, y no la comparación con repositorios externos en busca de plagio.

No sólo los manuscritos son pasibles de repetición total o parcial, o diversos grados de similaridad, sino también muchos otros tipos de objetos digitales. La repetición de código fuente es un problema en crecimiento que afecta sobre todo a las universidades, a medida que más estudiantes copian el software de otras tesis anteriores o de fuentes en internet [11]. Esta práctica no sólo reduce la calidad del resultado sino que también puede producir disputas de propiedad intelectual. Es claramente imposible buscar similaridades en el código fuente en forma manual, pero herramientas como *OWL Web Ontology Language* [12] -que un estándar W3- en combinación con el lenguaje de consulta SPARQL se han mostrado exitosas para eliminar y restringir, o al menos detectar tempranamente, el plagio de código fuente en tesis de grado, maestría, y doctorado [11]. Asimismo, dos colecciones pueden contener archivos a los fines prácticos completamente idénticos pero cuyos bitstreams difieren notoriamente. Dos archivos de audio en distinta calidad, dos imágenes idénticas en distintos formatos, dos documentos que sólo difieran en el título, ejemplifican posibles reiteraciones cuya detección, en los casos en los que fuera posible, demandaría ingenio y gran potencia de cómputo.

En este trabajo se reseñarán las técnicas utilizadas con éxito para la detección de repeticiones totales y parciales en repositorios y bibliotecas digitales, con énfasis en el análisis de textos. En la sección *Detección de archivos idénticos* se tratará brevemente el problema de detección de duplicaciones y dos ejemplos del mundo real en repositorios grandes. La sección *Detección de repeticiones parciales de texto* tratará en mayor detalle la detección de objetos de texto similares pero no idénticos. La sección *Detección de repeticiones en archivos multimedia* reseñará algunas técnicas para realizar la misma tarea en contenido audiovisual. Finalmente, en la sección *Un estudio empírico sobre el repositorio del SEDICI* se discutirán los resultados de aplicar **TODO: GERMÁN, COMPLETAR** sobre el repositorio del SEDICI.

II. Detección de archivos idénticos

En [13] se describen dos implantaciones de la arquitectura de federación de repositorios *aDORe* que almacenan más de cien millones de objetos digitales. Los autores señalan que en entornos que aspiran a la preservación digital de largo plazo, puede llegar a ser problemático asociar estrechamente el identificador interno de un objeto digital con la referencia que establece un protocolo basado en URIs. Esto se debe a que, en la práctica, las URLs de acceso a los objetos del repositorio cambian con el tiempo, por motivos varios que abarcan desde aspectos técnicos hasta aquellos relacionados con las políticas de custodia, mientras los identificadores internos pueden sobrevivir intactos incluso al migrar entre sistemas de manejo de contenidos.

Por este motivo, el uso de esquemas de identificación no basados en un protocolo permite definir una identidad global [14]. Cuando múltiples repositorios federados almacenan varias copias del mismo objeto y utilizan el mismo identificador interno no basado en la URL de acceso, todas pueden reconocerse sin ambigüedad; cosa que sería imposible al utilizar una URI dependiente del protocolo o de la localización. Es idéntico el escenario en el que un único repositorio guarda múltiples copias de un objeto con el

DRAFT

mismo identificador, por lo que es interesante estudiar para este propósito esquemas como *info*, *ARK*, o *tag*.

Cuando no se cuenta con un esquema de identificación consistente, o el mismo es inadecuado para la tarea por motivos de implementación, se debe recurrir necesariamente a la generación de hashes para detección de archivos idénticos. Sin embargo, al ocuparse de objetos digitales de gran tamaño y colecciones de decenas de millones de objetos incluso este proceso puede resultar prohibitivo. Un enfoque basado en la fragmentación del *byte stream* y la localización precisa de huellas únicas dependientes del tipo de archivo permite no sólo incrementar la escalabilidad sino reducir los tiempos de análisis y los costos del proceso de detección de repeticiones.

Por ejemplo, en [1] se describe la aplicación de un esquema de este tipo, capaz de descubrir documentos idénticos sin utilizar metadatos ni identificadores, sólo empleando pequeños fragmentos específicos al tipo de archivo y un algoritmo de partición de grafos. Este es aplicado a la fusión y limpieza periódica dentro de una colección de muchos millones de documentos técnicos y de soporte, cuyo versionado no es exhaustivo y entre los cuáles pueden aparecer duplicados y el mismo documento en múltiples formatos. Los autores enfatizan la necesidad de complementar la detección automática con una revisión manual posterior.

III. Detección de repeticiones parciales de texto

Detectar repeticiones completas es, en principio, una tarea sencilla. El proceso puede resumirse en tres etapas; una primera en la cuál se calculan los *hashes* de los objetos digitales considerados, una segunda de almacenamiento, y una tercera de detección de repeticiones en los *hashes*. En contraste, la detección de repeticiones parciales o similaridades elevadas es una tarea no sólo de mucho mayor complejidad sino también con mucha mayor demanda de almacenamiento y potencia computacional. El almacenamiento crece pues los conjuntos de características extraídas y las estructuras de datos requeridas exigen más información que un simple entero, y la potencia computacional necesaria refleja el hecho de que las posibles variaciones que generen una repetición parcial son numerosísimas.

Por este motivo, el proceso de detección de repeticiones parciales suele realizarse en dos grandes etapas. En primer lugar se utiliza un esquema de indizado y búsqueda para la selección rápida de un conjunto reducido de candidatos en una colección, mientras que en segundo lugar se aplican técnicas de comparación exhaustiva, mucho más costosas, sobre el antedicho conjunto. Existen dos familias principales de técnicas para la selección de candidatos: los métodos basados en *rankings* y aquellos basados en la determinación de huellas (del inglés *fingerprinting*) [15]. El proceso de detección exhaustiva es mucho más variado.

A continuación se describirán algunos ejemplos de métodos basados en ranking y en huellas que han sido aplicados con éxitos a repositorios digitales. Luego, se explica la comparación detallada final y finalmente se advierte sobre las diferencias introducidas por el tipo de texto.

III-A. Métodos basados en ranking

Los métodos basados en ranking, similares a los utilizados por motores de búsqueda, comienzan fragmentando los textos en palabras o *tokens* para generar un léxico y un conjunto de listas invertidas. El léxico almacena las palabras que aparecen en el texto mientras que las listas invertidas almacenan estadísticas de repetición, frecuencia, o cercanía a otras palabras. Ambos son utilizados para generar un índice que será luego recorrido por el motor de búsqueda. Distintas técnicas para almacenar el índice de forma tal que sea rápido de acceder y a la vez escalable para una enorme cantidad de documentos han sido utilizadas, como los *d-gaps*, u otros esquemas más sofisticados para la compresión de secuencias de enteros. A su vez, la información de términos almacenada en dicho índice puede ser pesada de acuerdo a la frecuencia dentro de cada documento y a la frecuencia global, a los fines de mejorar la relevancia de los resultados pues la identificación de términos menos comunes permite seleccionar en primer lugar aquellos que logran mayor poder de discriminación. También pueden penalizarse los conteos de palabras de acuerdo a la extensión del documento para contrarrestar el hecho de que en documentos más largos cada palabra ocurrirá naturalmente más veces.

En un motor de búsqueda tradicional, la consulta consiste en un conjunto de términos clave y posiblemente otros campos para restringir la búsqueda. Es diferente el caso que aquí nos ocupa, en donde la consulta o entrada al motor consiste en un documento completo y la salida en el subconjunto de documentos indizados que presentan un grado de similaridad suficientemente elevado. A pesar de esta considerable diferencia, el índice utilizado suele ser del mismo tipo; sí difieren el tipo de métricas utilizadas para la jerarquización de los resultados, y se agrega una segunda fase en donde sólo los resultados más relevantes se evalúan exhaustivamente contra el documento de origen [16]. De esta forma, el proceso de evaluación exhaustiva que sería prohibitivo de realizar contra todos los elementos de la colección es realizado exclusivamente contra aquellos de los cuáles hay motivos para sospechar que esconden una similaridad significativa.

El motor de búsqueda para descubrir archivos similares o repeticiones parciales requiere utilizar un conjunto de métricas para seleccionar y ordenar los resultados de la consulta realizada que difiere de su contraparte, la búsqueda basada exclusivamente en palabras clave [16]. Algunos de ellos incluyen comparaciones del histograma completo de términos individuales o n–gramas (en este contexto, conjunto de dos o más palabras sucesivas) utilizando distancias especializadas como PlagiRank o *hashing* localmente sensible. Esta última técnica será descrita en más detalle en la sección sobre archivos multimedia.

El algoritmo PlagiRank [17], si bien ha sido propuesto inicialmente para la detección de plagio, es útil para la

DRAFT

detección de documentos versionados y repeticiones parciales. Contrariamente a otras métricas de similaridad como las basadas en la distancia del coseno u Okapi BM25, más útiles para la búsqueda basada en términos, PlagiRank se adapta muy bien al descubrimiento de similaridades fuertes en textos pues, en contraste con las anteriores, no premia los documentos con mayor cantidad de apariciones de unos pocos términos de búsqueda sino que intenta capturar una similaridad global aunque no exacta en la cantidad de ocurrencias de palabras o n–gramas, y sus posiciones relativas en el histograma.

III-B. Métodos basados en huellas

Contrariamente a los métodos basados en ranking, que descomponen e indizan el texto de entrada, los métodos basados en huellas (fingerprinting) buscan representar de manera compacta cada documento a ser comparado, para luego aplicar una métrica de similaridad a estas representaciones acotadas. Cada huella se compone de un conjunto fijo de atributos que representan los aspectos clave del documento procesado, con suficiente granularidad para expresar relaciones de texto no superficiales.

Uno de los primeros esquemas de detección de similaridad mediante el uso de huellas que ha demostrado ser a la vez efectivo y escalable es el propuesto en [18] ya en 1996. El autor utiliza las denominadas *huellas selectivas de tamaño fijo*, que consisten en la elección cautelosa de fragmentos representativos del texto y reducen el enfoque a la construcción de una estrategia de selección y a la optimización del tamaño de las huellas para una colección dada. De esta forma logra cumplir con requerimientos de almacenamiento y tiempos de comparación inicial sorprendente bajos sin incrementar la tasa de falsos positivos en contraste con los casos base. Es interesante notar que el tipo de preprocesamiento utilizado hace a este esquema resiliente al ruido de conversión; de esta forma, textos similares que han pasado por procesos de conversión diversos como, por ejemplo, la conversión entre PDF, PostScript, o texto plano, no presentan dificultades. De esta forma la migración sucesiva a nuevos formatos eventualmente requerida por el proceso de preservación a largo plazo no colisionaría con la posibilidad de detectar reduplicaciones.

En [2] se propone un sistema de detección de duplicaciones parciales para bibliotecas digitales en tres etapas que los autores denominan MDR, acrónimo inglés de encontrar/detectar/revelar (MatchDetectReveal). Con cambios menores este puede servir tanto para detectar duplicaciones internas en un repositorio como plagio si se utiliza el mismo contra colecciones externas. La huella se contruye en base a los árboles de sufijos de los documentos procesados, utilizando una versión modificada del algoritmo de Ukkonen (ver subsección *Comparación detallada final*).

Border [19] extiende la noción de similaridad o repetición parcial para incluir un nuevo tipo de relación de *inclusión parcial*; es decir, cuando uno de los documentos se encuentra parcialmente incluído, posiblemente con modificaciones, en otro de mayor extensión en forma tal que una comparación de similaridad arrojaría un resultado negativo. El esquema propuesto se basa en la reducción a un conjunto de problemas de intersección de conjuntos que pueden resolverse de manera aproximada utilizando muestreo aleatorio y huellas de Rabin [20]. A pesar de ser un abordaje más flexible, presenta dificultades de escalabilidad para colecciones muy grandes.

III-C. Comparación detallada final

Tanto los métodos pertenecientes a las familias de ranking y huellas como otros que sirven al mismo propósito sirven a los fines de encontrar rápidamente un conjunto de candidatos, que luego deben ser procesados en mayor detalle con algoritmos de mayor costo computacional pero menor tasa de error. El problema general se denomina detección de coincidencias aproximadas (del inglés *approximate string matching*) y goza una frondosa literatura que data de los inicios de la computación (ver, por ejemplo, la reseña [21]). Reseñarla excede el alcance del presente trabajo.

El *alineamiento local* es una técnica que ha sido empleada con éxito para la comparación detallada final en repositorios de gran tamaño [22]. Originalmente concebida en el campo de la bioinformática para rastrear similaridades en el código genético y así descubrir organismos homólogos, ha encontrado posteriormente utilidad en la detección de coincidencias aproximadas en textos, tanto de lenguaje natural como de código fuente.

III-D. El problema que introduce el tipo de texto

Las técnicas y algoritmos reseñados más arriba deben considerarse en el marco específico de la detección de repeticiones en textos compuestos de lenguaje natural. Otros tipos de archivos, como puede ser el código fuente, requieren abordajes particulares pues de lo contrario las tasas de error en la detección se incrementan sensiblemente aunque en todos los casos se trate de texto y no de otro tipo de contenido [22].

IV. Detección de repeticiones parciales en archivos multimedia

No sólo en el contenido de los archivos de texto o en los objetos monolíticos pueden buscarse repeticiones. Técnicas de mayor complejidad, y que también requieren mayor potencia de cómputo, pueden utilizarse para descubrir repeticiones totales o parciales en archivos multimedia. La dificultad adicional que el contenido multimedia detenta se debe no sólo a la variedad de formatos en los que este puede presentarse sino también al hecho de que este puede estar codificado o comprimido con diferentes esquemas, cuyo resultado al descomprimir puede introducir suficiente variación del contenido para confundir comparaciones inocentes.

IV-A. Audio

En el caso del audio, el conjunto de técnicas de identificación de contenidos similares pero sujetos a ruido,

DRAFT

a artefactos de compresión/descompresión, o a pérdidas de calidad por codificación con menor resolución, se agrupan bajo el nombre de *audio hash*.

Por ejemplo, en [23] se comparan los algoritmos de Coskun y de Nilsima, dos técnicas de *hashing* localmente sensible (puede consultarse una exhaustiva revisión del tema en [24]), para detectar y eliminar tempranamente mensajes de voz no solicitados, tanto comerciales como spam, que hayan quedado registrados en contestadores automáticos. Incluso al tratarse de audio pregrabado y de llamadas VoIP, el audio final no resulta indistinguible entre observaciones ya sea por ruido en el segmento analógico, por demoras y pérdidas de paquetes en el segmento digital, por la utilización de diferentes calidad de almacenamiento en el receptor, o porque el emisor opta por generar los mensajes dinámicamente para dificultar la detección. Aún así, el esquema presentado alcanza una aceptable tasa de error de menos del 5 % si no se modifica notoriamente el diálogo o la vocalización. Cabe aclarar que los esquemas utilizados enfatizan la simplicidad y la velocidad de procesamiento, por lo que pueden lograrse tasas de error más bajas a costa de incrementar la potencia de procesamiento disponible. En el año de publicación (2014), el tiempo de procesamiento para un mensaje de treinta segundos se encontraba en el orden de medio segundo en hardware de propósito general. El algoritmo es paralelizable.

Es posible extender las técnicas anteriores para detectar similaridades en el audio con mucha mayor tolerancia a artefactos introducidos por el procesamiento, compresión, o almacenamiento, de forma tal que las misma sean útiles para la recuperación por contenido. Intuitivamente, entendemos que dos grabaciones de audio pueden referir al mismo contenido aunque la forma de onda difiera notoriamente. Así, reconocemos la misma pieza musical por la partitura aunque los ejecutantes difieran. Idénticamente, el mismo discurso puede ser pronunciado por dos hablantes distintos. En [25] se utilizan los picos locales de potencia en la señal de audio, conjuntamente con el análisis espectral de sus contornos, para formar secuencias características pasibles de ser indexadas. Una vez más las técnicas de hashing localmente sensible son aplicables a este caso, luego del preprocesamiento indicado. La precisión lograda alcanza el 80 %, aún en los casos de mayor dificultad como piezas musicales con distintos intérpretes y ejecutadas a distinta velocidad.

Las técnicas anteriores son efectivas para la detección de similaridades globales, pero si el objetivo es descubrir repeticiones parciales como en el caso de objetos de texto y no contenido repetido con variaciones, se debe recurrir a otro tipo de funciones de hashing denominadas *de trazo grueso* (del inglés *coarse grained fingerprinting*) [26]. Este tipo de esquemas han sido utilizados para detectar usos no autorizados y violaciones a los derechos de autor [27], pues permiten detectar la utilización de fragmentos de audio embebidos en otros de contenido como video o dentro de una mezcla de sonido. Idénticamente, la generación de huellas auditivas (del inglés *audio fingerprinting*) permite la recuperación de objetos de audio cuando sólo un fragmento es conocido, e incluso si el mismo ha sido sujeto a distorsión intencional o inintencional, con tasas de error despreciables siempre que la calidad de grabación supere un umbral aceptable [27]. Las aplicaciones de reconocimiento de canciones, como Shazam o las incluídas en Apple Siri, Google Now, o Microsoft Cortana, utilizan principios similares.

IV-B. Imagen y video

En forma similar al caso de objetos de audio, la detección de imagénes casi idénticas utiliza métricas de similaridad insensibles al ruido, al cambio de calidad, o la compresión/descompresión de la información visual, a la vez que modificaciones menores en el contenido producen valores de salida insignificantes. Sí debe destacarse que, debido al mayor volumen de información contenido en imágenes y videos en contraste con el audio y más aún con el texto, las consideraciones de escalabilidad deben ser priorizadas cuando las mismos se intentan aplicar en repositorios de gran tamaño.

Es interesante destacar el método min-Hash pues este permite graduar la sensibilidad por medio de un parámetro, haciendo oscilar el umbral de detección desde imágenes perfectamente idénticas hasta todo tipo de variaciones que, a pesar de diferir en términos de información cruda, son muy similares para un observador humanos *chum2008near*. Si bien este tipo de técnicas no han sido aplicadas a la deduplicación de objetos en repositorios digitales, sí han encontrado utilización en la detección de usos no autorizados de imágenes con derechos reservados, o la eliminación de marcas de agua en contenido públicamente accesible [**?**].

V. Un estudio empírico sobre el repositorio del SEDICI

TODO: GERMÁN, COMPLETÁ

VI. Conclusiones

TODO: GERMÁN, COMPLETÁ

Referencias

[1] G. Forman, K. Eshghi, and S. Chiocchetti, "Finding similar files in large document repositories," in *Proceedings of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining*, 2005, pp. 394–400.

[2] K. Monostori, A. Zaslavsky, and H. Schmidt, "Document overlap detection system for distributed digital libraries," in *Proceedings of the fifth ACM conference on Digital libraries*, 2000, pp. 226–227.

[3] H. A. Chowdhury and D. K. Bhattacharyya, "Plagiarism: Taxonomy, tools and detection techniques," *arXiv preprint arXiv:1801.06323*, 2018.

[4] G. López, A. C. Servetto, A. Echeverría, I. Jeder, M. D. Grossi, and E. M. Jiménez Rey, "Ontologías en arquitecturas dirigidas por modelos," in *XIII Workshop de Investigadores en Ciencias de la Computación*, 2011.

[5] D. M. Sánchez, J. M. Cavero, and E. Marcos, "Ontologías y mda: una revisión de la literatura," *Actas del II Taller sobre Desarrollo de Software Dirigido por Modelos, MDA y Aplicaciones (DSDM 2005)*, p. 21, 2005.

[6] G. Nie, Z.-C. Fu, D. Chen, and P. Liu, "Ontology-based thesis copy detection system [j]," *Computer Engineering*, vol. 19, no. 31, pp. 79–81, 2005.

DRAFT

[7] N. Do and L. Ho, "Domain-specific keyphrase extraction and near-duplicate article detection based on ontology," in *The 2015 IEEE RIVF International Conference on Computing & Communication Technologies-Research, Innovation, and Vision for Future (RIVF)*. IEEE, 2015, pp. 123–126.

[8] J. Deepika, V. Archana, V. Bagyalakshmi, P. Preethi, and G. Mahalakshmi, "A knowledge based approach to detection of idea plagiarism in online research publications." *International Journal on Internet & Distributed Computing Systems*, vol. 1, no. 2, 2011.

[9] T. A. E. Eisa, N. Salim, and S. Alzahrani, "Existing plagiarism detection techniques," *Online Information Review*, 2015.

[10] I. M. I. Subroto and A. Selamat, "Plagiarism detection through internet using hybrid artificial neural network and support vectors machine," *Telkomnika*, vol. 12, no. 1, p. 209, 2014.

[11] I. Smeureanu and B. Iancu, "Source code plagiarism detection method using protégé built ontologies," *Informatica Economica*, vol. 17, no. 3, p. 75, 2013.

[12] S. Bechhofer, F. Van Harmelen, J. Hendler, I. Horrocks, D. L. McGuinness, P. F. Patel-Schneider, L. A. Stein *et al.*, "Owl web ontology language reference," *W3C recommendation*, vol. 10, no. 02, 2004.

[13] H. Van de Sompel, R. Chute, and P. Hochstenbach, "The adore federation architecture: digital repositories at scale," *International Journal on Digital Libraries*, vol. 9, no. 2, pp. 83–100, 2008.

[14] R. Tansley, "Building a distributed, standards-based repository federation," *D-Lib Magazine*, vol. 12, no. 7, p. 1, 2006.

[15] T. C. Hoad and J. Zobel, "Methods for identifying versioned and plagiarized documents," *Journal of the American society for information science and technology*, vol. 54, no. 3, pp. 203–215, 2003.

[16] M. W. Berry and M. Browne, *Understanding search engines: mathematical modeling and text retrieval*. SIAM, 2005.

[17] M. Chawla, "An indexing technique for efficiently detecting plagiarism in large volumes of source code," *Unpublished Honours thesis, RMIT University, Melbourne, Australia*, 2003.

[18] N. Heintze *et al.*, "Scalable document fingerprinting," in *1996 USENIX workshop on electronic commerce*, vol. 3, no. 1. Citeseer, 1996.

[19] A. Z. Broder, "On the resemblance and containment of documents," in *Proceedings. Compression and Complexity of SEQUENCES 1997 (Cat. No. 97TB100171)*. IEEE, 1997, pp. 21–29.

[20] M. O. Rabin, "Fingerprinting by random polynomials," *Technical report*, 1981.

[21] P. A. Hall and G. R. Dowling, "Approximate string matching," *ACM computing surveys (CSUR)*, vol. 12, no. 4, pp. 381–402, 1980.

[22] S. Burrows, S. M. Tahaghoghi, and J. Zobel, "Efficient plagiarism detection for large code repositories," *Software: Practice and Experience*, vol. 37, no. 2, pp. 151–175, 2007.

[23] G. Zhang and S. Fischer-Hübner, "Detecting near-duplicate spits in voice mailboxes using hashes," in *International Conference on Information Security*. Springer, 2011, pp. 152–167.

[24] J. Wang, H. T. Shen, J. Song, and J. Ji, "Hashing for similarity search: A survey," *arXiv preprint arXiv:1408.2927*, 2014.

[25] C. Yang, "Macs: music audio characteristic sequence indexing for similarity retrieval," in *Proceedings of the 2001 IEEE Workshop on the Applications of Signal Processing to Audio and Acoustics (Cat. No. 01TH8575)*. IEEE, 2001, pp. 123–126.

[26] A. Saracoglu, E. Esen, T. K. Ates, B. O. Acar, U. Zubari, E. C. Ozan, E. Ozalp, A. A. Alatan, and T. Ciloglu, "Content based copy detection with coarse audio-visual fingerprints," in *2009 Seventh International Workshop on Content-Based Multimedia Indexing*. IEEE, 2009, pp. 213–218.

[27] C. Ouali, P. Dumouchel, and V. Gupta, "A robust audio fingerprinting method for content-based copy detection," in *2014 12th International Workshop on Content-Based Multimedia Indexing (CBMI)*. IEEE, 2014, pp. 1–6.

[28] Z. Zhou, Y. Wang, Q. J. Wu, C.-N. Yang, and X. Sun, "Effective and efficient global context verification for image copy detection," *IEEE Transactions on Information Forensics and Security*, vol. 12, no. 1, pp. 48–63, 2016.

DRAFT