Universidad de Cienfuegos "Carlos Rafael Rodríguez"

Facultad de Ingeniería
Carrera de Ingeniería Informática

Trabajo de diploma para optar por el título de Ingeniería en Informática

Título:

"Desarrollo de un nuevo método de Aprendizaje de la Ordenación, basado en el Procedimiento de Búsqueda el Pescador."

Autora:
Yaimi García Álvarez

Tutor:
Msc. Oscar José Alejo Machado

Cienfuegos, Cuba
Curso 2012-2013

Declaración de autoría

Declaro que soy el único autor de este trabajo y autorizo al Departamento de Informática de la Facultad de Ingeniería en la Universidad de Cienfuegos “Carlos Rafael Rodríguez”, para que hagan el uso que estimen pertinente con el trabajo de diploma.

Para que así conste firmo la presente a los ____ días del mes de _____ del 2013.

Yaimi García Álvarez

Nombre completo del autor

Msc. Oscar José Alejo Machado

Nombre completo del tutor

Los abajo firmantes certificamos que el presente trabajo ha sido revisado según acuerdo de la dirección de nuestro centro y el mismo cumple los requisitos que debe tener un trabajo de esta envergadura referente a la temática señalada.

Firma Tutor

Firma ICT

Firma Vicedecano

Pensamiento

"El conocimiento no es algo separado y que se baste a sí mismo, sino que está envuelto en el proceso por el cual la vida se sostiene y se desenvuelve"

John Dewey.

Agradecimientos

A mi tutor Oscarito que me motivó y guió, con su paciencia e incondicional ayuda.

A mi hermanito Yandy que estuvo ahí siempre que lo necesité.

A mi hermano Nani que me brindó su apoyo en todo momento.

A mi familia por su apoyo y su confianza.

A mi amiga Sadys por su amistad, apoyarme y soportarme todos estos años.

A todos los profesores de la carrera por las enseñanzas dadas.

A todas las personas que de una forma u otra han contribuido a mi formación y han hecho realidad este sueño.

Dedicatoria

A Mis Padres que son mi Tesoro y este es su mayor regalo.

Resumen

Uno de los problemas claves dentro del Aprendizaje de la Ordenación ("*Learning to Rank*") para la Recuperación de Información (R.I.) es desarrollar algoritmos que construyan modelos de ordenación optimizando directamente medidas de evaluación comúnmente usadas en R.I. En esta investigación, se propone un nuevo método de Aprendizaje de la Ordenación, denominado RankFSP, y basado en el Procedimiento de Búsqueda el Pescador (FSP, *siglas en inglés*).

El algoritmo propuesto es capaz de construir una función de ordenación optimizando directamente cualquier medida de evaluación según un determinado conjunto de entrenamiento. RankFSP implementa además una estrategia para evadir los mínimos locales basada en el reinicio de los puntos de pesca no mejorados.

Se desarrollaron experimentos, donde el método propuesto mostró una estabilidad reflejada en los tres primeros lugares de la ordenación, al ser evaluado su desempeño en los conjuntos de datos de LETOR 3.0 y 4.0. Se realizó además, un análisis estadístico donde se corroboró que ninguno de los métodos del Aprendizaje de la Ordenación mejoran de forma significativa la actuación de RankFSP; sin embargo, este supera en cuanto a precisión en la ordenación a los métodos FRank, ListNet, RankBoost y a RankSVM en OHSUMED, y a RankBoost y Regression en NP2004.

Finalmente, los resultados de la investigación muestran las ventajas de utilizar el Procedimiento de Búsqueda el Pescador para tratar el problema del Aprendizaje de la Ordenación.

Índice de Contenido

Índice de Figuras

Índice de Tablas

Introducción

La aparición de la Internet ha producido cambios profundos en la actividad humana, a tal punto que en la actualidad se denomina la era de la información. Las grandes cantidades de información, que se crean y se manejan actualmente, han hecho que su almacenamiento y recuperación sea un problema que está en constante investigación, bajo el nombre de Recuperación de Información (R.I.).

En este contexto, las aplicaciones de la R.I. enfrentan diversos problemas, la ordenación constituye uno de los más significativos a la hora de localizar información pertinente ante el exceso de información existente.

En la recuperación de documentos, este problema consiste en definir un orden representativo entre ellos, teniendo en cuenta su relevancia con respecto a la consulta de un usuario, de forma tal que la información relevante sea colocada en mejor posición respecto a aquella que es menos relevante o irrelevante.

Ante esta problemática, un grupo de investigadores comenzaron a desarrollar diversos modelos de R.I. entre los años 1970s y 1990s. Estos modelos se agruparon en dos grupos atendiendo a su concepción y a la relación que tenían con las consultas de los usuarios: modelos dependientes de la consulta y modelos independientes de la consulta.

Dentro de los modelos dependientes de la consulta están: los modelos booleanos, los vectoriales, los probabilísticos, los basados en el lenguaje, entre otros; y como modelos independientes de la consulta se pueden mencionar: *PageRank* y *BrowseRank*.

En la mayoría de estos modelos de R.I. un documento es representado como un conjunto de palabras claves representativas (términos que instancian rasgos o características de los documentos) por las cuales es indexado, y se define una función de ordenación (o función de recuperación) que permite asociar un grado de relevancia con un documento y una consulta.

Con el paso del tiempo, los modelos de ordenación se complejizan: se consideran nuevos rasgos para su formulación, se hace necesario hacerlos

variables y que además aprendiesen a partir de datos de entrenamiento. Todo esto unido a la incorporación de juicios de relevancia a las colecciones documentales en 1994, hace que se comience a pensar en emplear técnicas de aprendizaje automático.

Los métodos que se comenzaron a desarrollar desde entonces, utilizan técnicas de aprendizaje automático y dieron origen a un nuevo tópico vigente de investigación dentro de la rama de la Inteligencia Artificial y la Recuperación de Información: el Aprendizaje de la Ordenación, denominado en inglés *Learning to Rank*.

Estos nuevos métodos de Aprendizaje de la Ordenación se basan en el aprendizaje supervisado y persiguen sobre cualquier escenario, crear de forma automática, un modelo de ordenación usando datos de entrenamiento, técnicas de aprendizaje automático y medidas de desempeño de la R.I. como: MAP (*Mean AveragePrecision*), NDCG (*NormalizedDiscountedCumulativeGain*) y P@n (*Precision at n*).

El Aprendizaje de la Ordenación se enmarca inicialmente en tres categorías principales: Aproximación por Puntos (*PointwiseApproach)*, que trabaja la ordenación sobre documentos simples; Aproximación por Pares (*PairwiseApproach)*, en la cual se transforma o formaliza el problema en una clasificación binaria sobre pares de documentos y Aproximación por Listas (*ListwiseApproach)*, en la cual se minimiza directamente una función de pérdida definida sobre una lista de documentos.

La mayoría de los métodos referenciados dentro de estas categorías son capaces solamente de entrenar modelos de ordenación minimizando una función de pérdida relativa en cierto grado a las medidas de desempeño. Sin embargo, idealmente un algoritmo de aprendizaje debiese entrenar un modelo de ordenación que directamente optimizara medidas de desempeño con respecto a los datos de entrenamiento.

A partir del año 2005 se trabaja mayormente en la categoría *ListwiseApproach* específicamente en la optimización directa de medidas de desempeño y se identifican tres vertientes fundamentales: (1) minimizar una función de pérdida

que acote superiormente una función de pérdida básica definida sobre las medidas de R.I.; (2) aproximar las medidas de R.I. mediante funciones fáciles de manipular y (3) usar tecnologías especialmente diseñadas para optimizar medidas de R.I. no suaves. En esta última se enmarca el método que se propone en el presente trabajo.

Aunque los métodos que optimizan directamente medidas de desempeño, han propiciado un giro contundente en el estado del arte del Aprendizaje de la Ordenación, todavía las soluciones muestran dificultad para alcanzar una aproximación global debido a que en diversas ocasiones la función objetivo es discontinua, estacionaria y presenta problemas de convexidad, por lo que el gradiente se indefine y los métodos sólo garantizan una aproximación local. A esto se une el problema ya conocido de la variabilidad de la estabilidad, en cuanto a precisión en la ordenación, de los métodos referenciados para las diferentes colecciones de datos existentes.

Es el análisis de estos antecedentes lo que lleva a plantear el siguiente **problema científico**:

¿Cómo optimizar directamente cualquier medida de desempeño utilizada en la R.I., logrando una estabilidad de precisión aceptable frente a los métodos de Aprendizaje de la Ordenación referenciados en la literatura?

Se define como **objeto de estudio**: el Aprendizaje de la Ordenación para la Recuperación de Información y como **campo de acción**: los métodos de Aprendizaje de la Ordenación que optimizan directamente medidas de desempeño.

Encontrar una estrategia capaz de optimizar la solución, es una tarea compleja que sugiere el uso de técnicas computacionales basadas en *SoftComputing* (Algoritmos Evolutivos, Programación Genética, Lógica difusa, entre otros).

FSP (Fisherman Search Procedure) es un algoritmo de búsqueda que explora nuevas soluciones combinando la búsqueda guiada y la búsqueda local. Esta metaheurística fue diseñada con el propósito de desarrollar soluciones útiles y prácticas para una variedad de problemas de optimización, por lo que se considera que puede brindarle solución a la problemática planteada y se define

como **Idea a defender:** la concepción de un nuevo método de Aprendizaje de la Ordenación inspirado en FSP, optimizará directamente cualquier medida de desempeño utilizada en la R.I. y alcanzará una estabilidad de precisión aceptable frente a los métodos referenciados en la literatura.

Partiendo de esta idea, la presente investigación se traza el siguiente **objetivo general**:

- Proponer un nuevo método de Aprendizaje de la Ordenación inspirado en FSP, que se centre en la optimización directa de cualquier medida de desempeño utilizada en la R.I. y en una estabilidad de precisión aceptable frente a los métodos referenciados en la literatura.

De este se derivan los siguientes **objetivos específicos:**

- Analizar el estado del arte del Aprendizaje de la Ordenación y la metaheurística FSP.
- Diseñar un nuevo método de Aprendizaje de la Ordenación tomando como base el algoritmo FSP.
- Implementar el método propuesto.
- Realizar pruebas y comparaciones con colecciones de datos y métodos referenciados en la literatura.

Las **tareas** a realizar para cumplir con los objetivos propuestos son:

- Estudio de las principales categorías, modelos y artículos científicos publicados en el área de investigación del Aprendizaje de la Ordenación para la R.I.
- Implementación del algoritmo FSP adaptado al problema del Aprendizaje de la Ordenación.
- Determinación del rendimiento que se puede alcanzar con el modelo propuesto mediante su evaluación con una serie de colecciones documentales estándar.
- Comparación, utilizando métodos estadísticos, de los desempeños del modelo propuesto con otros modelos clásicos de Aprendizaje de la Ordenación, así como con los que optimizan directamente medidas de desempeño utilizadas en R.I.

Para el desarrollo de la investigación se emplea una metodología mixta ya que se utilizan elementos de la metodología cuantitativa y cualitativa. El enfoque cualitativo permite la interpretación del fenómeno y el enfoque cuantitativo, su explicación. Esta es una tendencia que se está utilizando cada vez con mayor frecuencia en las investigaciones.

La investigación cualitativa propone la estancia prolongada en el campo y la observación persistente de los focos principales de la investigación. Se recogen las informaciones y luego se analizan desde distintos ángulos a fin de contrastarlos, realizando el cruzamiento e interpretando y hallando las coincidencias de los resultados alcanzados.

Se consideran de utilidad los métodos cuantitativos, ya que los números constituyen instrumentos de modelación de la realidad, de aproximación al conocimiento y contribuyen a ordenar la información en el proceso de producción del conocimiento.

La metodología mixta posibilita aprovechar las tendencias, la direccionalidad que pueden mostrar algunos métodos cuantitativos sobre la base del análisis cualitativo, para cruzar y verificar la información obtenida aplicando diferentes métodos. Por eso se considera que la aproximación a la verdad está vinculada con la pluralidad metodológica, ya que el método debe poseer la capacidad de reflejar el movimiento, la dinámica, la esencia del objeto y esto no se logra sin la participación compensatoria de los diversos métodos existentes. [38]

Entre los métodos cualitativos y cuantitativos empleados en este trabajo se encuentran la modelación algorítmica, la experimentación y el análisis estadístico.

Los métodos científicos, según su etimología, son el camino hacia el conocimiento. En la presente investigación se utilizan varios de estos métodos:

- Nivel teórico.
  - Inducción – deducción: La inducción expresa el movimiento de lo particular a lo general, o sea se llega a generalizaciones partiendo del análisis de casos particulares, mientras la deducción expresa el movimiento de lo general a lo particular. La combinación de ambos

permite estructurar el conocimiento científico obtenido en la revisión bibliográfica.

- Histórico – lógico: El método lógico debe basarse en los datos que proporciona el método histórico, de manera que no se convierta en un simple razonamiento especulativo. De igual forma, lo histórico no debe limitarse a la simple descripción de los hechos, sino explicarlos a partir de la lógica de su desarrollo [39]. Este método da la posibilidad de conocer el problema de la ordenación en la Recuperación de Información en su origen y desarrollo, y proyectar el análisis de la evolución histórica de los métodos de Aprendizaje de la Ordenación con la lógica de su comportamiento futuro.
- Análisis y síntesis: A través del análisis se distinguen y revisan por separado los elementos relacionados con el Aprendizaje de la Ordenación para la R.I. Partiendo de esto, se emplea la síntesis para establecer nexos, comparar resultados, determinar características comunes y aspectos distintivos de los diferentes enfoques estudiados, lo que permite arribar a conclusiones.
- Comparación: Para establecer las similitudes y diferencias entre los modelos y categorías existentes dentro del Aprendizaje de la Ordenación para la R.I.

✍ Nivel matemático y estadístico.

- Métodos de la Estadística: Con la utilización de la estadística descriptiva (distribución de frecuencias, medidas de tendencia central y medidas de la variabilidad) y las pruebas no paramétricas para muestras relacionadas o pareadas, es posible evaluar el desempeño del método RankFSP y comparar sus resultados con los métodos referenciados en la literatura.

Se considera el siguiente **aporte práctico**: el algoritmo de aprendizaje concebido RankFSP permitirá a motores de búsqueda y a sistemas de recuperación de información, mejorar la pertinencia y la relevancia en el proceso de ordenación de los documentos recuperados.

El desarrollo de este método brinda la siguiente **novedad científica:** por vez primera se concibe un método de Aprendizaje de la Ordenación para la R.I.,

utilizando la metaheurística FSP.

Para el adecuado análisis y entendimiento de este documento, se estructurará el mismo en 3 capítulos.

Capítulo I.- Fundamentación Teórica. En este se define la R.I., sus objetivos y los retos que enfrentan los Sistemas de R.I. Se describe la evolución que han tenido los modelos de R.I. dando lugar al Aprendizaje de la Ordenación con nuevos métodos que emplean técnicas de Inteligencia Artificial y se enmarcan en tres categorías fundamentales: aproximación por puntos, por pares y por listas. Se presenta una descripción de la metodología empleada para el desarrollo de la investigación, así como las herramientas utilizadas. Al final se plantean algunas discusiones y trabajos futuros relacionados con las temáticas abordadas en la investigación.

Capítulo II.- FSP en el Aprendizaje de la Ordenación. En un primer momento, en este capítulo se describen los elementos básicos del Procedimiento de Búsqueda el Pescador, así como las posibles variantes sugeridas por los autores. Luego, se formaliza el problema de Aprendizaje de la Ordenación y posteriormente se detalla la propuesta algorítmica RankFSP como un nuevo método de Aprendizaje de la Ordenación basado en FSP. En este proceso de adaptación de FSP al Aprendizaje de la Ordenación se incluyen nuevos aportes encaminados a un mejor desempeño del método para el problema tratado.

Capítulo III.- Resultados Experimentales. En esta estructura capitular se desglosan y explican los conjuntos de datos, algoritmos y medidas de desempeño utilizadas para la evaluación del método propuesto. Se muestran los resultados obtenidos de la evaluación de RankFSP en cada uno de los conjuntos de datos y considerando las diversas medidas. Finalmente, se realiza un análisis estadístico, apoyado en pruebas no paramétricas, para corroborar la efectividad de la propuesta frente a los principales métodos referenciados en la literatura.

Capítulo I: Fundamentación teórica

1.1 Introducción

El presente capítulo aborda de forma panorámica y conceptual, las principales investigaciones que diversos autores han venido realizando dirigidas a la Recuperación de Información (R.I.) y al Aprendizaje de la Ordenación para la R.I.

Se plantean las principales características y los objetivos que persigue el Aprendizaje de la Ordenación como problema clave en la recuperación de documentos y se describen las categorías principales en las que se enmarcan los modelos que se desarrollan.

Se presenta un estudio métrico que se desarrolla con el objetivo de evaluar la producción científica en el Aprendizaje de la Ordenación.

Se explica brevemente la metaheurística FSP y se presenta la metodología empleada en el desarrollo de la investigación, así como las herramientas utilizadas. Finalmente se mencionan algunas discusiones y trabajos futuros relacionados con las temáticas abordadas en la investigación.

1.2 Recuperación de Información

La R.I. es un campo relacionado con la estructura, análisis, organización, almacenamiento, búsqueda y obtención de información [1] que se ha ido desarrollando desde la década de los sesenta del pasado siglo. Con el paso del tiempo ha sido influenciada por la popularización de Internet y los continuos y crecientes avances producidos en las tecnologías de la información que cada día almacenan y procesan mayor cantidad de información.

Su principal objetivo es proporcionarle información relevante al usuario para satisfacer su necesidad de información. El cómo lograrlo ha traído el interés de investigadores de diversas disciplinas y provocado un trabajo multi e interdisciplinar [2]. Fruto de este trabajo surgen los Sistemas de Recuperación de Información (S.R.I.) que permiten identificar aquellos documentos

pertenecientes a una colección que mejor responden a las necesidades de un usuario, expresadas mediante una consulta.
La representación y organización que realizan estos sistemas deberían proveer al usuario de un fácil acceso a la información en la que se encuentre interesado. Desafortunadamente, la caracterización de la necesidad informativa de un usuario no es un problema sencillo de resolver [3].

Los S.R.I. convencionales representan un documento como un conjunto de palabras claves representativas (términos que instancian rasgos o características de los documentos, por los cuales son indexados), y se define una función de ordenación que permite asociar un grado de relevancia al documento respecto a una consulta determinada. Esta información es utilizada para obtener un orden de los documentos colocando en mejores posiciones aquellos que contienen información más relevante.

En estos sistemas, tanto los términos de entrada como el texto de salida se pueden ordenar según ciertos criterios; el reto para los investigadores es desarrollar algoritmos que optimicen estos rasgos [4].

Para abordar este problema se han desarrollado un gran número de modelos que relacionan documentos y consultas a través de funciones de ordenación.

1.2.1 Modelos de R.I. convencionales

En el período de 1970 a 1990 surgieron varios modelos fundamentados en métodos formales. Estos se enmarcan en dos categorías:

- Modelos dependientes de la consulta:
  - Modelo Booleano (*Boolean Model*), Modelo Booleano Extendido [5]
  - Modelo del Espacio Vectorial (*Vector Space Model*) [6]
  - Modelos Probabilísticos (*Probabilistic Models*) [7]
  - Modelo BM25 [8]
  - Modelo del lenguaje estadístico (*Statistical language model*) [9]
  - *Latent Semantic Indexing* (LSI) [10]
  - Modelos del lenguaje para la RI (LMIR, *Language Models for Information Retrieval*) [11]
- Modelos independientes de la consulta

- *PageRank* [12]
- *TrustRank* [13]
- *BrowseRank* [14]

El *modelo booleano* es el más usado históricamente. Está basado en la teoría de conjuntos y el álgebra booleana. Su efectividad radica en dividir los términos de la búsqueda en conjuntos, por ello es muy fácil de implementar y entender [15]. Su problema es que devuelve resultados de todo o nada y la Recuperación de Información no es un proceso exacto.

En el *modelo de espacio vectorial*, los documentos y las búsquedas se interpretan como vectores de términos, y se representa cada término en el vector con un peso *w* dentro de ese documento [15]. La función de similitud entre el documento y la consulta será el coseno del ángulo entre los vectores que los representan. La funcionalidad de este modelo radica en la elección correcta de los pesos de cada término.

El *modelo probabilístico* utiliza la teoría de probabilidades para modelar la incertidumbre del proceso de Recuperación de Información. En este, los documentos se ordenan de forma decreciente según su probabilidad de relevancia respecto a la información requerida. Sin embargo, no se usan frecuencias de términos dentro del documento ni longitud de documentos.

Estos modelos resultaron significativos en este período de grandes retos científicos y actualmente influyen de forma directa o indirecta en la mayoría de las investigaciones que se realizan en el campo de la R.I. [16]

El hecho de que muchos de los procesos que lleva a cabo la R.I. tengan un marcado carácter impreciso e incierto (caracterización del contenido de un documento mediante vectores de términos, descripción de la necesidad de información de un usuario mediante una consulta, ...) condujo a que con el paso del tiempo, los modelos de ordenación se fueran haciendo más complejos: se consideraron nuevos rasgos para su formulación, se hizo necesario hacerlos variables y que además aprendiesen a partir de datos de entrenamiento. Todo esto unido a la incorporación de juicios de relevancia a las

colecciones documentales en 1994, hizo que se comenzara a pensar en emplear técnicas de aprendizaje automático.

Los métodos que se comenzaron a desarrollar desde entonces, utilizan técnicas de aprendizaje automático y dieron origen a un nuevo tópico vigente de investigación dentro de la rama de la Inteligencia Artificial y la Recuperación de Información: el Aprendizaje de la Ordenación, denominado en inglés *Learning to Rank*.

1.3 Aprendizaje de la ordenación

1.3.1 Descripción del problema

En el proceso de aprendizaje, el sistema recibe un conjunto de consultas y los documentos recuperados correspondientes, además de las etiquetas de los documentos con respecto a las consultas. Estas etiquetas representan niveles o categorías de relevancia en un orden total.

El propósito del aprendizaje es construir un modelo de ordenación capaz de obtener los mejores resultados sobre un conjunto de datos, haciendo corresponder el conjunto ordenado según el modelo con el conjunto de documentos idealmente ordenados de acuerdo a los juicios o categorías de relevancia, que representan las preferencias de determinado usuario.

La correspondencia entre ambos conjuntos es determinada a través de una medida de desempeño, que cuantifica la relevancia del conjunto de documentos recuperados según las necesidades manifestadas por el usuario en su consulta. Entre las medidas de desempeño más utilizadas en R.I. están: MAP (*Mean Average Precision*) [17], NDCG (*Normalized Discounted Cumulative Gain*) [18] y P@n (*Precision at n*) [17]. Para una mejor comprensión de las medidas de desempeño, consulte el apartado 3.3 Medidas de Evaluación.

El objetivo del algoritmo de aprendizaje es entrenar un modelo de ordenación que directa o indirectamente optimice una de estas medidas de desempeño con respecto a los datos del conjunto de entrenamiento definido anteriormente. Específicamente, una función de ordenación de la forma $f: Q \times D \rightarrow \mathbb{R}$, asigna

puntuaciones reales de adecuación a los documentos recuperados del conjunto D con respecto a la correspondiente consulta perteneciente al conjunto Q y a continuación ordena tales documentos a partir de sus puntuaciones. [16]

El desempeño de esta función de ordenación se evalúa teniendo en cuenta el orden resultante y el orden ideal. Finalmente, el modelo realiza un ajuste general de sus parámetros, según el comportamiento evaluado. Este proceso de aprendizaje se repite hasta abarcar todos los datos de entrenamiento.

Cuando se ha ajustado el modelo de ordenación, se pasa a la etapa de prueba, en la que dada una consulta y la lista de documentos recuperados en relación a esta, el sistema debe ser capaz de retornar la lista de documentos ordenados de acuerdo a su relevancia.

Idealmente, la función de ordenación en esta etapa, alcanzará resultados suficientemente buenos en precisión que propicien la satisfacción del usuario.

Este proceso de aprendizaje puede ser comprendido mejor si se analiza el esquema de la Figura 1.1.

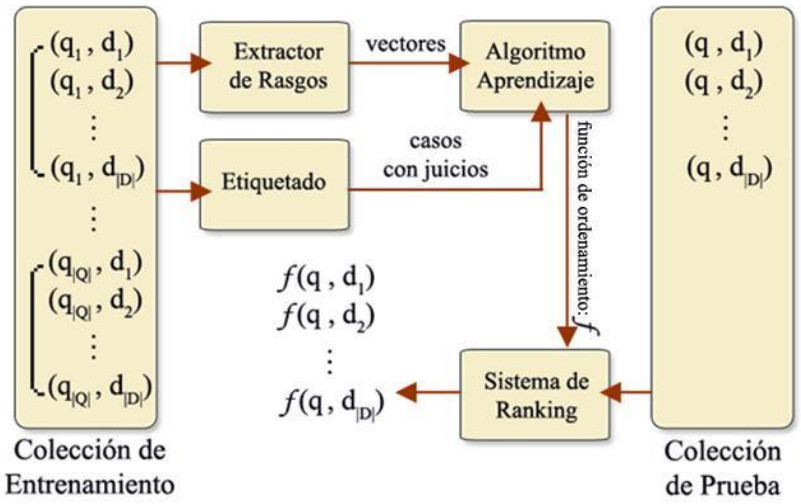


Figura 1.1. Esquema General del Aprendizaje de la Ordenación para la R.I. [16]

1.3.2 Principales categorías y modelos

Los métodos existentes de Aprendizaje de la Ordenación se dividen en tres categorías: (1) Aproximación por puntos *(Pointwise Approach)*, que realiza la ordenación sobre un par consulta - documento; (2) Aproximación por pares *(Pairwise Approach)*, en la que de manera

general se transforma o formaliza el problema de la ordenación en una clasificación binaria sobre pares de documentos y (3) Aproximación por listas *(Listwise Approach)*, en la cual se toma la lista de documentos recuperados, con respecto a una consulta, como instancias para entrenar el modelo de ordenación.

1.3.2.1 Aproximación por puntos (Pointwise approach)

En la aproximación por puntos el espacio de entrada contiene los vectores de atributos de cada par consulta - documento. El espacio de salida representa el grado de relevancia de cada documento. En el espacio de hipótesis se consideran funciones que toman los vectores de atributos y predicen un valor de relevancia del documento respecto a la consulta. La función de pérdida se obtiene a partir de problemas de regresión, clasificación o regresión ordinal. [19]

Una gran ventaja de estas aproximaciones es su sencillez, pues aplican técnicas previamente desarrolladas cuyo funcionamiento está respaldado por la comunidad de aprendizaje automático. No obstante, la sencillez constituye también su principal desventaja.

Esto es porque, conceptualmente, las tareas de regresión, clasificación y ordenación son muy distintas y se requiere de métodos más específicos que consideren relaciones de orden, pues en esto consiste el problema original. [19]

1.3.2.2 Aproximación por pares (Pairwise approach)

Una evolución de los métodos anteriores consiste en escoger pares de documentos cuya relevancia relativa con respecto a una consulta sea conocida.

En la aproximación por pares el espacio de entrada contiene pares de documentos representados por sus vectores de atributos. El espacio de salida representa los órdenes relativos de los pares de documentos con respecto a la consulta en el *ranking* esperado. Por tanto, se espera que las funciones del espacio de hipótesis sean capaces de ordenar correctamente pares de documentos respecto a su relevancia frente a

una consulta. Las funciones de pérdida, por lo general, tienden a cuantificar el número de pares mal ordenados. [19]
En el proceso de aprendizaje se toman pares de documentos de la lista recuperada y para cada par se asigna una etiqueta representando la relevancia relativa de los dos documentos [20]. Esta etiqueta puede tomar dos valores: correctamente ordenado e incorrectamente ordenado. De esta forma el problema de la ordenación queda formalizado como un problema de clasificación.

La gran ventaja de usar estos modelos es que, con frecuencia, la fuente de información única o más fiable es respecto a pares de documentos y predecir el orden relativo es más cercano a la naturaleza de la ordenación que predecir la etiqueta de la clase o el grado de relevancia. Además existen metodologías de clasificación que pueden ser directamente aplicadas [20].

Una de sus desventajas reside en el tipo de funciones de error que emplea. Estas suelen estar desconectadas de las medidas de desempeño de la R.I. o en el mejor de los casos, representan cotas superiores a las mismas.

El principal problema de estas dos primeras categorías es que los modelos desarrollados realizan la ordenación con respecto a un solo documento o a un par de estos, ignorando que el aprendizaje es una tarea de predicción sobre una lista de documentos.

1.3.2.3 Aproximación por listas (Listwise Approach)

En la aproximación por listas, el espacio de entrada contiene el grupo entero de documentos asociados con una consulta. El espacio de salida consiste en permutaciones que definen *rankings* o conjuntos de puntuaciones de relevancia para cada documento. Las funciones del espacio de hipótesis tienen como objetivo reproducir permutaciones sobre los elementos o puntuaciones de relevancia. A su vez, las funciones de error pueden tener en cuenta la diferencia entre la permutación de predicción y la original, o bien usar métricas de evaluación en R.I. [19]

En esta categoría se presentan dos fuertes vertientes:

- *Listwise loss minimization* que minimiza una función de pérdida definida sobre las permutaciones, las cuales son diseñadas considerando las propiedades de la ordenación en la R.I.
- *Direct optimization of IR measure* en la que se trata de optimizar medidas de evaluación de R.I., o por lo menos, alguna función definida en correlación a tales medidas.

A partir del año 2005, los métodos aproximados que realizan la optimización directa de medidas de desempeño han despertado el interés de muchos investigadores y se han desarrollado importantes trabajos, que se agrupan en tres nuevas categorías dentro del Aprendizaje de la Ordenación:

- Minimizar una función de pérdida que acote superiormente una función de pérdida básica definida sobre las medidas de R.I.
- Aproximar las medidas de desempeño de R.I. mediante funciones fáciles de manipular.
- Utilizar tecnologías especialmente diseñadas para optimizar medidas de desempeño de R.I. no suaves.

Estos métodos son, a priori, los más completos y correctos con respecto al problema a solucionar.[19]

1.4 Métodos desarrollados

En la Figura 1.2, se muestra una línea de tiempo del Aprendizaje de la Ordenación, donde se reflejan los métodos desarrollados en cada una de las categorías generales a partir del año 2000 y hasta el 2009.

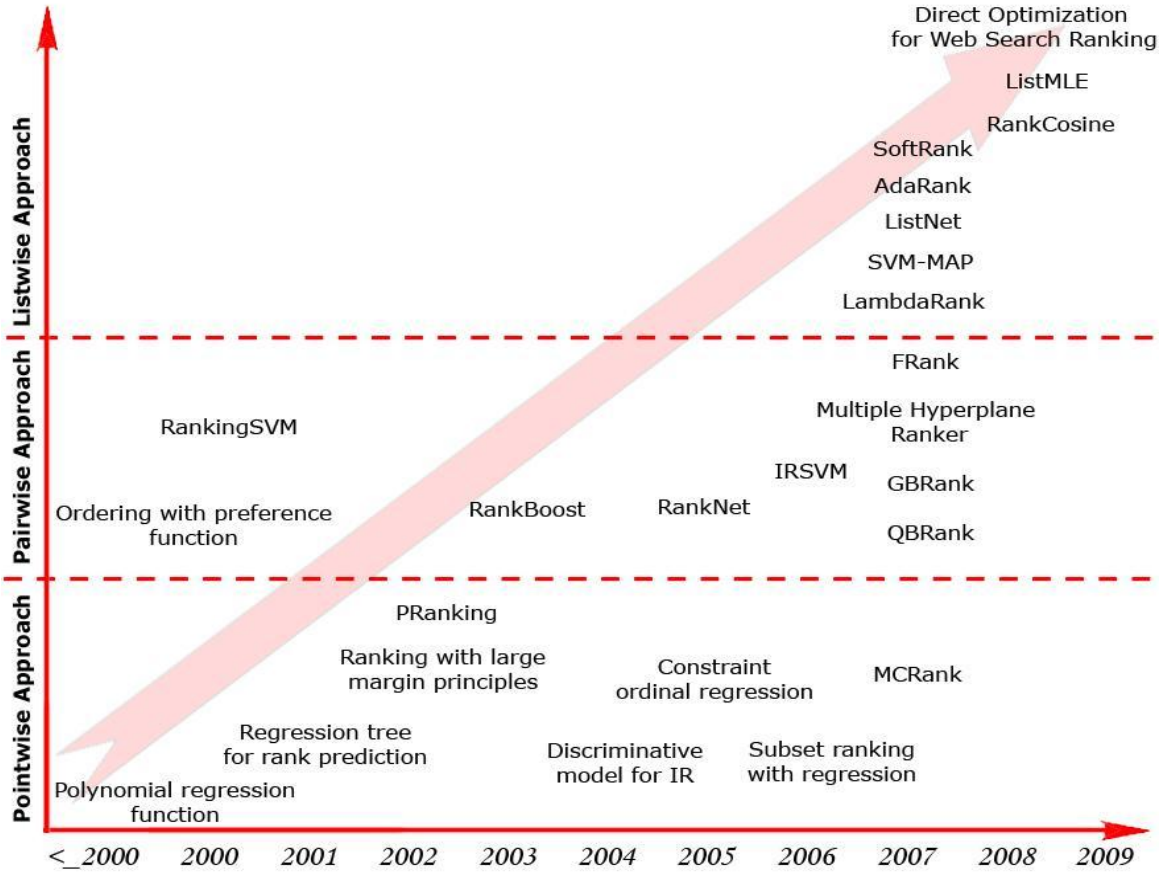


Figura 2.2. Línea de tiempo y tendencia categórica del Aprendizaje de la Ordenación. [16]

1.4.1 Problemas existentes

La mayoría de los métodos referenciados dentro de las categorías de aproximación por puntos, por pares y por listas (en la vertiente *Listwise loss minimization*) son capaces s*olamente* de entrenar modelos de ordenación minimizando una función de pérdida relativa en cierto grado a las medidas de desempeño. Ejemplo de ello, son los métodos RankingSVM [21] y RankBoost [22] que entrenan modelos de ordenación minimizando el error de clasificación sobre pares de instancias que representan documentos.

El principal problema de esta vertiente es que la conexión entre la función de pérdida y las medidas de desempeño designadas es incierta y no necesariamente la optimización de esta función resulta en la optimización de la medida de desempeño [23]. Además, un algoritmo de aprendizaje debiese entrenar un modelo de ordenación que directamente optimizara medidas de desempeño con respecto a los

datos de entrenamiento.

Con esta perspectiva se ha venido trabajando, y actualmente la optimización directa de medidas de desempeño en el aprendizaje se ha convertido en un novedoso e interesante tópico de investigación. Varios métodos para clasificación (Ej. [24]) y ordenación (Ej. [25], [26], [27], [28], [29], [30], [31], [32], [33], [34], [35], [36]) han sido propuestos sobre la base de diferentes enfoques.

Sin embargo, a pesar de todos estos avances y logros, y aunque los métodos que optimizan directamente medidas de desempeño, han propiciado un giro contundente en el estado del arte del Aprendizaje de la Ordenación, todavía las soluciones muestran dificultad para alcanzar una aproximación global, debido a que en diversas ocasiones la función objetivo es discontinua, estacionaria y presenta problemas de convexidad, por lo que el gradiente se indefine y los métodos sólo garantizan una aproximación local.

A esto se une el problema ya conocido de la variabilidad de la estabilidad, en cuanto a precisión en la ordenación, de los métodos referenciados para las diferentes colecciones de datos existentes.

Por lo cual, el camino del Aprendizaje de la Ordenación resulta en un tópico vigente de investigación para la comunidad científica.

1.5 Estudio Métrico

Resulta necesario realizar estudios para analizar el comportamiento de la producción científica, que se genera como resultado de las investigaciones a nivel mundial.

Para lograr resultados concretos y factibles, se han vinculado modelos matemáticos a la Bibliotecología y las Ciencias de la Información, dando lugar a disciplinas métricas (Bibliometría, Informetría, Cienciometría, etc.), que permiten medir, a partir de indicadores, el estado del arte de las ciencias, disciplinas y áreas del conocimiento, a través de la evaluación de la producción científica.

De esta manera resulta pertinente para los investigadores, en un momento de mucha actividad científica; poder evaluar la producción científica del tema que se investiga y a partir de entonces tener una visión global para iniciar nuevas líneas de trabajo, así como para continuar o redireccionar las ya existentes.

Este estudio métrico fue realizado sobre la base de un proceso de búsqueda y recuperación de registros bibliográficos relevantes de la Base de Datos Scopus en el período 2002-2012.

Las principales contribuciones de esta evaluación están encaminadas a brindar información relevante de la evolución y tendencias del Aprendizaje de la Ordenación.

1.5.1 Scopus: Base de Datos bibliográfica en línea

La veracidad de los resultados que se obtienen de un estudio bibliométrico, depende en gran medida, de la fuente de información de la cual se extraen los datos. Los trabajos publicados en las fuentes primarias suelen recopilarse en forma abreviada en las bases de datos. Es por ello que la consulta a las bases de datos apropiadas es un método adecuado para obtener información sobre las publicaciones y constituye la base para estudiar el comportamiento y las tendencias de la producción científica en una ciencia o disciplina.

Las bases de datos de mayor prestigio e impacto internacional, son **Web of Science** y **SciVerse Scopus** (comúnmente llamada Scopus). Si se toma como punto de referencia el universo de revistas científicas arbitradas que componen el directorio internacional de publicaciones seriadas Ulrich's, Web of Science procesa sólo el 25 % de ellas, mientras que Scopus abarca el 50%. Además, *Scopus* procesa el 95 % de las fuentes que ingresan al *Web of Science.* [37]

Scopus, creada en 2004 por Elsevier *B. V.,* es la mayor base de datos de citas y resúmenes de literatura arbitrada y de fuentes de alta calidad en el Web. Cubre cerca de 18 000 publicaciones seriadas de más de 5 000 casas editoras; 16 500 son revistas arbitradas. Contiene más de 40 millones de registros procedentes de publicaciones seriadas (revistas y series monográficas) y comerciales. Presenta además, una extensa cobertura de materiales de

conferencias (más de 3,6 millones), páginas Web en Internet (unos 318 millones) y patentes (23 millones). [37]

La retrospectividad del procesamiento de los artículos y sus referencias (necesarias para los análisis de citación) se remonta al año 1996, aunque existe una gran cantidad de artículos fuentes (es decir, sin sus referencias) de fechas anteriores.

Scopus tiene como fortaleza, el área de los estudios de citación en el contexto científico internacional, por lo que ha producido gran interés entre investigadores y académicos, tanto por su cobertura documental, como por su amigable interfaz y sus múltiples funcionalidades. Además, posee herramientas para seguir, analizar y representar el comportamiento de la actividad científica, mediante el empleo de los datos de citación de las obras y los autores. Por tal razón, se ha convertido en un fuerte competidor de los productos y servicios creados por el Institute for Scientific Information (ISI).[37]

Por las razones antes mencionadas se ha utilizado para el estudio métrico a Scopus como Base de Datos bibliográfica en la recuperación de registros relevantes acerca del Aprendizaje de la Ordenación.

1.5.2 Evolución de la temática "*Learning to Rank*"

La evolución de la producción científica sobre "*Learning to Rank*" en el período 2002-2012, se ha comportado de manera significativamente creciente, tal y como se muestra en la figura 1.3. Los puntos de la línea roja representa la cantidad de registros identificados en cada año, mientras que la línea negra representa la tendencia al aumento de dicha producción científica. Para el año 2005 se habían identificado solamente 4 registros sobre este tema , sin embargo , en el 2011 , la producción científica había aumentado a 117 registros. Para noviembre de 2012 se habían recuperado un total de 72 registros. [37]

Figura 1.3. Evolución del *Learning to Rank*. [37]

1.5.3 Temáticas más tratadas o exploradas en el "*Learning to Rank*"

Las temáticas más tratadas son aquellos temas más abordados en los artículos publicados sobre determinada área del conocimiento.

Para realizar el análisis de este indicador se seleccionó el campo descriptor. Al normalizar y procesar los mismos, se obtuvo un total de 1884 descriptores. De ello resultó que las temáticas más tratadas son: **Information Retrieval (IR)** (227), **Data sets** (92), **Ranking functions** (81), **Search engines** (74), **World Wide Web** (70), **Learning algorithms** (69), **Ranking model** (67), y así sucesivamente. [37]

Todos estos temas de mayor significación y que poseen una frecuencia de aparición superior a 30 registros en el período estudiado, son mostrados en la figura 1.4, donde se conoce de manera muy general, las temáticas más tratadas sobre el aprendizaje de la ordenación.

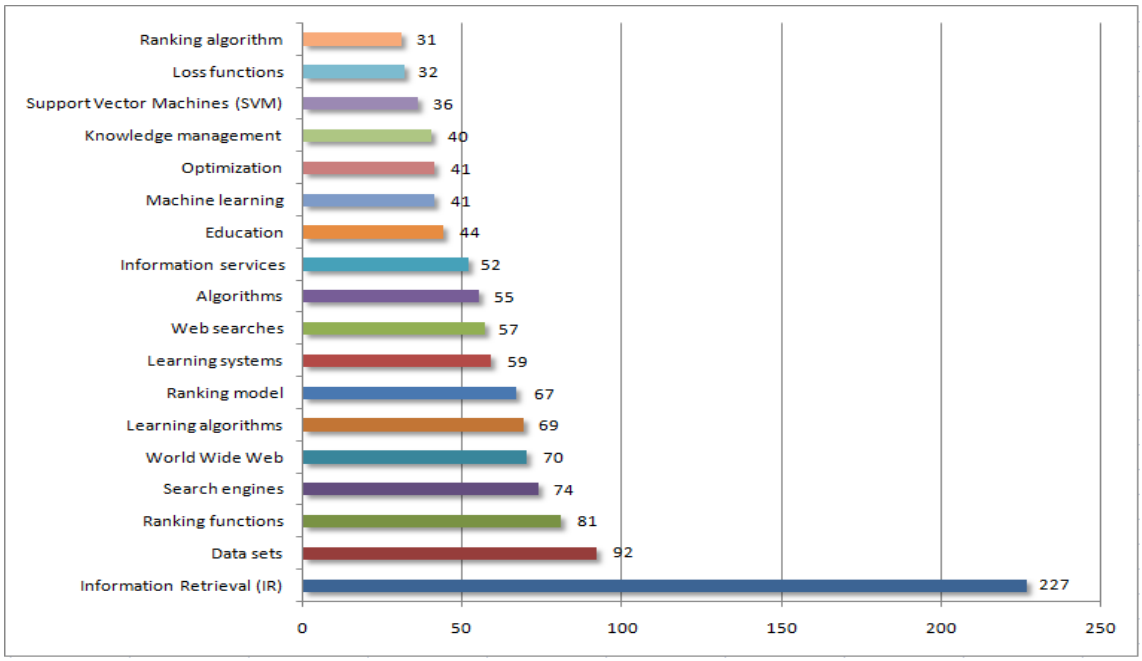


Figura 1.4. Temáticas más exploradas en L2R. [37]

1.5.4 Áreas del conocimiento en las que más se aplica el "*Learning to Rank*"

La figura 1.5 representa las áreas del conocimiento en las que más se aplica el Aprendizaje de la Ordenación. Como se observa, el área del conocimiento de mayor vinculación es **Computer Science** (ciencia de la computación), con un total de 369 publicaciones para un 55.9% del total recuperadas. Esto evidentemente está asociado al alto nivel de desarrollo y eficiencia que exigen las tecnologías y tendencias en el mundo de la ciencia de la computación, tanto para el progreso de la ciencia, como para la creación de herramientas y aplicaciones informáticas. [37]

El Aprendizaje de la Ordenación también ha sido aplicado con resultados prometedores en otras esferas de vital importancia como **Mathematics** (matemáticas) con 71 publicaciones, **Decision Sciences** (ciencias de la decisión) con 61, **Engineering** (ingeniería) con 54, **Business, Management and Accounting** (negocio, dirección y contabilidad) con 44, entre otras.

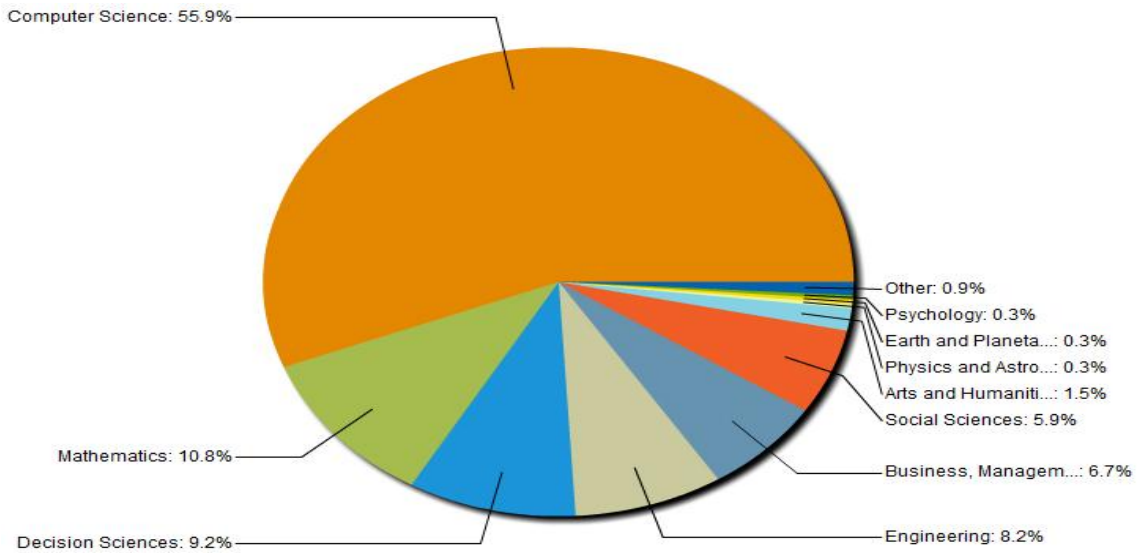


Figura 1.5. Áreas del conocimiento en las que más se aplica el "*Learning to Rank*". [37]

1.5.5 Productividad de autores en el "*Learning to Rank*"

La cantidad de autores identificados que abordan el tema de Aprendizaje de la Ordenación en mayor o menor grado es de 778. El análisis de este indicador arrojó como resultado, que los autores más productivos son Li, H. con 26 publicaciones, Liu, T. Y. con 21, Wang, Y. con 16, y así sucesivamente tal y como se muestra en la figura 1.6.

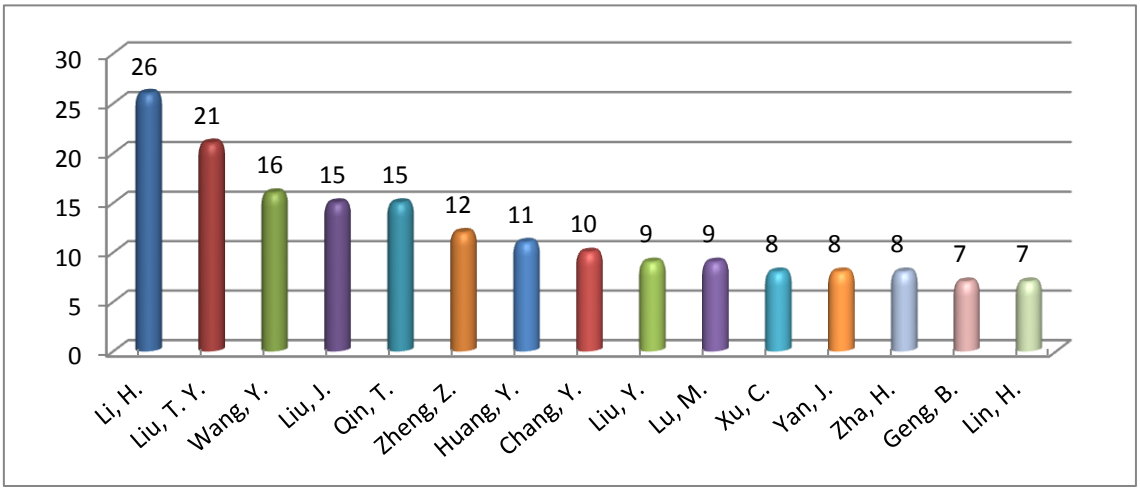


Figura 1.6. Autores de mayor productividad científica en "*Learning to Rank*". [37]

1.5.6 Países con mayor actividad científica en el "*Learning to Rank*"

El análisis de este indicador permitió conocer los países en los que más se produce información científica sobre el Aprendizaje de la Ordenación. La figura

1.7 representa, utilizando un gráfico circular, la cantidad de publicaciones correspondientes a cada país, destacándose como los más productivos del tema, **China** con 179 publicaciones y **Estados Unidos** con 126. Le siguen **Reino Unido** con 32, **Canadá** con 15 y así sucesivamente.

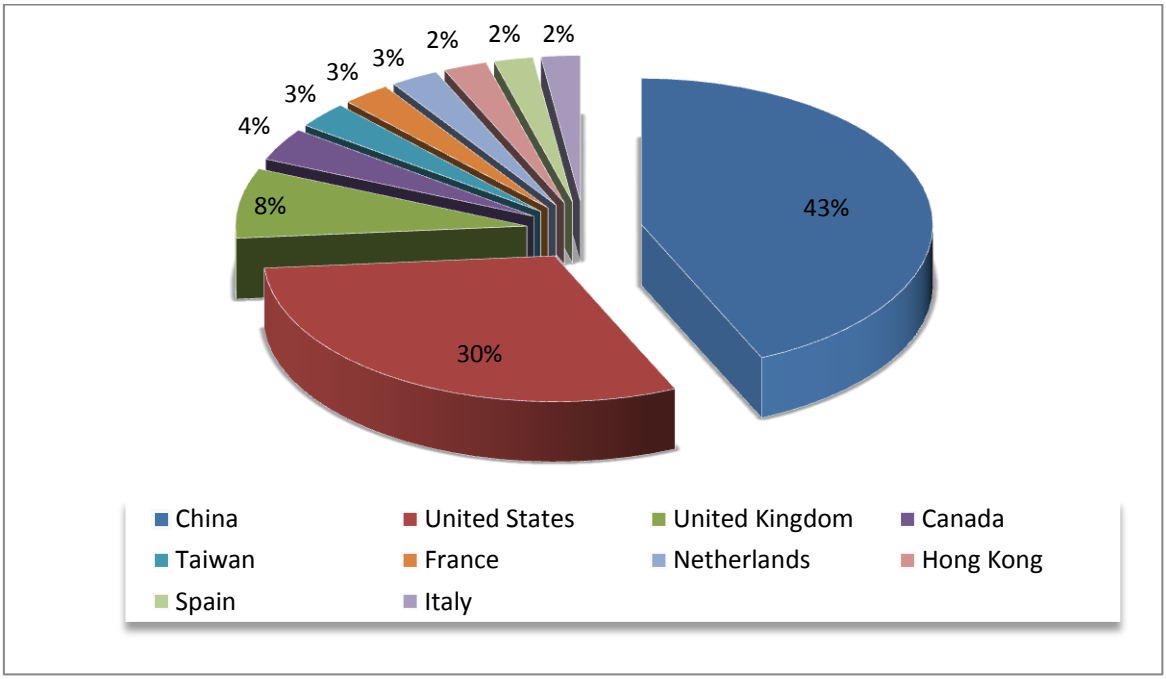


Figura 1.7. Países con mayor productividad científica en L2R. [37]

1.5.7 Temáticas en auge y estables

Se puede constatar que un total de 50 temáticas pueden ser consideradas en auge y estables; pues son aquellas que además de ser las más abordadas en los artículos publicados, tienen que cumplir la condición de haber sido tratadas continuamente en al menos cada uno de los últimos cinco años del período estudiado.

En los mapas correspondientes a las figuras 1.8 y 1.9, se puede apreciar la intensidad de la producción científica relacionada con cada temática en particular, para los años 2011 y 2012, respectivamente. Se seleccionó el año 2011 para ilustrar este comportamiento, pues en este período se alcanza la mayor producción de artículos relacionados con estas temáticas en auge y estables, y en sentido general, para el Aprendizaje de la Ordenación (ver apartado 1.5.2 Evolución de la temática "*Learning to Rank*"). Por su parte, la elección del año 2012 es evidente, por ilustrar el estado actual de tales temáticas. Es importante mencionar que si se analizan los mapas topológicos

por años del período que se estudia con respecto a cada temática, pues notaremos que tales temas son estables en su aparición y en la relación que se establece entre cada una. [37]

Para facilitar la interpretación de las figuras 1.8 y 1.9, las temáticas fueron etiquetadas de la siguiente manera. Las temáticas de color negro sin enfatizar representan aquellas temáticas en las que la producción de artículos acumulados en todos los años comprendidos hasta ese instante de tiempo fue menor que 50, las enfatizadas en color negro corresponden con producciones entre 50 y 100 artículos, y por su parte, las enfatizadas en color rojo (en este caso, sólo una) son las tratadas en más de 100 publicaciones. [37]

Se puede decir entonces, que la temática más tratada fue **Information Retrieval (IR)** (recuperación de información) con un total de 227 artículos asociados. Las temáticas que más prevalecen con una cantidad de 50 a 100 publicaciones son: **ranking functions** (funciones de ordenamiento), **ranking model** (modelo de ordenamiento), **learning algorithms** (algoritmos de aprendizaje), **data sets** (conjuntos de datos), **search engines** (motores de búsqueda), **world wide web**, **web searches** (búsquedas web), **algorithms** (algoritmos) y **learning systems** (sistemas de aprendizaje). [37]

La proximidad entre los clúster en los que se enmarca cada temática nos muestra además, la relación existente entre cada una de ellas en particular, así como la fuerte vinculación entre las más tratadas o exploradas en cada año.

Por lo cual, también se puede concluir, que entre estas temáticas relacionadas existe un fuerte lazo y que en menor medida, pero con una fuerte vinculación se abordaron otros temas como: **knowledge management** (gestión del conocimiento), **Natural language processing systems** (sistemas de procesamiento del lenguaje natural), **ranking algorithms** (algoritmos de ordenamiento), **support vector machines** (máquinas de soporte vectorial), y así sucesivamente se pueden seguir asociando otras temáticas. [37]

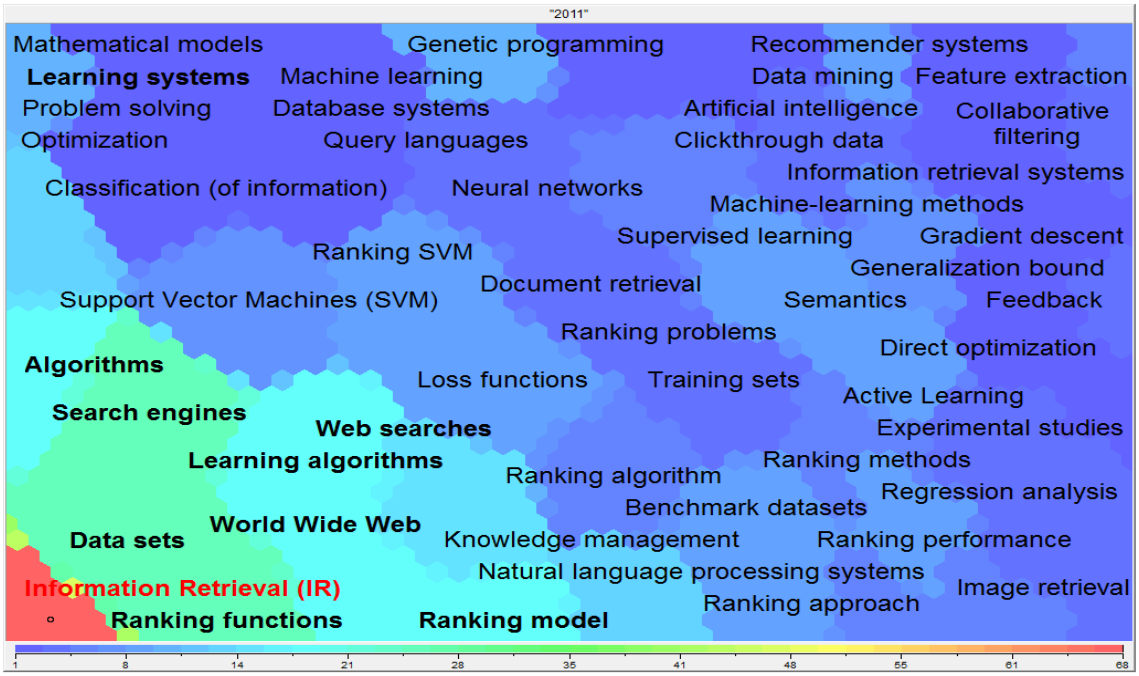


Figura 1.8. Temáticas en auge y estables en el año 2011. [37]

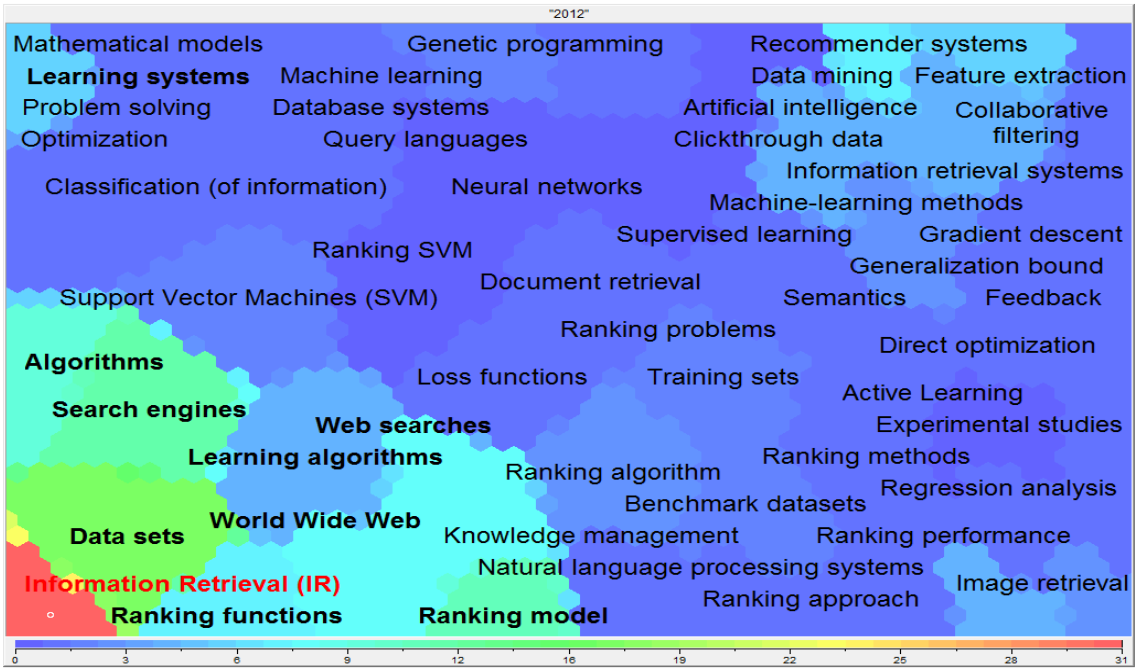


Figura 1.9. Temáticas en auge y estables en el año 2012. [37]

1.6 Fundamentación de la metodología utilizada.

Para el desarrollo de la investigación se emplea una metodología mixta ya que se utilizan elementos de la metodología cuantitativa y cualitativa. El enfoque cualitativo permite la interpretación del fenómeno y el enfoque cuantitativo, su

explicación. Esta tendencia se está utilizando cada vez con mayor frecuencia en las investigaciones.

La investigación cualitativa propone la estancia prolongada en el campo y la observación persistente de los focos principales de la investigación. Se recogen las informaciones y luego se analizan desde distintos ángulos a fin de contrastarlos, realizando el cruzamiento e interpretando y hallando las coincidencias de los resultados alcanzados.

Se consideran de utilidad los métodos cuantitativos, ya que los números constituyen instrumentos de modelación de la realidad, de aproximación al conocimiento y contribuyen a ordenar la información en el proceso de producción del conocimiento.

La metodología mixta posibilita aprovechar las tendencias, la direccionalidad que pueden mostrar algunos métodos cuantitativos sobre la base del análisis cualitativo, para cruzar y verificar la información obtenida aplicando diferentes métodos. Por eso se considera que la aproximación a la verdad está vinculada con la pluralidad metodológica, ya que el método debe poseer la capacidad de reflejar el movimiento, la dinámica, la esencia del objeto y esto no se logra sin la participación compensatoria de los diversos métodos existente. [38]

Entre los métodos cualitativos y cuantitativos empleados en este trabajo se encuentran la modelación algorítmica, la experimentación y el análisis estadístico.

Los métodos científicos, según su etimología, son el camino hacia el conocimiento. En la presente investigación se utilizan varios de estos métodos:

- Nivel teórico.
  - Inducción – deducción: La inducción expresa el movimiento de lo particular a lo general, o sea se llega a generalizaciones partiendo del análisis de casos particulares, mientras la deducción expresa el movimiento de lo general a lo particular. La combinación de ambos permite estructurar el conocimiento científico obtenido en la revisión bibliográfica.

- Histórico – lógico: El método lógico debe basarse en los datos que proporciona el método histórico, de manera que no se convierta en un simple razonamiento especulativo. De igual forma, lo histórico no debe limitarse a la simple descripción de los hechos, sino explicarlos a partir de la lógica de su desarrollo [39]. Este método da la posibilidad de conocer el problema de la ordenación en la Recuperación de Información en su origen y desarrollo, y proyectar el análisis de la evolución histórica de los métodos de Aprendizaje de la Ordenación con la lógica de su comportamiento futuro.
- Análisis y síntesis: A través del análisis se distinguen y revisan por separado los elementos relacionados con el Aprendizaje de la Ordenación para la R.I. Partiendo de esto, se emplea la síntesis para establecer nexos, comparar resultados, determinar características comunes y aspectos distintivos de los diferentes enfoques estudiados, lo que permite arribar a conclusiones.
- Comparación: Para establecer las similitudes y diferencias entre los modelos y categorías existentes dentro del Aprendizaje de la Ordenación para la R.I.

➢ Nivel matemático y estadístico.

- Métodos de la Estadística: Con la utilización de la estadística descriptiva (distribución de frecuencias, medidas de tendencia central y medidas de la variabilidad) y las pruebas no paramétricas para muestras relacionadas o pareadas, es posible evaluar el desempeño del método RankFSP y comparar sus resultados con los métodos referenciados en la literatura.

1.7 Tendencias y herramientas de desarrollo.

Para la implementación del método de Aprendizaje de la Ordenación RankFSP se utilizó Java como lenguaje de programación, debido a que es multiplataforma y cumple con todos los requerimientos necesarios para desarrollar nuevos modelos. Además, se piensa incorporarlo a L2RLab [40], un entorno integrado de experimentación para la tarea del Aprendizaje de la Ordenación. Otro motivo es lograr uniformidad con la comunidad científica ya

que la mayoría de los métodos desarrollados están programados en Java o se pretenden programar en este lenguaje.

Java se está moviendo rápido al plano de los lenguajes tradicionales de inteligencia artificial como Lisp y Prolog. Hay ya varios programas y APIs en Java para técnicas de *SoftComputing* y aprendizaje automático, de libre distribución y disponible en línea como GPL (*open source*). Esto muestra la popularidad explosiva de Java en el campo de inteligencia artificial y *SoftComputing*. [41]

Como IDE (entorno integrado de desarrollo) de programación se emplea Eclipse, que definido por la Fundación del mismo nombre es: "*una especie de herramienta universal - un IDE abierto y extensible para todo y nada en particular*". La versión que se utiliza en este trabajo es portable, por lo que a la ventaja de ser libre, se le suma la posibilidad de estar ejecutando a la vez varias versiones del método propuesto lo que permite ahorrar tiempo en la etapa de experimentación, ajuste de parámetros y pruebas con las distintas colecciones de datos.

1.8 Conclusiones

En el presente capítulo se evidencia como los métodos tradicionales de Recuperación de Información, han ido cediendo su espacio a nuevas técnicas de aprendizaje automático, debido a la necesidad imperante de afrontar nuevos retos y problemas de ordenación que exigen mayor precisión para lograr la satisfacción del usuario.

Es importante resaltar, después de haber analizado cada categoría, que en los últimos años se observa una tendencia hacia los métodos que optimizan directamente medidas de desempeño, pues estos brindan elementos teóricos y resultados prometedores en cuanto al rendimiento alcanzado en la ordenación con respecto a las propuestas tradicionales.

A pesar de estos avances, los métodos desarrollados muestran dificultad para alcanzar una aproximación global y precisiones estables

a nivel de consulta. Todo lo cual dirige su atención a utilizar nuevas técnicas propias o híbridas de *SoftComputing*; en tal caso, FSP es un algoritmo de optimización global que puede dar respuesta a esta problemática.

Para el desarrollo de esta investigación se emplea una metodología mixta, aprovechando las ventajas que brinda la integración de los métodos cualitativos y cuantitativos.

Capítulo 2: FSP en el Aprendizaje de la Ordenación

2.1 Introducción

En este capítulo se describen las principales características del procedimiento de búsqueda "El Pescador", así como su modelación algorítmica. Esta metaheurística se adapta al problema del Aprendizaje de la Ordenación, proponiéndose un nuevo método que permite optimizar directamente las medidas de desempeño utilizadas en el Aprendizaje de la Ordenación para la R.I.

2.2 Optimización global con FSP

2.2.1 Introducción

FSP (Fisherman Search Procedure). Es un método de optimización global que explora nuevas soluciones combinando la búsqueda guiada y la búsqueda local. Esta metaheurística fue diseñada con el propósito de desarrollar soluciones útiles y prácticas para una variedad de problemas de optimización combinatoria.

Las principales ventajas son: la fácil implementación, proporciona una descripción explícita de la idea y el modelo basado en la propia concepción de esta metaheurística, puede ser aplicado a un gran número de problemas de optimización, así como a los que surgen en situaciones del mundo real en diferentes áreas de ciencias aplicadas, ingeniería y economía.

2.2.2 Algoritmo clásico

El Procedimiento de Búsqueda el Pescador (*Fisherman Search Procedure*, FSP) es un método de optimización global, inspirado en el comportamiento cognoscitivo y las destrezas y habilidades observadas en un pescador [42] .

Este método explora nueva soluciones usando una combinación de búsqueda guiada y búsqueda local.

Inicialmente, se definen un conjunto de N puntos de captura en toda la región de pesca (espacio de búsqueda para el problema en cuestión).

Básicamente cada punto de captura está compuesto por un vector de posición, x_i (localizado en el espacio de búsqueda) y una memoria de la mejor solución encontrada por el pescador en la vecindad del punto de captura p_i. Se tiene que $x_i \in X$, donde $X = \{x_1, x_2,..., x_N\}$ denota el conjunto de vectores de posición de los puntos de captura.

La memoria global de pescador es definida como g_{best} (es decir, la mejor solución encontrada entre todos los puntos de captura).

A partir de entonces se define una trayectoria de pesca que abarque cada uno de los puntos de captura (es decir, el pescador parte del punto x_1 hasta el x_2 y así consecutivamente).

En cada punto de pesca el pescador lanza su maya (red de pesca) L veces.

La maya del pescador está compuesta por un conjunto de vectores de posición $y_{ij} \in Y$, $Y = \{y_{i1}, y_{i2},... y_{iM}\}$, creados y definidos a partir de un punto de referencia (el punto de captura x_i correspondiente), donde y_{ij} denota el $j^{ésimo}$ vector en x_i, i = 1, 2, ..., N; j = 1, 2, ..., M. La maya del pescador es referenciada con el valor M, el cual representa la cantidad de vectores de posición de la red.

La expresión usada para crear los vectores de posición de la red de pesca es la siguiente:

$$y_{ij} = x_i + A_j \tag{2.1}$$

donde A_j es un vector *n*-dimensional compuesto por números aleatorios en el rango [-c,c], siendo c un número real denominado coeficiente de amplitud.

Los vectores de posición son evaluados y si alguno de estos logra un valor de aptitud (*fitness*) mejor que el valor p_i del punto de captura a partir del cual se lanzó la maya, entonces la referencia que tiene el pescador de este determinado punto de captura es actualizada con esta nueva posición y la maya se redefine considerando este nuevo vector de referencia.

Cuando se actualiza el valor de p_i para el $i^{ésimo}$ punto de captura, si p_i es mejor que el g_{best}, este último es también actualizado.

Todo este procedimiento de pesca es repetido T veces y es ilustrado en la figura 2.1.

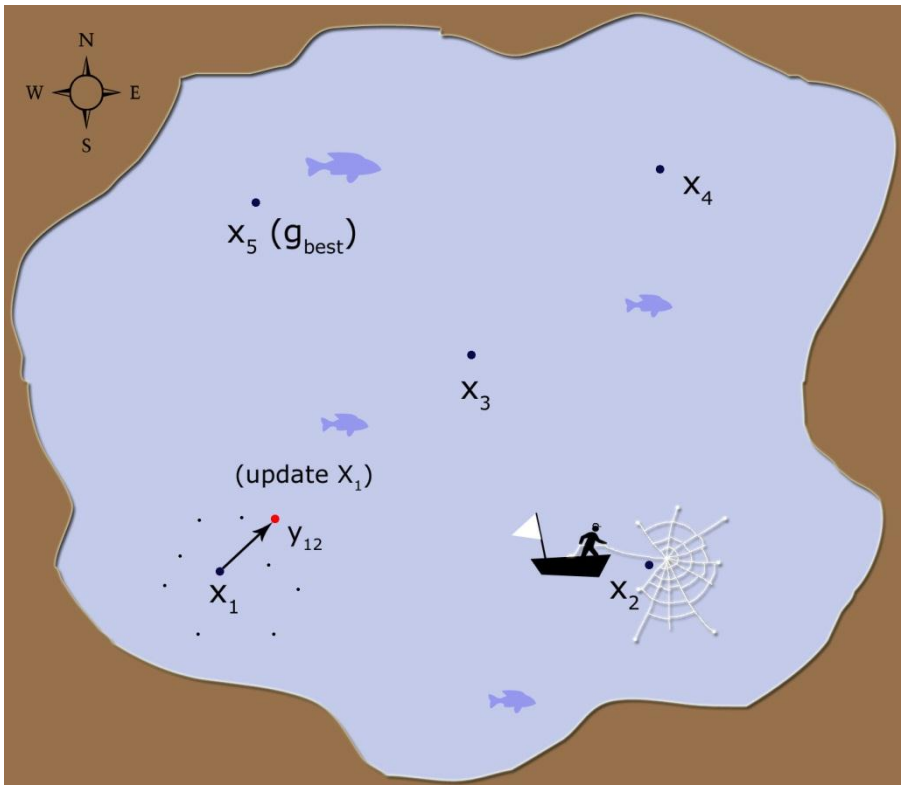


Figura 2.1. Esquema gráfico que representa el desempeño del pescador en un espacio unidimensional.

Esta cantidad de iteraciones tiene un criterio de parada acorde a la convergencia del algoritmo: x número de iteraciones sin mejorar la precisión en la ordenación puede llevar a terminar el proceso.

Luego de haber avanzado la pesca, después de cada iteración, la tasa de mejoras en cada punto de captura es analizada. Si el valor de la función en p_i para el $i^{ésimo}$ punto de captura mejora con respecto a la iteración previa, los autores recomiendan disminuir gradualmente el valor del coeficiente de amplitud, con la idea de reducir el riesgo de saltarse un óptimo global, en el caso de que la solución esté cerca. Esta estrategia persigue la idea intuitiva de ir compactando la maya de pesca para garantizar la captura.

Dado el caso contrario, que el valor de la función en p_i para el $i^{ésimo}$ punto de captura no mejora, el valor del coeficiente de amplitud es aumentado con la idea de explorar nuevas zonas de pesca. A partir de entonces, cuando se

encuentra otra buena solución, el valor del coeficiente de amplitud es fijado a su valor inicial y comienza de nuevo su chequeo.

2.2.3 Variantes de FSP

Considerando las concepciones inicial de FSP, los autores proponen 4 variantes del método que pueden ser discutidas, implementadas y evaluadas en trabajos futuros.

En la *variante #1*, el pescador actualiza la posición de un específico punto de captura *i* solamente cuando uno de los puntos de la maya mejora el valor de la función en p_i. Sobre la base de esta idea, si después de un lanzamiento de la maya el valor de la función en p_i no mejora, no se realizan otros lanzamientos en el correspondiente punto de captura hasta la próxima iteración.

Cada vez que la maya es lanzada y el valor de la función en p_i para el *i-ésimo* punto de captura es mejorada, los valores relativos a la cantidad de vectores de la red de pesca y el coeficiente de amplitud son decrementados en un factor específico (0.99). En la próxima iteración del método los valores de estos dos parámetros son fijados nuevamente a su valor por defecto.

En un principio, la *variante #2* es similar a la extensión #1, debido a que el pescador no actualiza su posición para la pesca (punto de captura *i*) a menos que uno de los vectores de la maya (lanzados a partir de *i*), mejore el valor de la función en p_i. Mientras la posición del punto de captura se mantenga mejorando, la red de pesca es lanzada continuamente y el valor del coeficiente de amplitud son disminuidos según el factor de la variante #1.

Después de cada iteración, aquellos puntos de captura cuya posición no mejoró son redefinidos a nuevas posiciones de pesca. Este reinicio puede ser aleatorio o guiado hacia zonas de pesca inexploradas.

En la *variante #3*, en cada punto de captura se realizan todos los lanzamientos de la maya. Diferente a otras variantes, la cantidad de lanzamientos no está condicionada a mejoras de posición. En este caso, cada vez que la maya es lanzada en un determinado punto de captura, la posición de este punto de pesca es siempre actualizada con el mejor punto obtenido en la red hasta ese momento.

Esta idea está basada en la intuición que un gran pez (solución óptima) puede estar en zonas donde vivan pequeños peces (soluciones no tan buenas), de los cuales este se alimenta.

Por su parte en la *variante #4*, se parte por dividir la zona de pesca (espacio de búsqueda) en subregiones. A cada una de estas subregiones se envía un pescador, el cual puede aplicar como estrategia de pesca su propia variante. Esta cooperativa de pescadores comparte una memoria común donde es almacenada la mejor posición (g_{best}) lograda finalmente entre todos los pescadores.

Aunque todavía el paralelismo no se ha usado sistemáticamente para acelerar o mejorar la efectividad de las metaheurísticas, las aplicaciones paralelas son muy robustas y abundan en la literatura (consultar en [43] un estudio al respecto).

En general, cada variante tiene sus ventajas y desventajas. En la variante #1 el pescador trata de delimitar la mejor posición de pesca a través de grandes pasos, pero puede caer atrapado en un mínimo local. A pesar de las similitudes con la variante #1, en la variante #2 el pescador usa la reinicio de los puntos de captura como una estrategia clave para evitar una convergencia temprana en mínimos locales. Por otro lado, la variante #3 está concebida con la idea de que la trayectoria, pasando a través de zonas de pesca no muy prometedoras, pudiese llevarnos a las mejores zonas de captura, las que pudiesen ser inaccesibles de otro modo (según la actuación de otra variante). La variante #4 sugiere una aplicación paralela para reforzar la robustez del método.

Finalmente, se puede concluir que la selección y aplicación de las variantes de FSP mencionadas, y otras, dependen de las características y situaciones a donde se pudiesen encaminar futuros problemas de optimización.

2.3 Método Propuesto. RankFSP

2.3.1 Formulación general

Supongamos que $Q = \{q_1, q_2, \dots, q_m\}$ es el conjunto de consultas en el entrenamiento, *D* constituye la colección documental y que $Y = \{r_1, r_2, \dots, r_k\}$ es el conjunto de juicios de relevancia. Un juicio de relevancia constituye una medida de cuan bueno es un documento recuperado con respecto a las necesidades de un usuario, reflejadas en la consulta formulada.

Cada consulta q_i (representada como una lista de términos $\{t_1, t_2, \dots, t_{h(q_i)}\}$ donde $h(q_i)$ denota la cantidad de términos para la i-ésima consulta) está asociada con una lista de documentos recuperados $d_i = \{d_{i_1}, d_{i_2}, \dots, d_{i_{n(q_i)}}\}$ y una lista de etiquetas $y_i = \{y_{i_1}, y_{i_2}, \dots, y_{i_{n(q_i)}}\}$, donde $n(q_i)$ denota el tamaño de la lista; $d_i \subseteq D$ e $y_i \subseteq Y$ para la consulta $q_i \in Q$, $d_{ij} \in d_i$ indica el j-ésimo documento en d_i, así como $y_{ij} \in y_i$ denota la etiqueta del documento d_{ij}. De esta manera cada documento d_{ij} está representado como una lista de rasgos o características.

Finalmente el conjunto de entrenamiento queda definido como sigue:

$$S = \{(q_i, d_i, y_i)\}_{i=1}^{m} \tag{2.2}$$

El ordenamiento se concentra en obtener una predicción para una consulta dada q_i y la lista de documentos asociados d_i, usando el modelo de ordenación.

El modelo de ordenación, denotado como f, es una función a nivel de documento, la cual constituye una combinación lineal de los rasgos en un vector de rasgos $\emptyset(q_i, d_{ij})$:

$$f(q_i, d_{ij}) = w^T \emptyset(q_i, d_{ij}) \tag{2.3}$$

Donde w denota el vector de peso y $\emptyset(q_i, d_{ij})$ es el vector de rasgos creado para cada par consulta-documento (q_i, d_{ij}), donde $i = 1, 2, \dots, m$ y $j = 1, 2, \dots, n(q_i)$. En la ordenación para la consulta q_i se asigna una puntuación para cada uno de los documentos usando la función $f(q_i, d_{ij})$

y ordenando posteriormente tales documentos en correspondencia con las puntuaciones que le fueron dadas.

Así se obtiene una predicción denotada como π_i, que es la predicción realizada por el modelo de ordenación sobre d_i en términos de la consultaq_i. Se emplea Π_i para denotar el conjunto de todas las predicciones posibles sobre d_i, y se usa $\pi_i(j)$ para denotar la posición del elemento j, por ejemplo d_{ij}. El ordenamiento se concentraría en obtener una predicción $\pi_i \in \Pi_i$ para una consulta dada q_iy la lista de documentos asociados a d_i, usando el modelo de ordenación.

En la R.I, las medidas de desempeño son usadas para evaluar la efectividad de un modelo de ordenación, el cual usualmente está basado en consultas. Esto significa que la medida se define sobre una lista de documentos ordenados con respecto a una consulta. Entre las medidas de desempeño más empleadas están: MAP (*Mean Average Precision*) [17], NDCG (*Normalized Discounted Cumulative Gain*) [18], y P@n (*Precision @t n*) [17]. Para un análisis más detallado, ver el apartado 3.3 Medidas de Evaluación.

En este trabajo se utiliza una función general $E(\pi_i, y_i) \in [0, +1]$ para representar las medidas de evaluación. El primer argumento de E es la predicción π_i creada usando el modelo de ordenación. El segundo argumento y_i es la lista de juicios que determinan el orden ideal. E mide la correspondencia entre π_i e y_i. La mayoría de las medidas de desempeño retornan valores reales entre $[0, +1]$.

La predicción ideal puede ser denotada como π_i^*.

Ahora, como puede existir más de una predicción ideal para una consulta, se denota Π_i^* como el conjunto de posibles predicciones ideales para la consulta q_i. Por tanto $\pi_i^* \in \Pi_i^*$ y se tiene que $E(\pi_i^*, y_i) = 1$.

En la Tabla 2.1 se presenta un resumen de las notaciones descritas anteriormente.

Tabla 2.1. Resumen de notaciones para el modelo de Aprendizaje de la Ordenación

Notación	Descripción
$q_i \in Q$	Consulta
$d_i = \{d_{i_1}, d_{i_2}, \dots, d_{i_{n(q_i)}}\}$	Lista de documentos recuperados para q_i
$d_{ij} \in d_i$	*j*-ésimo documento en d_i
$y_i = \{y_{i_1}, y_{i_2}, \dots, y_{i_{n(q_i)}}\}$	Lista de juicios para d_i con respecto a q_i
$y_{ij} \in \{r_1, r_2, \dots, r_k\}$	Juicio de d_{ij} con respecto a q_i
$S = \{(q_i, d_i, y_i)\}_{i=1}^{m}$	Conjunto de entrenamiento
$\pi_i \in \Pi_i$	Predicción del modelo de ordenación para q_i
$\pi_i^* \in \Pi_i^*$	Predicción ideal para q_i
$\emptyset(q_i, d_{ij})$	Vector de rasgos para (q_i, d_{ij})
f	Modelo de ordenación
$E(\pi_i, y_i) \in [0, +1]$	Evaluación de π_i con respecto a y_i para q_i

Idealmente, el objetivo del modelo de ordenación es maximizar la precisión alcanzada sobre un conjunto de entrenamiento, en términos de una medida de desempeño de R.I.

Para ello, podemos plantear la minimización de una función de pérdida básica definida en [27] como sigue:

$$\boldsymbol{R(f) = \sum_{i=1}^{m}(E(\pi_i^*, y_i) - E(\pi_i, y_i)) = \sum_{i=1}^{m}(1 - E(\pi_i, y_i))} \quad (2.4)$$

donde π_i es la predicción determinada por el modelo de ordenación para la consulta q_i.

2.3.2 Algoritmo

En este apartado, un nuevo método de Aprendizaje de la Ordenación para la R.I es descrito formalmente. Este método está basado en FSP y es capaz de optimizar cualquier medida de evaluación usada en la R.I. Como se mencionó

anteriormente, el algoritmo se denomina RankFSP y se muestra en la figura 2.2.

1: Entrada: $S = \{(q_i, d_i, y_i)\}_{i=1}^{m}$, E, T, N, L y M
2: **Desde** $i = 1$ hasta N **hacer**
3: Inicialización aleatoria $x_i = p_i$
4: Actualizar g_{best}
5: **Fin Desde**
6: **Desde** $i = 1$ hasta T **hacer**
7: **Desde** $j = 1$ hasta N **hacer**
8: *Mientras* (($L \neq 0$) y (Se mejore el p_j)) *hacer*
9: **Desde** $k = 1$ hasta M **hacer**
10: Inicialice y_{jk} con la expresión (3)
11: Evalúe y_{jk} en S, con la expresión (2)
12: **Fin Desde**
13: Actualizar x_j, p_j
14: *Fin Mientras*
15: Actualizar g_{best}
16: **Fin Desde**
17: Cambiar la posición de los puntos de captura no mejorados
18: **Fin Desde**
19: Construir el modelo de ordenación f basado en vector de posición g_{best}
20: Salida: f

Figura 2.2. Algoritmo RankFSP

Los parámetros de entrada de RankFSP son: un conjunto de entrenamiento S, una medida de desempeño E, un número de iteraciones T, una cantidad de puntos de captura N, un número de lanzamientos L y la red de pesca M. El valor de M representa la cantidad de puntos (vectores de posición) de la red de pesca.

Antes de explicar el método RankFSP, es importante describir como se representan y ajustan cada uno de los elementos del modelo de FSP en la concepción de este método de Aprendizaje de la Ordenación.

Como el algoritmo propuesto usa una función de ordenación lineal basada en rasgos (atributos extraídos de un documento recuperado a partir de una determinada consulta), el tamaño de un punto de captura (vector de posición) está determinado por el número de rasgos. Cada punto de captura constituye una posible solución, este punto es usado como vector de pesos en la función

de ordenación (vea la sección 3.4, para más detalles acerca de la configuración de los parámetros).

Además, resulta importante mencionar que en la adaptación al problema del Aprendizaje de la Ordenación, los elementos considerados del método FSP se corresponden con su variante # 2.

Primero, RankFSP crea un conjunto específico de puntos de captura e inicializa cada uno de estos aleatoriamente; siempre actualizando el vector de posición g_{best}en cada iteración. Entonces, el pescador comienza su trayectoria de pesca. T rondas son ejecutadas en las cuales el pescador lanza su red de pesca en cada uno de los puntos de captura, con el objetivo de encontrar el mejor vector de posición g_{best} obtenido en el área de pesca (espacio de búsqueda). En cada punto de captura x_j , los vectores de posición de la red de pesca y_{jk} son evaluados con respecto al conjunto de entrenamiento correspondiente. La ecuación 2.4 representa la función de aptitud usada para evaluar la posición de cada vector de posición. Esta función de aptitud usa la medida de desempeño E y la predicción π_j obtenidas a partir de la aplicación de la expresión 2.1 sobre S. Al final de cada iteración el valor del vector de posición x_j y p_j del punto de captura es actualizado.

Mientras que el pescador encuentre mejores vectores de posición con cada lanzamiento de la maya y no haya agotado los L lanzamientos establecidos, puede continuar pescando en el punto de captura. Cuando el pescador termina la pesca, el valor del punto de captura del mejor vector de posición g_{best} es actualizado.

Durante la pesca, el algoritmo implementa una estrategia para evitar una convergencia temprana hacia mínimos locales. Esta estrategia consiste en modificar la posición de los puntos de pesca no mejorados durante la pesca. Con esta meta los valores de precisión alcanzados por cada punto de captura en la última iteración G son almacenados. Estos valores permiten analizar el comportamiento de las soluciones encontradas por el pescador en un determinado punto de captura para un rango de iteraciones. A partir de estos valores se calcula la tasa de variación, como la diferencia entre el primero y

último valor, dividido por la cantidad de valores. La tasa de variación nos da una medida de cuanto ha variado el desempeño que alcanza el pescador en un punto de captura con respecto al conjunto de entrenamiento. Por esta razón, se puede usar este valor calculado como un indicador para concluir o no que los puntos de captura están atrapados en un mínimo local.

Para controlar estos dos estados (atrapado y no atrapado), se define un umbral de cambio, denominado como th, cuyos valores son fijados en [0,1]. En este sentido, si el valor resultante es menor que th (por ejemplo, 0.01), se puede considerar que estamos en presencia de una convergencia temprana, y se procede a cambiar la posición del correspondiente punto de captura.

El cambio de posición de los puntos de captura puede ser ejecutado de dos maneras:

- La primera resulta en inicializar aleatoriamente la posición de los puntos de captura

- La segunda consiste en mover de forma controlada la posición de los puntos de captura de tal forma que abarquen nuevas zonas de pesca. En este caso, sería necesario conocer cuales áreas de pesca han sido exploradas y cuáles no.

Para el caso contrario, donde el valor resultante es mayor o igual al umbral de cambio th, no se procede a realizar operación alguna sobre los puntos de captura. Este proceso de medir la variabilidad del desempeño de los puntos de captura en un rango de G iteraciones, será repetido hasta la última ronda de iteraciones del algoritmo.

Según pruebas empíricas para la tarea del Aprendizaje de la Ordenación, el parámetro G, puede ser fijado en un valor entre 5 y 10 iteraciones. El valor de th puede ser fijado en correspondencia con el comportamiento del pescador sobre la colección de datos analizada y la variabilidad de desempeño de los vectores de posición, considerando la cantidad de estos. Se recomienda para las colecciones de datos analizadas en este trabajo, fijar el valor de th entre 0.01 y 0.05.

Finalmente, el modelo de ordenación f es construido con el vector de posición g_{best} obtenido en la última ronda.

Se puede concluir que RankFSP es un método de Aprendizaje de la Ordenación el cual puede ser clasificado dentro de los métodos que optimizan directamente medidas de desempeño, es decir, dentro de la categoría de aproximación por listas. Específicamente este pudiese ser localizado en la sub-categoría de aquellos métodos que usan tecnologías especialmente diseñadas para optimizar medidas de R.I no suaves.

2.4 Conclusiones

En este capítulo se presentó RankFSP, un novedoso método de Aprendizaje de la Ordenación, que es capaz de optimizar directamente cualquier medida de desempeño de la R.I.

Este método se basa en FSP, metaheurística funcional para resolver problemas de optimización global, que explora nuevas soluciones usando una combinación efectiva de búsqueda guiada y búsqueda local.

Además de todas las características ventajosas de FSP, en la concepción del método RankFSP, se adicionó una nueva estrategia basada en el reinicio de los puntos de pesca para evitar una convergencia temprana hacia mínimos locales.

Capítulo 3: Resultados experimentales

3.1 Introducción

En este capítulo se presentan los resultados experimentales de la evaluación del método propuesto en términos del rendimiento (precisión) alcanzado en la ordenación para conjuntos de datos seleccionados de LETOR[1] 3.0 (tomados de las colecciones de OHSUMED y Gov) y LETOR[1] 4.0 (MQ2007 y MQ2008). Se consideraron como criterios para evaluar el desempeño del algoritmo en la ordenación, las medidas P@n, MAP y NDCG, ampliamente utilizadas en la R.I.

Este estudio experimental se basó en una comparación directa con los principales métodos que establecen actualmente el estado del arte del Aprendizaje de la Ordenación. Finalmente, se tomaron los resultados arrojados por cada algoritmo considerado y se realizaron pruebas no paramétricas utilizando un modelo estadístico de comparación de medias para muestras relacionadas o pareadas.

3.2 Colecciones de datos utilizadas

Con el creciente desarrollo de modelos para el Aprendizaje de la Ordenación, se hizo necesario crear conjuntos de datos de referencia que pudieran usarse en la comparación de algoritmos de aprendizaje existentes y en la evaluación de los nuevos modelos que surgían.

Para tratar este problema Microsoft Research Asia[2] propone LETOR.

LETOR es un estándar de referencia para la investigación en Aprendizaje de la Ordenación [44]. Contiene los rasgos típicos, los juicios de relevancia, las particiones de datos, las herramientas de evaluación y varios métodos de Aprendizaje de la Ordenación de referencia.

La versión 1.0 fue lanzada en Abril de 2007, la 2.0 en Diciembre de ese mismo año, en Diciembre de 2008, la versión 3.0 y la versión 4.0 está disponible desde julio de 2009. En esta investigación se utilizan conjuntos de datos de las

[1] Disponible en: http://research.microsoft.com/en-us/um/beijing/projects/letor/
[2] Disponible en: http://research.microsoft.com/en-us/labs/asia/default.aspx

colecciones OHSUMED, Gov, Gov2, TREC 2007 y TREC 2008 pertenecientes a las versiones 3.0 y 4.0 de LETOR.

Estas colecciones de datos contienen consultas, el contenido de los documentos recuperados, y juicios humanos de relevancia de los documentos con respecto a las consultas. Con vistas a la tarea del Aprendizaje de la Ordenación se extrajeron rasgos de estos conjuntos de datos, incluyendo rasgos convencionales (frecuencia de término, frecuencia inversa del documento, longitud del documento, BM25, y modelo de lenguaje para R.I.) y nuevos rasgos propuestos por SIGIR[3] (*HostRank*, propagación del rasgo y *PageRank*). Finalmente se empaquetaron en LETOR estos rasgos extraídos, las consultas y los juicios de relevancia. [17]

Con la intensión de potenciar una comparación directa entre modelos, se prefijó un esquema de experimentación, donde fue considerada una validación cruzada con 5 particiones, incluyendo en cada partición un subconjunto de entrenamiento, validación y prueba.

3.2.1 LETOR OHSUMED

La colección OHSUMED fue creada para la investigación en R.I. Esta es un subconjunto de Medline, una base de datos sobre publicaciones médicas. La colección consta de 348 566 registros de 270 revistas médicas, correspondientes al período de 1987-1991. [17]

En OHSUMED hay 106 consultas sobre necesidades de búsqueda médica y cada una de estas tiene un número de documentos asociados. El grado de relevancia de los documentos con respecto a las consultas es dado por humanos que los agrupan en tres niveles: definitivamente relevante “2”, parcialmente relevante “1” o no relevante “0”. Existen un total de 16 140 pares de consulta-documento etiquetados con juicios de relevancia, teniendo en cuenta 45 rasgos.

3.2.2 LETOR GOV, TREC 2007 y TREC 2008

[3] Special Interest Group on Information Retrieval, www.sigir.org

En TREC[4] 2003 y 2004, hay una línea peculiar de investigación para la recuperación de información en la Web, denominada la huella de la Web. La meta de esta línea es estudiar el comportamiento de la recuperación cuando la colección a ser investigada está en una estructura de grandes hipervínculos como Internet. Las huellas de la Web usan la colección Gov, que es basada en una selección del dominio ".gov" realizada en enero de 2002. Hay en total 1 053 110 documentos html en esta colección, junto con 11 164 829 hipervínculos. [17]

La destilación del tema es una tarea que permite encontrar puntos de entrada a aquellos sitios Web que están principalmente dedicados al tema de la consulta. El objetivo es devolver la página principal del sitio en lugar de las páginas que contengan alguna información, teniendo en cuenta que las páginas iniciales proporcionan una mejor apreciación global del contenido del sitio. Los asesores humanos utilizan juicios binarios para la clasificación. Una página es juzgada relevante solo si es la página de entrada de algún sitio que está principalmente dedicado al tema de la consulta.

Dentro de LETOR Gov se encuentran los conjuntos TD2003 (*Top Distillation* 2003), TD2004 (*Top Distillation* 2004), NP2003 (*Named Page Finding* 2003), NP2004 (*Named Page Finding* 2004), HP2003 (*Homepage Finding* 2003) y HP2004 (*Homepage Finding* 2004). En esta colección se utilizan un total de 575 consultas y son considerados 64 rasgos para representar la correspondencia de cada par consulta documento.

Por su parte LETOR 4.0 contiene 8 conjuntos de datos como referencia para diversas tareas de *ranking*. Este usa la colección de páginas web Gov2 y dos conjuntos de consultas a partir de Million Query track pertenecientes a TREC 2007 y TREC 2008. Estas últimas se utilizan en la presente investigación y son denominadas de forma reducida MQ2007 y MQ2008. MQ2007 tiene aproximadamente 1700 consultas y MQ2008 alrededor de 800 consultas, todas con sus respectivos documentos etiquetados.

[4] Text REtrieval Conference, trec.nist.gov

3.3 Medidas de evaluación

Las principales medidas de evaluación empleadas en R.I., se dividen en dos categorías que son concebidas de acuerdo a la relevancia:

- ✓ Relevancia binaria
  - Precisión en *n* (P@n: *Precision at n*)
  - Precisión media promedio (MAP: *Mean Average Precision*)
- ✓ Múltiples niveles de relevancia
  - Ganancia cumulativa descontada normalizada (NDCG: *Normalized Discounted Cumulative Gain*)

3.3.1 P@n

La precisión en *n* mide la relevancia de los documentos que se encuentran por encima de *n* (incluyendo éste) como resultado del ordenamiento con respecto a una consulta dada:

$$P@n = \frac{\#\text{ de doc relevantes en los primeros } \boldsymbol{n} \text{ resultados}}{\boldsymbol{n}} \tag{3.1}$$

Por ejemplo, si los 10 primeros documentos retornados a partir de una consulta son (relevante, irrelevante, irrelevante, relevante, relevante, relevante, irrelevante, irrelevante, relevante, relevante), entonces de P@1 a P@10 los valores serán {1, 1/2, 1/3, 2/4, 3/5, 4/6, 4/7, 4/8, 5/9, 6/10} respectivamente. [17]

3.3.2 MAP

Para una única consulta, la precisión media se define como el promedio de los valores de P@n para todos los documentos relevantes:

$$AP = \frac{\sum_{n=1}^{N}(P@n \times rel(n))}{\#\ total\ de\ docs\ relevantes\ para\ esta\ consulta} \tag{3.2}$$

donde N es el número de documentos recuperados, y $rel(n)$ es una función binaria que retorna la relevancia del *n*-ésimo documento:

$$rel\ (n) = \begin{matrix} 1 \\ 0 \end{matrix} \begin{cases} : el\ n-ésimo\ documento\ es\ relevante \\ : el\ n-ésimo\ documento\ no\ es\ relevante \end{cases} \tag{3.3}$$

El cálculo de AP para el ejemplo citado en la explicación de P@n sería:

$$AP = \frac{1\times 1+\frac{1}{2}\times 0+\frac{1}{3}\times 0+\frac{2}{4}\times 1+\frac{3}{5}\times 1+\frac{4}{6}\times 1+\frac{4}{7}\times 0+\frac{4}{8}\times 0+\frac{5}{9}\times 1+\frac{6}{10}\times 1}{6}$$

$$AP = \frac{1+\frac{2}{4}+\frac{3}{5}+\frac{4}{6}+\frac{5}{9}+\frac{6}{10}}{6} = 0.6533$$

Dado un conjunto de consultas, el valor de MAP, será el promedio de los valores de AP para cada una de las consultas.

3.3.3 NDCG

Las medidas de evaluación P@n y MAP pueden manejar solamente casos con juicios de relevancia binarios: relevante o irrelevante. Para los casos en los que se quiere manipular múltiples niveles de juicios de relevancia surge en el año 2002 NDCG [18], que sigue dos reglas fundamentales:

1. Los documentos altamente relevantes son más valiosos que los documentos marginalmente relevantes.
2. Un documento (de cualquier nivel de relevancia) de baja posición en el ordenamiento, tiene menos valor para el usuario, porque su probabilidad de ser examinado por dicho usuario es mucho menor.

De acuerdo a las reglas anteriores, el valor NDCG de una lista de ordenamiento en la posición *n* es calculado como sigue:

$$N(n) = Z_n \sum_{j=1}^{n} \frac{2^{r(j)}-1}{\log(1+j)} \qquad (3.4)$$

donde $r(j)$ es el valor de relevancia del *j*-ésimo documento en la lista de ordenamiento, y la constante de normalización Z_n es escogida para que la lista ideal obtenga un valor NDCG de 1. [17]

3.4 Comparación con métodos referenciados

Para la evaluación del desempeño del método propuesto y su comparación con los principales trabajos realizados en este campo investigativo; se siguieron las pautas experimentales publicadas en LETOR.

Se utilizó en el proceso de aprendizaje o entrenamiento para RankFSP una validación cruzada con cinco particiones (*five-fold cross validation*), lo cual posibilita realizar comparaciones directas en términos de precisión con los métodos publicados para el Aprendizaje de la Ordenación.

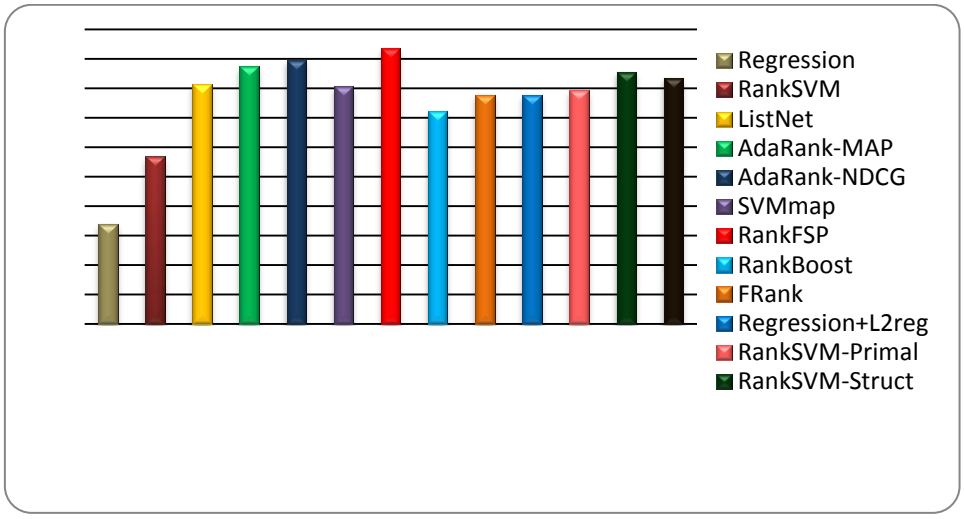
Los valores de los parámetros que forman parte de los datos de entrada del método y que fueron definidos en la sección 2.3.2 (Algoritmo), se fijaron considerando como referencia las propuestas realizadas en [42], después de un estudio completo del comportamiento del método para diferentes tipos de funciones estándar (De Jong's functions test), utilizadas en tareas de optimización.

Para los puntos de captura (N) se fijaron valores entre 80 y 100. Se consideraron 100 iteraciones (T) y de 3 a 5 lanzamientos (L) para todos los casos. La cantidad de vectores de posición de la maya (M) fueron ajustados empíricamente, comenzando por el valor 80. Por otra parte, se corroboró que el coeficiente de amplitud (c) para los conjuntos de datos utilizados potencia al algoritmo si su valor es fijado en 5.0 y 10.0, dependiendo del conjunto de datos utilizado.

En todos los experimentos llevados a cabo con RankFSP, se consideró a MAP como función *E* en la expresión (2.4).

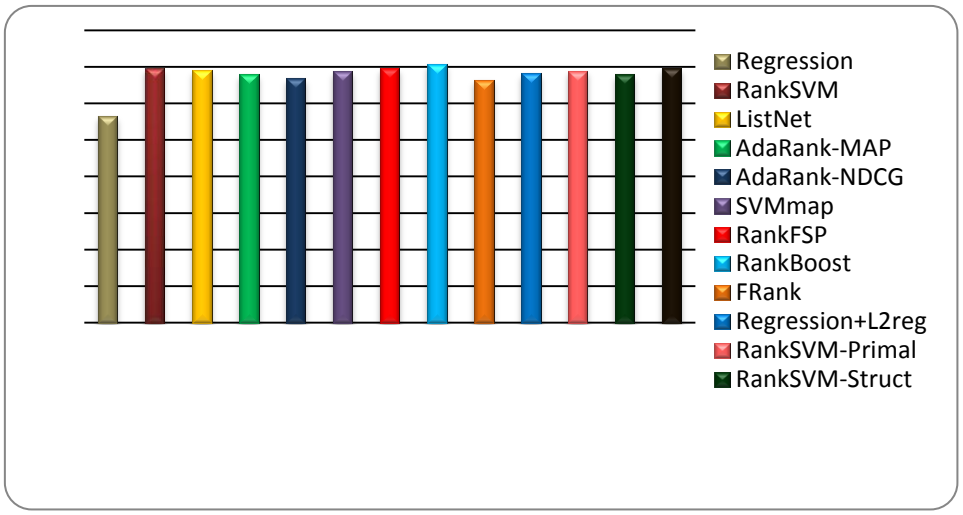
3.4.1 Desempeños obtenidos

En las Figuras 3.1 a 3.5 se muestran los resultados del rendimiento alcanzado en la ordenación sobre los conjuntos de datos seleccionados para el estudio, por cada algoritmo comparado; considerando la medida de desempeño MAP. Al lado de cada uno de los gráficos pueden apreciarse los valores obtenidos por cada algoritmo lo que confirma a primera vista el buen desempeño y estabilidad de RankFSP.



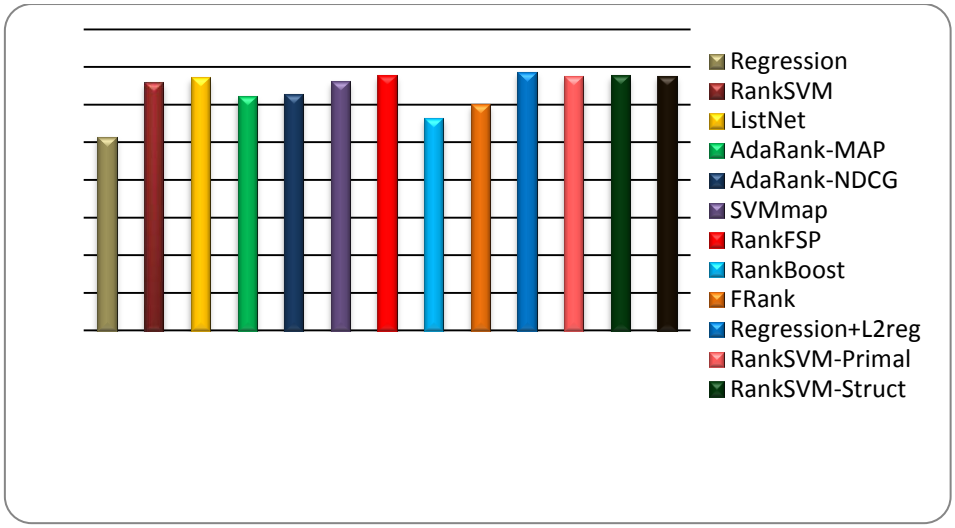
MAP
(-) 0,422
(-) 0,4334
(-) 0,4457
(-) 0,4487
(-) 0,4498
(-) 0,4453
0,4518 (1)
(-) 0,4411
(-) 0,4439
(-) 0,4439
(-) 0,4446
(-) 0,4478
(-) 0,4468

Figura 3.1: Rendimiento alcanzado en la ordenación sobre OHSUMED, considerando MAP como medida de desempeño.



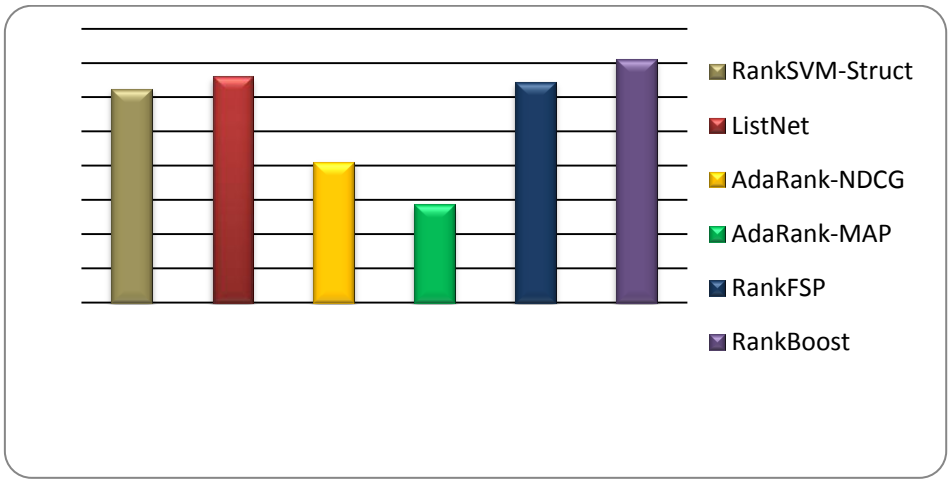
MAP
(-) 0,5644
(+) 0,6957
(-) 0,6895
(-) 0,6783
(-) 0,6678
(-) 0,6869
0,6955 (3)
(+) 0,7074
(-) 0,664
(-) 0,6824
(-) 0,6883
(-) 0,6788
(=) 0,6958

Figura 3.2: Rendimiento alcanzado en la ordenación sobre NP2003, considerando MAP como medida de desempeño.



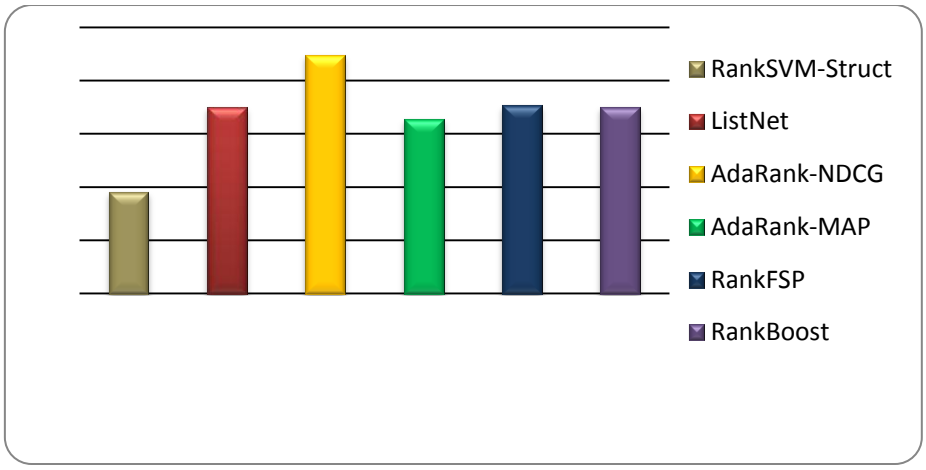
MAP
(-) 0,5142
(-) 0,6588
(-) 0,672
(-) 0,622
(-) 0,6269
(-) 0,662
0,6772 (2)
(-) 0,564
(-) 0,6008
(+) 0,6866
(-) 0,6755
(-) 0,6771
(-) 0,676

Figura 3.3: Rendimiento alcanzado en la ordenación sobre NP2004, considerando MAP como medida de desempeño.



MAP
(-) 0,4645
(+) 0,4652
(-) 0,4602
(-) 0,4577
0,4649 (3)
(+) 0,4662

Figura 3.4: Rendimiento alcanzado en la ordenación sobre MQ2007, considerando MAP como medida de desempeño.



MAP
(-) 0,4696
(-) 0,4775
(+) 0,4824
(-) 0,4764
0,4777 (2)
(-) 0,4775

Figura 3.5: Rendimiento alcanzado en la ordenación sobre MQ2008, considerando MAP como medida de desempeño.

Con los mejores vectores de pesca (posición) obtenidos en el aprendizaje empleando la medida de desempeño MAP como función *E*, se realiza la evaluación del método RankFSP para las otras medidas de desempeño de la R.I.

En las tablas 3.1 a 3.5 pueden observarse los resultados obtenidos para P@n. Para facilitar el análisis de estas tablas, se coloca el signo (-) a aquellos métodos que no superan el resultado de RankFSP, el signo (=) a los que tienen igual valor al método propuesto y el signo (+) a los que están por encima de dicho valor.

Al analizar estas tablas puede apreciarse cuan competitivos suelen ser los valores de rendimiento (precisión) obtenidos en las diez primeras posiciones de la ordenación por el método propuesto, en relación a los algoritmos publicados en LETOR.

Tabla 3.1: Valores del rendimiento alcanzado en la ordenación sobre OHSUMED, considerando P@n como medida de desempeño en las posiciones de la 1 a la 10.

(a)

Algoritmos	P@1	P@2	P@3	P@4	P@5
Regression	(-) 0,5965	(-) 0,6006	(-) 0,5768	(-) 0,5605	(-) 0,5337
RankSVM	(-) 0,5974	(-) 0,5494	(-) 0,5427	(-) 0,5443	(-) 0,5319
ListNet	(-) 0,6524	(-) 0,6093	(+) 0,6016	(-) 0,5745	(-) 0,5502
AdaRank-MAP	(-) 0,6338	(-) 0,5959	(-) 0,5895	(-) 0,5887	(-) 0,5674
AdaRank-NDCG	(+) 0,6719	(+) 0,6236	(+) 0,5984	(-) 0,5838	(-) 0,5767

SVMmap	(-) 0,6433	(+) 0,6197	(-) 0,5802	(-) 0,5768	(-) 0,5523
RankFSP	0,6718	0,6193	0,5926	0,5930	0,5786
RankBoost	(-) 0,5576	(-) 0,5481	(-) 0,5609	(-) 0,558	(-) 0,5447
FRank	(-) 0,6429	(+) 0,6195	(-) 0,5925	(-) 0,584	(-) 0,5638
Regression+L2reg	(-) 0,6437	(-) 0,6102	(-) 0,5896	(-) 0,5675	(-) 0,5505
RankSVM-Primal	(-) 0,6156	(-) 0,6052	(-) 0,5866	(-) 0,5773	(-) 0,5639
RankSVM-Struct	(-) 0,6338	(-) 0,6097	(-) 0,5898	(-) 0,5915	(-) 0,5696
SmoothRank	(=) 0,671	(+) 0,6377	(+) 0,6172	(-) 0,586	(-) 0,5711

(b)

Algoritmos	**P@6**	**P@7**	**P@8**	**P@9**	**P@10**
Regression	(-) 0,5045	(-) 0,5001	(-) 0,4837	(-) 0,4752	(-) 0,4666
RankSVM	(-) 0,5253	(-) 0,5097	(-) 0,4933	(-) 0,492	(-) 0,4864
ListNet	(-) 0,5373	(-) 0,5267	(-) 0,5236	(-) 0,5138	(-) 0,4975
AdaRank-MAP	(-) 0,5566	(-) 0,5392	(-) 0,5239	(-) 0,5077	(-) 0,4976
AdaRank-NDCG	(-) 0,5563	(+) 0,5511	(+) 0,5354	(=) 0,5211	(+) 0,5087
SVMmap	(-) 0,5281	(-) 0,5108	(-) 0,506	(-) 0,4928	(-) 0,491
RankFSP	0,5577	0,5470	0,5330	0,5211	0,5058
RankBoost	(-) 0,5297	(-) 0,5241	(-) 0,513	(-) 0,5024	(-) 0,4966
FRank	(-) 0,5517	(-) 0,5445	(-) 0,5251	(-) 0,5152	(-) 0,5016
Regression+L2reg	(-) 0,5423	(-) 0,5189	(-) 0,5144	(-) 0,5056	(-) 0,5014
RankSVM-Primal	(-) 0,5364	(-) 0,5286	(-) 0,5192	(-) 0,5087	(+) 0,5071
RankSVM-Struct	(-) 0,5426	(-) 0,5325	(-) 0,5249	(-) 0,5203	(+) 0,5071
SmoothRank	(-) 0,5531	(-) 0,5376	(-) 0,5261	(-) 0,5109	(+) 0,506

Tabla 3.2: Valores del rendimiento alcanzado en la ordenación sobre NP2003, considerando P@n como medida de desempeño en las posiciones de la 1 a la 10.

(a)

Algoritmos	**P@1**	**P@2**	**P@3**	**P@4**	**P@5**
Regression	(-) 0,4467	(-) 0,2833	(-) 0,22	(-) 0,1767	(-) 0,1453
RankSVM	(=) 0,58	(+) 0,37	(+) 0,2711	(+) 0,2083	(+) 0,1707
ListNet	(-) 0,5667	(+) 0,37	(-) 0,2667	(+) 0,2083	(+) 0,172
AdaRank-MAP	(=) 0,58	(-) 0,3567	(-) 0,2511	(-) 0,1917	(-) 0,16
AdaRank-NDCG	(-) 0,56	(-) 0,3467	(-) 0,2467	(-) 0,1967	(-) 0,1613
SVMmap	(-) 0,56	(+) 0,38	(-) 0,2689	(+) 0,21	(+) 0,1707
RankFSP	0,58	0,3659	0,2702	0,2074	0,1703
RankBoost	(+) 0,6	(+) 0,3767	(-) 0,2689	(-) 0,205	(-) 0,1693
FRank	(-) 0,54	(-) 0,3533	(-) 0,2533	(-) 0,2033	(-) 0,168
Regression+L2reg	(-) 0,5467	(+) 0,37	(+) 0,2733	(+) 0,2117	(+) 0,172
RankSVM-Primal	(-) 0,5733	(=) 0,36	(+) 0,2733	(+) 0,2117	(-) 0,1693
RankSVM-Struct	(-) 0,5533	(+) 0,3667	(-) 0,2622	(+) 0,2083	(+) 0,1707
SmoothRank	(=) 0,58	(+) 0,3733	(-) 0,26	(+) 0,2083	(+) 0,172

(b)

Algoritmos	**P@6**	**P@7**	**P@8**	**P@9**	**P@10**

Regression	(-) 0,1256	(-) 0,1105	(-) 0,0983	(-) 0,0889	(-) 0,0813
RankSVM	(=) 0,1433	(+) 0,1267	(+) 0,1117	(+) 0,1	(+) 0,092
ListNet	(+) 0,1456	(+) 0,1295	(+) 0,1133	(+) 0,1015	(+) 0,0927
AdaRank-MAP	(-) 0,1367	(-) 0,1181	(-) 0,105	(-) 0,0941	(-) 0,0867
AdaRank-NDCG	(=) 0,14	(-) 0,1229	(-) 0,1083	(=) 0,0985	(-) 0,0893
SVMmap	(=) 0,1433	(-) 0,1257	(=) 0,11	(=) 0,0985	(-) 0,0893
RankFSP	0,1433	0,1265	0,1109	0,0985	0,091
RankBoost	(=) 0,1433	(+) 0,1276	(+) 0,1125	(+) 0,103	(+) 0,094
FRank	(=) 0,1433	(-) 0,1238	(=) 0,11	(-) 0,0978	(-) 0,0907
Regression+L2reg	(+) 0,1456	(+) 0,1267	(+) 0,1117	(+) 0,1	(+) 0,0913
RankSVM-Primal	(-) 0,1422	(-) 0,1257	(=) 0,11	(+) 0,0993	(-) 0,0907
RankSVM-Struct	(=) 0,1433	(-) 0,1257	(+) 0,1117	(+) 0,1	(+) 0,092
SmoothRank	(+) 0,1456	(-) 0,1257	(=) 0,11	(+) 0,1	(=) 0,09

Tabla 3.3: Valores del rendimiento alcanzado en la ordenación sobre NP2004, considerando P@n como medida de desempeño en las posiciones de la 1 a la 10.

(a)

Algoritmos	**P@1**	**P@2**	**P@3**	**P@4**	**P@5**
Regression	(-) 0,3733	(-) 0,2667	(=) 0,2	(-) 0,1733	(-) 0,144
RankSVM	(-) 0,5067	(+) 0,3733	(-) 0,2622	(+) 0,22	(+) 0,1787
ListNet	(-) 0,5333	(+) 0,3733	(+) 0,2667	(-) 0,2167	(+) 0,1787
AdaRank-MAP	(-) 0,48	(-) 0,3333	(-) 0,2444	(-) 0,1933	(-) 0,1627
AdaRank-NDCG	(-) 0,5067	(-) 0,32	(-) 0,2489	(-) 0,2067	(-) 0,1653
SVMmap	(-) 0,52	(=) 0,36	(+) 0,2667	(+) 0,22	(+) 0,1787
RankFSP	0,56	0,3601	0,2658	0,2185	0,1782
RankBoost	(-) 0,4267	(-) 0,2867	(-) 0,2311	(-) 0,1833	(-) 0,152
FRank	(-) 0,48	(-) 0,3133	(-) 0,2356	(-) 0,19	(-) 0,16
Regression+L2reg	(+) 0,5733	(=) 0,36	(-) 0,2622	(-) 0,2167	(-) 0,176
RankSVM-Primal	(=) 0,56	(=) 0,36	(-) 0,2533	(-) 0,2133	(-) 0,1733
RankSVM-Struct	(=) 0,56	(=) 0,36	(-) 0,2578	(-) 0,2167	(-) 0,1733
SmoothRank	(-) 0,5467	(=) 0,36	(+) 0,2667	(+) 0,22	(+) 0,1787

(b)

Algoritmos	**P@6**	**P@7**	**P@8**	**P@9**	**P@10**
Regression	(-) 0,1311	(-) 0,1181	(-) 0,1033	(-) 0,0919	(-) 0,0827
RankSVM	(-) 0,1489	(-) 0,1295	(-) 0,1133	(-) 0,1022	(-) 0,0933
ListNet	(+) 0,1533	(+) 0,1333	(+) 0,1183	(-) 0,1052	(-) 0,0947
AdaRank-MAP	(-) 0,14	(-) 0,1219	(-) 0,11	(-) 0,0978	(-) 0,088
AdaRank-NDCG	(-) 0,1422	(-) 0,1238	(-) 0,1133	(-) 0,1007	(-) 0,0907
SVMmap	(+) 0,1533	(+) 0,1333	(+) 0,1183	(-) 0,1052	(+) 0,096
RankFSP	0,1512	0,1321	0,1171	0,1059	0,095
RankBoost	(-) 0,1333	(-) 0,12	(-) 0,1067	(-) 0,0948	(-) 0,088
FRank	(-) 0,14	(-) 0,1238	(=) 0,11	(-) 0,1007	(-) 0,0933
Regression+L2reg	(+) 0,1533	(+) 0,1352	(+) 0,1183	(+) 0,1067	(+) 0,096
RankSVM-Primal	(-) 0,1511	(-) 0,1314	(-) 0,1167	(-) 0,1052	(-) 0,0947

RankSVM-Struct	(-) 0,1511	(-) 0,1314	(-) 0,1167	(-) 0,1052	(-) 0,0947
SmoothRank	(+) 0,1556	(+) 0,1371	(+) 0,12	(+) 0,1067	(+) 0,096

Tabla 3.4: Valores del rendimiento alcanzado en la ordenación sobre MQ2007, considerando P@n como medida de desempeño en las posiciones de la 1 a la 10.

(a)

Algoritmos	P@1	P@2	P@3	P@4	P@5
RankSVM-Struct	(+) 0,47464	(+) 0,44956	(-) 0,4315	(-) 0,41936	(-) 0,4135
ListNet	(+) 0,464	(+) 0,4471	(+) 0,4334	(+) 0,4248	(-) 0,4126
AdaRank-NDCG	(-) 0,4475	(-) 0,4374	(-) 0,4305	(-) 0,4169	(-) 0,4068
AdaRank-MAP	(-) 0,4392	(-) 0,4301	(-) 0,423	(-) 0,4126	(-) 0,4054
RankFSP	0,4601	0,4451	0,4331	0,4235	0,4121
RankBoost	(+) 0,4823	(+) 0,4542	(+) 0,4348	(+) 0,4242	(+) 0,4185

(b)

Algoritmos	P@6	P@7	P@8	P@9	P@10
RankSVM-Struct	(+) 0,40482	(+) 0,3994	(+)0,39314	(+)0,38684	(+)0,38332
ListNet	(=) 0,4036	(+) 0,3968	(+) 0,3911	(+) 0,3847	(+) 0,3798
AdaRank-NDCG	(-) 0,3994	(-) 0,3925	(-) 0,3868	(-) 0,3826	(-) 0,3756
AdaRank-MAP	(-) 0,3953	(-) 0,3873	(-) 0,3823	(-) 0,3782	(-) 0,3738
RankFSP	0,403	0,3958	0,3899	0,3832	0,3788
RankBoost	(+) 0,4094	(+) 0,4016	(+) 0,3947	(+) 0,3901	(+) 0,3862

Tabla 3.5: Valores del rendimiento alcanzado en la ordenación sobre MQ2008, considerando P@n como medida de desempeño en las posiciones de la 1 a la 10.

(a)

Algoritmos	P@1	P@2	P@3	P@4	P@5
RankSVM-Struct	(-) 0,4273	(-) 0,40686	(-) 0,39032	(+)0,36956	(+)0,34744
ListNet	(-) 0,4451	(+) 0,412	(-) 0,3835	(+) 0,3651	(+) 0,3426
AdaRank-NDCG	(-) 0,4515	(+) 0,4222	(+) 0,395	(+) 0,3696	(+) 0,3454
AdaRank-MAP	(-) 0,4426	(+) 0,4165	(-) 0,3899	(+) 0,3677	(+) 0,3449
RankFSP	0,458	0,4116	0,3918	0,3645	0,341
RankBoost	(-) 0,4579	(-) 0,4114	(-) 0,3916	(-) 0,3642	(-) 0,3403

(b)

Algoritmos	P@6	P@7	P@8	P@9	P@10
RankSVM-Struct	(+) 0,32652	(-) 0,30212	(-) 0,2822	(-) 0,26474	(-) 0,2491
ListNet	(-) 0,3204	(-) 0,3006	(-) 0,2791	(-) 0,2627	(-) 0,2476
AdaRank-NDCG	(+) 0,3227	(-) 0,2993	(-) 0,2795	(-) 0,2618	(-) 0,2454
AdaRank-MAP	(+) 0,3216	(-) 0,2992	(-) 0,2796	(-) 0,262	(-) 0,2454
RankFSP	0,3213	0,303	0,2842	0,2655	0,2492
RankBoost	(=) 0,321	(-) 0,3021	(+) 0,2846	(-) 0,2652	(-) 0,2487

Al igual que se hizo para la medida de desempeño P@n, en las tablas 3.6 a 3.10 se muestran los resultados de la evaluación de RankFSP para NDCG.

Tabla 3.6: Valores del rendimiento alcanzado en la ordenación sobre OHSUMED, considerando NDCG como medida de desempeño en las posiciones de la 1 a la 10.

(a)

Algoritmos	NDCG@1	NDCG@2	NDCG@3	NDCG@4	NDCG@5
Regression	(-) 0,4456	(-) 0,4532	(-) 0,4426	(-) 0,4368	(-) 0,4278
RankSVM	(-) 0,4958	(-) 0,4331	(-) 0,4207	(-) 0,424	(-) 0,4164
ListNet	(-) 0,5326	(-) 0,481	(-) 0,4732	(-) 0,4561	(-) 0,4432
AdaRank-MAP	(-) 0,5388	(-) 0,4789	(-) 0,4682	(-) 0,4721	(-) 0,4613
AdaRank-NDCG	(-) 0,533	(-) 0,4922	(-) 0,479	(-) 0,4688	(-) 0,4673
SVMmap	(-) 0,5229	(-) 0,4909	(-) 0,4663	(-) 0,4625	(-) 0,4516
RankFSP	0,5584	0,4940	0,4824	0,4818	0,4784
RankBoost	(-) 0,4632	(-) 0,4504	(-) 0,4555	(-) 0,4543	(-) 0,4494
FRank	(-) 0,53	(+) 0,5008	(-) 0,4812	(-) 0,4694	(-) 0,4588
Regression+L2reg	(-) 0,5361	(-) 0,4784	(-) 0,474	(-) 0,4601	(-) 0,4568
RankSVM-Primal	(-) 0,546	(+) 0,501	(+) 0,4855	(-) 0,4766	(-) 0,4689
RankSVM-Struct	(-) 0,5515	(+) 0,5	(+) 0,485	(+) 0,482	(-) 0,4729
SmoothRank	(-) 0,5576	(+) 0,5149	(+) 0,4964	(+) 0,485	(-) 0,4776

(b)

Algoritmos	NDCG@6	NDCG@7	NDCG@8	NDCG@9	NDCG@10
Regression	(-) 0,4216	(-) 0,4217	(-) 0,4187	(-) 0,4136	(-) 0,411
RankSVM	(-) 0,4159	(-) 0,4133	(-) 0,4072	(-) 0,4124	(-) 0,414
ListNet	(-) 0,44	(-) 0,4409	(-) 0,446	(-) 0,4459	(-) 0,441
AdaRank-MAP	(-) 0,4579	(-) 0,4558	(-) 0,4506	(-) 0,4464	(-) 0,4429
AdaRank-NDCG	(-) 0,4597	(-) 0,4596	(-) 0,4575	(-) 0,4541	(-) 0,4496
SVMmap	(-) 0,4411	(-) 0,4335	(-) 0,4332	(-) 0,4304	(-) 0,4319
RankFSP	0,4707	0,4675	0,4607	0,4586	0,4539
RankBoost	(-) 0,4439	(-) 0,4412	(-) 0,4361	(-) 0,4326	(-) 0,4302
FRank	(-) 0,455	(-) 0,4529	(-) 0,4476	(-) 0,446	(-) 0,4433
Regression+L2reg	(-) 0,4536	(-) 0,4449	(-) 0,4444	(-) 0,4427	(-) 0,4436
RankSVM-Primal	(-) 0,4552	(-) 0,4534	(-) 0,45	(-) 0,449	(-) 0,4504
RankSVM-Struct	(-) 0,4584	(-) 0,457	(-) 0,4587	(-) 0,4568	(-) 0,4523
SmoothRank	(-) 0,4702	(-) 0,4667	(+) 0,4608	(-) 0,4555	(+) 0,4568

Tabla 3.7: Valores del rendimiento alcanzado en la ordenación sobre NP2003, considerando NDCG como medida de desempeño en las posiciones de la 1 a la 10.

(a)

Algoritmos	NDCG@1	NDCG@2	NDCG@3	NDCG@4	NDCG@5
Regression	(-) 0,4467	(-) 0,5567	(-) 0,6135	(-) 0,6351	(-) 0,6423
RankSVM	(=) 0,58	(+) 0,7233	(+) 0,7654	(+) 0,7737	(=) 0,7823
ListNet	(-) 0,5667	(=) 0,72	(-) 0,7579	(+) 0,7729	(+) 0,7843
AdaRank-MAP	(=) 0,58	(-) 0,7033	(-) 0,7286	(-) 0,7352	(-) 0,7482
AdaRank-NDCG	(-) 0,56	(-) 0,6867	(-) 0,7161	(-) 0,7361	(-) 0,7447
SVMmap	(-) 0,56	(+) 0,74	(+) 0,7673	(+) 0,7823	(+) 0,7881
RankFSP	0,58	0,7229	0,7644	0,7720	0,7823
RankBoost	(+) 0,6	(+) 0,73	(-) 0,7636	(-) 0,7703	(-) 0,7818

FRank	(-) 0,54	(-) 0,6967	(-) 0,7261	(-) 0,7494	(-) 0,7595
Regression+L2reg	(-) 0,5467	(+) 0,7267	(+) 0,7729	(+) 0,7829	(+) 0,7887
RankSVM-Primal	(-) 0,5733	(=) 0,7	(-) 0,7631	(+) 0,7748	(-) 0,7748
RankSVM-Struct	(-) 0,5533	(-) 0,7167	(-) 0,7503	(-) 0,7703	(-) 0,7789
SmoothRank	(=) 0,58	(+) 0,7333	(-) 0,7544	(+) 0,7777	(+) 0,7892

(b)

Algoritmos	NDCG@6	NDCG@7	NDCG@8	NDCG@9	NDCG@10
Regression	(-) 0,6526	(-) 0,6597	(-) 0,6597	(-) 0,6629	(-) 0,6659
RankSVM	(+) 0,7849	(+) 0,7944	(+) 0,7922	(+) 0,7943	(+) 0,8003
ListNet	(+) 0,7882	(+) 0,8001	(+) 0,7956	(+) 0,7977	(+) 0,8018
AdaRank-MAP	(-) 0,7546	(-) 0,757	(-) 0,757	(-) 0,7591	(-) 0,7641
AdaRank-NDCG	(-) 0,7563	(-) 0,7623	(-) 0,7589	(-) 0,7652	(-) 0,7672
SVMmap	(+) 0,7907	(+) 0,7978	(+) 0,7933	(+) 0,7954	(-) 0,7975
RankFSP	0,7848	0,7940	0,7915	0,7929	0,8002
RankBoost	(+) 0,787	(+) 0,7976	(+) 0,7954	(+) 0,8028	(+) 0,8068
FRank	(-) 0,7659	(-) 0,7683	(-) 0,7683	(-) 0,7683	(-) 0,7763
Regression+L2reg	(+) 0,7938	(+) 0,7986	(+) 0,7964	(+) 0,7985	(+) 0,8025
RankSVM-Primal	(-) 0,7773	(-) 0,7857	(-) 0,7812	(-) 0,7854	(-) 0,7894
RankSVM-Struct	(-) 0,7802	(-) 0,7873	(-) 0,7873	(-) 0,7894	(-) 0,7955
SmoothRank	(+) 0,7943	(+) 0,7967	(+) 0,7923	(+) 0,7986	(-) 0,7986

Tabla 3.8: Valores del rendimiento alcanzado en la ordenación sobre NP2004, considerando NDCG como medida de desempeño en las posiciones de la 1 a la 10.

(a)

Algoritmos	NDCG@1	NDCG@2	NDCG@3	NDCG@4	NDCG@5
Regression	(-) 0,3733	(-) 0,5133	(-) 0,5554	(-) 0,6021	(-) 0,6135
RankSVM	(-) 0,5067	(+) 0,7267	(+) 0,7503	(+) 0,7928	(+) 0,7957
ListNet	(-) 0,5333	(+) 0,7267	(+) 0,7587	(+) 0,7879	(+) 0,7965
AdaRank-MAP	(-) 0,48	(-) 0,66	(-) 0,6979	(-) 0,7137	(-) 0,731
AdaRank-NDCG	(-) 0,5067	(-) 0,6133	(-) 0,6722	(-) 0,7122	(-) 0,7122
SVMmap	(-) 0,52	(=) 0,7	(+) 0,7489	(+) 0,7847	(+) 0,7869
RankFSP	0,57	0,7	0,7325	0,7745	0,7767
RankBoost	(-) 0,4267	(-) 0,5533	(-) 0,6274	(-) 0,6433	(-) 0,6512
FRank	(-) 0,48	(-) 0,6	(-) 0,6431	(-) 0,6697	(-) 0,687
Regression+L2reg	(+) 0,5733	(=) 0,7	(+) 0,7352	(+) 0,7752	(+) 0,7774
RankSVM-Primal	(-) 0,56	(=) 0,7	(-) 0,7236	(-) 0,7662	(-) 0,7719
RankSVM-Struct	(-) 0,56	(=) 0,7	(-) 0,7321	(+) 0,7746	(-) 0,7746
SmoothRank	(-) 0,5467	(=) 0,7	(+) 0,7437	(+) 0,7803	(+) 0,7825

(b)

Algoritmos	NDCG@6	NDCG@7	NDCG@8	NDCG@9	NDCG@10
Regression	(-) 0,6393	(-) 0,6536	(-) 0,6536	(-) 0,6536	(-) 0,6536
RankSVM	(+) 0,7957	(+) 0,8005	(+) 0,8005	(+) 0,8047	(+) 0,8062
ListNet	(+) 0,8037	(+) 0,8084	(+) 0,8128	(+) 0,8128	(+) 0,8128
AdaRank-MAP	(-) 0,7413	(-) 0,7431	(-) 0,7497	(-) 0,7497	(-) 0,7497

AdaRank-NDCG	(-) 0,7225	(-) 0,7273	(-) 0,7384	(-) 0,7384	(-) 0,7384
SVMmap	(+) 0,7947	(+) 0,7994	(+) 0,8039	(+) 0,8039	(+) 0,8079
RankFSP	0,7888	0,7895	0,7940	0,7980	0,7980
RankBoost	(-) 0,6667	(-) 0,681	(-) 0,6854	(-) 0,6854	(-) 0,6914
FRank	(-) 0,6992	(-) 0,7087	(-) 0,7132	(-) 0,7216	(-) 0,7296
Regression+L2reg	(+) 0,7903	(+) 0,7998	(+) 0,7998	(+) 0,804	(+) 0,804
RankSVM-Primal	(-) 0,7816	(-) 0,7864	(-) 0,7908	(-) 0,795	(-) 0,795
RankSVM-Struct	(-) 0,7843	(-) 0,789	(-) 0,7935	(-) 0,7977	(-) 0,7977
SmoothRank	(+) 0,798	(+) 0,8075	(+) 0,8075	(+) 0,8075	(+) 0,8075

Tabla 3.9: Valores del rendimiento alcanzado en la ordenación sobre MQ2007, considerando NDCG como medida de desempeño en las posiciones de la 1 a la 10.

(a)

Algoritmos	NDCG@1	NDCG@2	NDCG@3	NDCG@4	NDCG@5
RankSVM-Struct	(+) 0,40962	(+)0,40736	(-) 0,40628	(-) 0,40844	(-) 0,41426
ListNet	(+) 0,4002	(+) 0,4063	(+) 0,4091	(+) 0,4144	(+) 0,417
AdaRank-NDCG	(-) 0,3876	(-) 0,3967	(-) 0,4044	(-) 0,4067	(-) 0,4102
AdaRank-MAP	(-) 0,3821	(-) 0,39	(-) 0,3984	(-) 0,4024	(-) 0,407
RankFSP	0,4001	0,4057	0,4085	0,4139	0,4163
RankBoost	(+) 0,4134	(+) 0,4094	(-) 0,4072	(-) 0,4122	(+) 0,4183

(b)

Algoritmos	NDCG@6	NDCG@7	NDCG@8	NDCG@9	NDCG@10
RankSVM-Struct	(-) 0,4195	(-) 0,42522	(-) 0,43064	(-) 0,4362	(+)0,44386
ListNet	(+) 0,4229	(+) 0,4275	(+) 0,4328	(+) 0,4381	(+) 0,444
AdaRank-NDCG	(-) 0,4156	(-) 0,4203	(-) 0,4258	(-) 0,4319	(-) 0,4369
AdaRank-MAP	(-) 0,4113	(-) 0,4154	(-) 0,4217	(-) 0,4278	(-) 0,4335
RankFSP	0,4221	0,4269	0,432	0,4372	0,4433
RankBoost	(+) 0,4227	(-) 0,4267	(+) 0,4323	(+) 0,4392	(+) 0,4464

Tabla 3.10: Valores del rendimiento alcanzado en la ordenación sobre MQ2008, considerando NDCG como medida de desempeño en las posiciones de la 1 a la 10.

(a)

Algoritmos	NDCG@1	NDCG@2	NDCG@3	NDCG@4	NDCG@5
RankSVM-Struct	(-) 0,36266	(-) 0,39848	(-) 0,42858	(+)0,45086	(+)0,46954
ListNet	(-) 0,3754	(+) 0,4112	(+) 0,4324	(+) 0,4568	(+) 0,4747
AdaRank-NDCG	(-) 0,3826	(+) 0,4211	(+) 0,442	(+) 0,4653	(+) 0,4821
AdaRank-MAP	(-) 0,3754	(+) 0,4141	(+) 0,437	(+) 0,4609	(+) 0,4794
RankFSP	0,3861	0,3995	0,4289	0,4482	0,467
RankBoost	(-) 0,3856	(-) 0,3993	(-) 0,4288	(-) 0,4479	(-) 0,4666

(b)

Algoritmos	NDCG@6	NDCG@7	NDCG@8	NDCG@9	NDCG@10
RankSVM-Struct	(+)0,48512	(+)0,49052	(+)0,45644	(+)0,22392	(-) 0,22792
ListNet	(+) 0,4894	(+) 0,4978	(+) 0,463	(+) 0,2265	(+) 0,2303
AdaRank-NDCG	(+) 0,4948	(+) 0,4993	(+) 0,4636	(+) 0,227	(+) 0,2307

AdaRank-MAP	(+) 0,4917	(+) 0,497	(+) 0,4609	(+) 0,2254	(+) 0,2288
RankFSP	0,4819	0,489	0,457	0,2216	0,2257
RankBoost	(-) 0,4816	(+) 0,4898	(-) 0,4568	(-) 0,2214	(-) 0,2255

3.4.2 Análisis estadístico

Después de haber obtenido los resultados del desempeño de cada algoritmo estudiado, se realizaron pruebas estadísticas para fundamentar con un basamento matemático el nivel de significación y comportamiento reflejados por la precisión alcanzada en la ordenación a nivel de consulta.

Para ello, primeramente se tomaron de cada algoritmo considerado, los cinco ficheros de puntuaciones (de extensión *.sco*) correspondientes a los cinco conjuntos de prueba prefijados por cada uno de los conjuntos de datos.

Luego, se utilizó una de las herramienta de evaluación propuesta por LETOR: Eval-Score-3.0.pl [45] para los conjuntos de datos de LETOR 3.0 con el argumento (*flag=1*) en cada fichero de puntuación. Para LETOR 4.0 se pudiese emplear Eval-Score-4.0.pl [46] pero en la Web oficial no se han publicado aún los ficheros score (.sco) para cada uno de los métodos referenciados.

Como resultado se obtuvo un nuevo fichero con el desempeño en cuanto a precisión alcanzada por cada algoritmo a nivel de consulta, en relación a la medida de evaluación MAP.

A continuación, fueron tomados los valores de precisión en la ordenación determinados por cada algoritmo según la medida de evaluación MAP.

Luego, se aplicaron pruebas no paramétricas usando un modelo estadístico de comparación de medias para muestras relacionadas o pareadas.

Este modelo fue aplicado porque se ajusta bien a la necesidad de comparar diferentes algoritmos en un mismo grupo de consultas; todo lo cual permitió llevar a cabo un estudio comparativo de los desempeños de cada algoritmo con un alto nivel de veracidad y exactitud.

Específicamente, se llevó a cabo una comparación de carácter horizontal se le denomina también análisis longitudinal, comparación de muestras dependientes o análisis de varianza de segunda vía (tow way ANOVA) [47].

Para este propósito, se empleó el test de Friedman [48]. Si el resultado de este test es significativo, entones significa que existen diferencias significativas entre al menos dos de los sets. Como resultado se puede concluir que existe una gran posibilidad de que al menos dos de las muestras presenten poblaciones con diferencias en los valores de las medias.

Para poder determinar cuál de los algoritmos presenta diferencias significativas, es necesario analizar si la distribución de los datos coincide en el sentido que no haya predominio de los incrementos ni de reducciones en la diferencia de los mismos. Para verificar esta hipótesis, Wilcoxon [49,50] propone calcular las diferencias por pares y ordenarlas en conjunto. Si la hipótesis fundamental es cierta, el número y magnitud de veces en que los resultados de un algoritmo son mayores que el de otro no debe diferir mucho del número y magnitud de veces que ocurre lo contrario y las diferencias ordenadas deben equilibrarse.

Para los casos en que se realizó una prueba de Friedman se utilizaron intervalos de confianza del 95%, mientras que en los casos en los cuales se realizó una prueba de Wilcoxon se exigió intervalos de confianza del 99%.

Específicamente, como variable dependiente, fue seleccionada la precisión a nivel de consulta obtenida por cada uno de los algoritmos sobre cada conjunto de datos (p.e. 150 valores por cada método para NP2003), considerando a MAP como medida de evaluación.

Una vez aplicado el Test de Friedman, en su interpretación se pudo constatar que el nivel crítico (Sig.) asociado a cada uno de estos algoritmos y para todos los conjuntos estudiados es menor que 0.05, por lo cual se procede a rechazar la hipótesis nula de igualdad de medias y se concluye que el desempeño en la ordenación (ranking) sobre OHSUMED, NP2003 y NP2004, considerando a MAP, no es la misma en los 8 algoritmos de aprendizaje.

Este resultado puede ser consultado en las tablas 3.11 a 3.16, además de otros estadísticos y las medias alcanzadas por cada algoritmo frente a cada conjunto de datos.

Tabla 3.11: Estadísticos del test de Friedman obtenidos en OHSUMED

Test Statistics[a]

N	106
Chi-Square	33,532
df	7
Asymp. Sig.	,000

a. Friedman Test

Tabla 3.12: Media alcanzada por cada algoritmo en OHSUMED

	Mean Rank
AdaRankMAP	4,74
AdaRankNDCG	5,02
FRank	4,19
ListNet	4,71
RankBoost	4,25
RankFSP	5,28
RankSVM	4,01
Regression	3,80

Tabla 3.13: Estadísticos del test de Friedman obtenidos en NP2003

Test Statistics[a]

N	150
Chi-Square	64,585
df	7
Asymp. Sig.	,000

a. Friedman Test

Tabla 3.14: Media alcanzada por cada algoritmo en NP2003

	Mean Rank
AdaRankMAP	4,45
AdaRankNDCG	4,45
FRank	4,41
ListNet	4,70
RankBoost	4,91
RankFSP	4,76
RankSVM	4,78
Regression	3,55

Tabla 3.15: Estadísticos del test de Friedman obtenidos en NP2004

Test Statistics[a]

N	75
Chi-Square	34,423
df	7
Asymp. Sig.	,000

a. Friedman Test

Tabla 3.16: Media alcanzada por cada algoritmo en NP2004

	Mean Rank
AdaRankMAP	4,49
AdaRankNDCG	4,69
FRank	4,55
ListNet	4,76
RankBoost	3,65
RankFSP	5,01
RankSVM	4,95
Regression	3,91

Dado este resultado, se procedió a aplicar el test de Wilcoxon para determinar entre cuales algoritmos y para cuales conjuntos de datos se manifestaban tales diferencias.

Para el caso de OHSUMED, se determinó que todos los métodos analizados presentan mejoras significativas con respecto a Regression menos RankSVM. Por su parte, RankFSP supera en términos de precisión a FRank, ListNet, RankBoost y a RankSVM; y AdaRankNDCG mejora a RankBoost y RankSVM. Además, RankSVM es superado por los métodos AdaRankMAP y ListNet.

En el caso del conjunto de datos NP2003, todos los algoritmos de aprendizaje de la ordenación mejoran significativamente el desempeño de Regression; se aprecia además que RankBoost supera los métodos AdaRankNDCG y FRank. Entre los demás algoritmos no se establecen diferencias significativas.

Finalmente en NP2004, tanto RankFSP, AdaRankMAP, AdaRankNDCG, ListNet, como RankSVM superan significativamente a RankBoost y Regression. Además FRank muestra mejores desempeños que RankBoost. Los demás métodos no presentan diferencias significativas entre sí.

No se pudo realizar un análisis estadístico para evaluar el desempeño de los métodos en MQ2007 y MQ2008, pues no se publican en LETOR las

puntuaciones alcanzadas por estos a nivel de consulta. Pero fueron comparados tanto gráficas, como tabularmente.

Todo el procesamiento estadístico fue realizado utilizando el paquete SPSS[5] (versión 20.0).

3.5 Discusión y trabajos futuros

Como trabajos futuros vinculados a la presente investigación están:

- ✓ Conducir más experimentos con nuevas y referenciadas colecciones de datos de mediano y gran tamaño para validar la eficiencia, y evaluar el desempeño del método RankFSP bajo disímiles condiciones.
- ✓ Utilizar otras medidas de desempeño, por ejemplo, P@n y NDCG; como función general *E*, con el objetivo de evaluar el comportamiento de la precisión en la ordenación, considerando el mismo diseño experimental conformado para MAP.
- ✓ Aplicar otras variantes de error en la función de pérdida básica, como por ejemplo, el Error Cuadrático Medio.
- ✓ Rediseñar la función de *ranking* considerando las relaciones y el comportamiento de los rasgos.

En sentido general, sería interesante poder diseñar modelos de Aprendizaje de la Ordenación que implementasen algoritmos de aprendizaje *online*, que se adapten a los cambios que ocurren en la Web a tiempo real.

Por otro lado, la tendencia en LETOR resulta en proponer nuevos conjuntos de entrenamiento de gran tamaño (p.e: MSLR-WEB10K[6]), cuya manipulación reviste un reto para muchos modelos de aprendizaje. En tal caso, sería oportuno utilizar elementos de la computación paralela, el aprendizaje de conjunto y llevar a cabo aproximaciones a algoritmos existentes que reduzcan el costo y mantengan la precisión.

Incorporar nuevas estrategias a los métodos de Aprendizaje de la Ordenación para evadir el sobre-ajuste y el estancamiento de las soluciones en mínimos

[5] Statistical Product and Service Solutions http://www.spss.com/
[6] Disponible en: http://research.microsoft.com/en-us/projects/mslr/

locales, también constituye una tarea en la que se debe enfocar la comunidad de investigadores.

Finalmente como pasó en su momento que la inserción de juicios de relevancia dados por especialistas, reformuló el camino investigativo de los modelos de Aprendizaje de la Ordenación; ahora, avizorando un futuro emergente, sería oportuno pensar en la concepción de nuevos modelos que tomasen en consideración no sólo las consultas, los documentos recuperados para tales consultas y los juicios de relevancia, sino además, el contexto en el que se formulan las consultas.

3.6 Conclusiones

Después de haber realizado un estudio experimental para evaluar el desempeño del método propuesto, considerando para ello el rendimiento alcanzado en la ordenación para diversos conjuntos de datos pertenecientes a Letor 3.0 y 4.0; se pudo demostrar que RankFSP manifiesta un alto nivel competitivo con respecto a los principales métodos referenciados en la literatura.

Por otro lado, se corroboró estadísticamente el desempeño de los algoritmos de aprendizaje. Para ello se realizó un análisis estadístico, donde los resultados obtenidos permiten afirmar que ninguno de los métodos mejora de forma significativa la actuación de RankFSP; sin embargo, este supera en cuanto a precisión en la ordenación a los métodos FRank, ListNet, RankBoost y a RankSVM en OHSUMED, y a RankBoost y Regression en NP2004.

Finalmente se puede concluir que para las colecciones de datos analizadas, los métodos que optimizan directamente medidas de desempeño mejoran de forma significativa el desempeño de los métodos tradicionales; sin embargo, en la mayoría de los casos, estos no presentan diferencias significativas entre sí.

Conclusiones

En la presente investigación se evidencia como los métodos tradicionales de Recuperación de Información, han ido cediendo su espacio a nuevas técnicas de aprendizaje automático, debido a que los nuevos problemas de ordenación requieren soluciones y modelos de gran precisión que permitan mejorar la satisfacción de los usuarios.

En este sentido, el Aprendizaje de la Ordenación ha venido a desempeñar un rol vital para muchas aplicaciones de R.I. En este trabajo, enfocando el problema de la ordenación en la recuperación de documentos, se han presentado y descrito las diferentes categorías en las que se enmarcan los métodos desarrollados para dar solución a esta problemática y además se exponen resultados métricos de la producción científica en tal campo.

Se presentó un nuevo método de Aprendizaje de la Ordenación para la Recuperación de Información, utilizando la optimización directa de medidas de desempeño y basándose en el Procedimiento de Búsqueda el Pescador. Este método propuesto, denominado RankFSP, puede optimizar directamente cualquier medida de desempeño utilizada en la R.I.

En el estudio experimental para evaluar el desempeño de RankFSP, considerando para ello el rendimiento alcanzado en la ordenación para los conjuntos de datos OHSUMED, NP2003, NP2004, MQ2007 y MQ2008, se pudo demostrar como el método propuesto manifiesta un alto nivel competitivo (siempre entre los tres primeros lugares en la ordenación) en relación a los principales métodos referenciados en la literatura.

Se realizó además, todo un análisis estadístico, donde los resultados obtenidos permiten afirmar que ninguno de los métodos mejora de forma significativa la actuación de RankFSP; sin embargo, este supera en cuanto a precisión en la ordenación a los métodos FRank, ListNet, RankBoost y a RankSVM en OHSUMED, y a RankBoost y Regression

en NP2004; además el valor de su media se ubica entre las tres mejores posiciones para todos los casos.

Considerando esto, se puede afirmar finalmente que para la mayoría de las colecciones de datos de LETOR 3.0 y 4.0 los métodos que optimizan directamente medidas de desempeño mejoran de forma significativa el desempeño de las propuestas tradicionales; sin embargo, estos en la mayoría de los casos no presentan diferencias significativas entre sí.

Recomendaciones

Como direcciones futuras vinculadas a la presente investigación están:

- ✓ Conducir más experimentos con nuevas y referenciadas colecciones de datos de mediano y gran tamaño para validar la eficiencia, y evaluar el desempeño del método RankFSP bajo disímiles condiciones.
- ✓ Utilizar otras medidas de desempeño, por ejemplo, P@n y NDCG; como función general *E*, con el objetivo de evaluar el comportamiento de la precisión en la ordenación, considerando el mismo diseño experimental conformado para MAP.
- ✓ Rediseñar la función de *ranking* considerando las relaciones y el comportamiento de los rasgos.

Referencias bibliográficas

[1] Salton, G. y McGill, M. H., *Introduction to modern information retrieval*. New York: McGraw-Hill, 1984.

[2] Salvador Oliván, J.A. y Arquero Áviles, R., «Una aproximación al concepto de Recuperación de Información en el marco de la ciencia de la documentación», *Investigación Bibliotecológica*, vol. 20, págs. 13-43, Dic. 2006.

[3] Baeza-Yates, R. y Ribeiro-Neto, B., *Modern information retrieval*. NewYork: Addison-Wesley, 1999.

[4] Salvador Oliván, J.A. y Arquero Avilés, R., «La investigación en Recuperación de Información: Revisión de tendencias actuales y críticas», *Cuadernos de Documentación Multimedia*, vol. 15, 2004.

[5] Salton, G., E.A.F., y Wu, H., «Extended Boolean information retrieval», *Proceedings of the Communications of the ACM*, vol. 26, n^{o}. 11, págs. 1022-1036, 1983.

[6] Wong, S.K., W.Z., y Wong, P.C., «Generalized vector space model in information retrieval», *Proceedings of the Eighth Annual ACM-SIGIR Conference on Research and Development in Information Retrieval, Association for Computing Machinery*, págs. 18-25, 1985.

[7] Croft, W.B. y D.J.H., «Using probabilistic models of information retrieval without relevance information», *Proceedings of the Journal of Documentation*, vol. 35, n^{o}. 4, págs. 285-295, 1975.

[8] Robertson, S., S.W., Jones, S., Hancock-Beaulieu, M., y Gatford, M., «Okapi at TREC-3», *Proceedings of TREC-3*, 1995.

[9] Manning, C.D. y H.S., «Foundations of Statistical Language Processing», *The MIT Press*, 1999.

[10] Deerwester, S.C., Landauer, T.K., Furnas. G.W., y Harshman R.A., «Indexing by latent semantic analysis», *Proceedings of the Journal of the American Society of Information Science*, vol. 41, n^{o}. 6, págs. 391-407, 1990.

[11] Lewis, D.D. y Spärck Jones, K., «Natural language processing for information retrieval», *Proceedings of CACM*, vol. 39, nº. 1, págs. 92-101, 1996.

[12] Brin, S. y L.P., «The anatomy of a large-scale hypertextual web search engine», *Proceedings of WWW*, págs. 107-117, 1998.

[13] Gyongyi, Z., H.G.M., y Pedersen, J., «Combating web spam with TrustRank», *Proceedings of the 30th VLDB Conference*, 2004.

[14] Liu, Y. et al., «BrowseRank: letting web users vote for page importance», *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*, 2008.

[15] «MODELOS DE RECUPERACION DE INFORMACION»,[Online].Available: http://modelosrecuperacion.freeservers.com/..

[16] Alejo, Oscar J., Fernández, Juan M., Huete, Juan F., y Pérez, Ramiro, «Optimización directa de Medidas de Desempeño en el Aprendizaje de la Ordenación usando Enjambre de Partículas», Diploma de Estudios Avanzados del Doctorado en Soft Computing, Universidad de Granada, 2010.

[17] Tie-Yang, Liu, J.X, Qin, T., W.-Y., Xiong, y Li, H., «Letor: Benchmark dataset for research on learning to rank for information retrieval», *Proceedings of SIGIR*, 2007.

[18] K. Jarvelin, A., «Cumulated Gain-Based Evaluation of IR Techniques», *Proceedings of ACM Transactions on Information Systems*, 2002.

[19] Vargas Sandoval, S., «Learning to Rank: Técnicas de Aprendizaje Automático en Recuperación de Información.» 21-Jun-2011.

[20] Zhe Cao, Tao Qin, Tie-Yan Liu, Ming-Feng Tsai, y Hang Li, «Learning to Rank: From Pairwise Approach to Listwise Approach», *Proceedings of the 24th International Conference on Machine Learning*, 2007.

[21] Herbrich, R., Graepel, T., y Obermayer, K., «Large margin rank boundaries for ordinal regression», *Proceedings of the Advances in Large Margin Classifiers*, 2000.

[22] Freund, Y., R.I., Schapire, R., y Singer, Y., «An efficient boosting algorithm for combining preferences», *Proceedings of JMLR*, págs. 933-969, 2003.

[23] Hamed Valizadegan, Rong Jin, Ruofei Zhang, y Jianchang Mao, «Learning to Rank by Optimizing NDCG Measure». .

[24] Joachims, T., «A support vector method for multivariate performance measures», *Proceedings of ICML*, 2005.

[25] Yue, Y., Radlinski, F., F. T., y Joachims, T., «A support vector method for optimizing average precision», *Proceedings of SIGIR*, 2007.

[26] Chapelle, O., Smola, A., y L. Q., «Large Margin Optimization of Ranking Measures», *Proceedings of NIPS*, 2007.

[27] Xu, J., Liu, T-Y, Lu, M., Li, H., y Ma, W-Y., «Directly Optimizing Evaluation Measures in Learning to Rank», *Proceedings of SIGIR*, 2008.

[28] Zheng, Z., H.Z., Zhang, T., Chapelle, O., Chen, K., y Sun, G., «A General Boosting Method and its Application to Learning Ranking Functions for Web Search», *Proceedings of NIPS*, 2007.

[29] Xu, J. y Li, H., «Adarank: a boosting algorithm for information retrieval», *Proceedings of SIGIR*, 2007.

[30] Burges, C., R.R., y Le, Q., «Learning to rank with nonsmooth cost functions», *Proceedings of NIPS*, 2006.

[31] Taylor, M., J.G., Robertson, S., y Minka, T., «Softrank: Optimizing non-smooth rank metrics», *Proceedings of SIGIR*, 2007.

[32] Fan, W., P.P., y Wallace, L., «Nonlinear franking function representations in genetic programming-based ranking discovery for personalized search», *Proceedings of Decision Support Systems*, 2006.

[33] Yeh, J-Y, Liu, J-Y, Ke, H-R, y Chang, W-P, «Learning to rank for information retrieval using genetic programming», *Proceedings of SIGIR*, 2007.

[34] Cossock, D. y T.Z., «Subset ranking using regression», *Proceedings of COLT*, 2006.

[35] Metzler, D., W.C., y McCallum, A., «Direct maximization of rank-based

metrics for information retrieval», *Proceedings of Technical report, CIIR*, 2005.

[36] Chapelle, O., «Direct Optimization for Web Search Ranking», *Proceedings of SIGIR*, 2009.

[37] Alejo, Oscar J., Fernández, Juan M., and Huete, Juan F., "Estudio métrico de la producción científica sobre 'Learning to Rank' en el período 2002-2012."

[38] Zayas Agüero, Pedro M., *El rombo de las investigaciones de las Ciencias Sociales*. .

[39] Bijarro Hernández, Francisco, *Desarrollo estratégico para la Investigación Científica*. eumed.net.

[40] Moreno, Eleazar y Alejo, Oscar J., «L2RLab: Herramienta Informática para asistir el proceso de experimentación y análisis de modelos de Aprendizaje de la Ordenación.», Trabajo de diploma, Universidad de Cienfuegos, 2012.

[41] Sione, Palu, «Java in Soft Computing», 29-Abr-2002. [Online]. Available: http://www.developer.com/Java-in-Soft-Computing.html.

[42] Alejo, Oscar J., Fernández, Juan M., Huete, Juan F., and Concepción, Eduardo R., "FISHERMAN SEARCH PROCEDURE."

[43] V.-D. Cung, S.L. Martins, C.C. Ribeiro and C. Roucairol (2002) Strategies for the parallel implementation of metaheuristics. In: C.C. Ribeiro and C.C. Ribeiro (eds.), *Essays and Surveys in Metaheuristics.* Kluwer Academic Publishers, pp. 263–308.

[44] Zhang, Min, Kuang, Da, Hua, G., Liu, Y., y Ma, S., «Is learning to rank effective for Web search?», *Proceedings of SIGIR*, 2009.

[45] Xu, J., Tie-Yang, Liu, y Hang Li, «The Evaluation Tool in LETOR». [Online]. Available:https://research.microsoft.com/enus/um/beijing/projects/letor/LETOR3.0/EvaluationTool.zip.

[46] Xu, J., Tie-Yang, Liu, y Hang Li, «The Evaluation Tool in LETOR». [Online]. Available:http://research.microsoft.com/enus/um/beijing/projects/letor/LETOR4.0/Evaluation/Eval-Score-4.0.pl.txt

[47] Cárdenas, J., *Análisis Univariado. Capítulo # 6.- Análisis de varianza multifactorial*, C.-A.d.v.m.D.J.R.C.P.F.d.E. UCLV, Editor. 2003, Facultad de Economía, UCLV.

[48] Sheskin, D.J., *Handbook of Parametric and nonparametric statistical procedures*. 3rd ed. ed. 2004: Chapman & Hall / CRC.

[49] Wilcoxon, F., *Individual comparisons by ranking methods.* Biometrics Bulletin, 1945. **1**: p. 80–83.

[50] Siegel, S., *Estadística no paramétrica.* 1988: Trillas.

Bibliografía

[1] Salton, G. y McGill, M. H., *Introduction to modern information retrieval*. New York: McGraw-Hill, 1984.

[2] Salvador Oliván, J.A. y Arquero Áviles, R., «Una aproximación al concepto de Recuperación de Información en el marco de la ciencia de la documentación», *Investigación Bibliotecológica*, vol. 20, págs. 13-43, Dic. 2006.

[3] Baeza-Yates, R. y Ribeiro-Neto, B., *Modern information retrieval*. NewYork: Addison-Wesley, 1999.

[4] Salvador Oliván, J.A. y Arquero Avilés, R., «La investigación en Recuperación de Información: Revisión de tendencias actuales y críticas», *Cuadernos de Documentación Multimedia*, vol. 15, 2004.

[5] Salton, G., E.A.F., y Wu, H., «Extended Boolean information retrieval», *Proceedings of the Communications of the ACM*, vol. 26, nº. 11, págs. 1022-1036, 1983.

[6] Wong, S.K., W.Z., y Wong, P.C., «Generalized vector space model in information retrieval», *Proceedings of the Eighth Annual ACM-SIGIR Conference on Research and Development in Information Retrieval, Association for Computing Machinery*, págs. 18-25, 1985.

[7] Croft, W.B. y D.J.H., «Using probabilistic models of information retrieval without relevance information», *Proceedings of the Journal of Documentation*, vol. 35, nº. 4, págs. 285-295, 1975.

[8] Robertson, S., S.W., Jones, S., Hancock-Beaulieu, M., y Gatford, M., «Okapi at TREC-3», *Proceedings of TREC-3*, 1995.

[9] Manning, C.D. y H.S., «Foundations of Statistical Language Processing», *The MIT Press*, 1999.

[10] Deerwester, S.C., Landauer, T.K., Furnas. G.W., y Harshman R.A., «Indexing by latent semantic analysis», *Proceedings of the Journal of the American Society of Information Science*, vol. 41, nº. 6, págs. 391-407, 1990.

[11] Lewis, D.D. y Spärck Jones, K., «Natural language processing for information retrieval», *Proceedings of CACM*, vol. 39, nº. 1, págs. 92-101, 1996.

[12] Brin, S. y L.P., «The anatomy of a large-scale hypertextual web search engine», *Proceedings of WWW*, págs. 107-117, 1998.

[13] Gyongyi, Z., H.G.M., y Pedersen, J., «Combating web spam with TrustRank», *Proceedings of the 30th VLDB Conference*, 2004.

[14] Liu, Y. et al., «BrowseRank: letting web users vote for page importance», *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*, 2008.

[15] «MODELOS DE RECUPERACION DE INFORMACION»,[Online].Available: http://modelosrecuperacion.freeservers.com/..

[16] Alejo, Oscar J., Fernández, Juan M., Huete, Juan F., y Pérez, Ramiro, «Optimización directa de Medidas de Desempeño en el Aprendizaje de la Ordenación usando Enjambre de Partículas», Diploma de Estudios Avanzados del Doctorado en Soft Computing, Universidad de Granada, 2010.

[17] Tie-Yang, Liu, J.X, Qin, T., W.-Y., Xiong, y Li, H., «Letor: Benchmark dataset for research on learning to rank for information retrieval», *Proceedings of SIGIR*, 2007.

[18] K. Jarvelin, A., «Cumulated Gain-Based Evaluation of IR Techniques», *Proceedings of ACM Transactions on Information Systems*, 2002.

[19] Vargas Sandoval, S., «Learning to Rank: Técnicas de Aprendizaje Automático en Recuperación de Información.» 21-Jun-2011.

[20] Zhe Cao, Tao Qin, Tie-Yan Liu, Ming-Feng Tsai, y Hang Li, «Learning to Rank: From Pairwise Approach to Listwise Approach», *Proceedings of the 24th International Conference on Machine Learning*, 2007.

[21] Herbrich, R., Graepel, T., y Obermayer, K., «Large margin rank boundaries for ordinal regression», *Proceedings of the Advances in Large Margin Classifiers*, 2000.

[22] Freund, Y., R.I., Schapire, R., y Singer, Y., «An efficient boosting algorithm for combining preferences», *Proceedings of JMLR*, págs. 933-969, 2003.

[23] Hamed Valizadegan, Rong Jin, Ruofei Zhang, y Jianchang Mao, «Learning to Rank by Optimizing NDCG Measure». .

[24] Joachims, T., «A support vector method for multivariate performance measures», *Proceedings of ICML*, 2005.

[25] Yue, Y., Radlinski, F., F. T., y Joachims, T., «A support vector method for optimizing average precision», *Proceedings of SIGIR*, 2007.

[26] Chapelle, O., Smola, A., y L. Q., «Large Margin Optimization of Ranking Measures», *Proceedings of NIPS*, 2007.

[27] Xu, J., Liu, T-Y, Lu, M., Li, H., y Ma, W-Y., «Directly Optimizing Evaluation Measures in Learning to Rank», *Proceedings of SIGIR*, 2008.

[28] Zheng, Z., H.Z., Zhang, T., Chapelle, O., Chen, K., y Sun, G., «A General Boosting Method and its Application to Learning Ranking Functions for Web Search», *Proceedings of NIPS*, 2007.

[29] Xu, J. y Li, H., «Adarank: a boosting algorithm for information retrieval», *Proceedings of SIGIR*, 2007.

[30] Burges, C., R.R., y Le, Q., «Learning to rank with nonsmooth cost functions», *Proceedings of NIPS*, 2006.

[31] Taylor, M., J.G., Robertson, S., y Minka, T., «Softrank: Optimizing non-smooth rank metrics», *Proceedings of SIGIR*, 2007.

[32] Fan, W., P.P., y Wallace, L., «Nonlinear franking function representations in genetic programming-based ranking discovery for personalized search», *Proceedings of Decision Support Systems*, 2006.

[33] Yeh, J-Y, Liu, J-Y, Ke, H-R, y Chang, W-P, «Learning to rank for information retrieval using genetic programming», *Proceedings of SIGIR*, 2007.

[34] Cossock, D. y T.Z., «Subset ranking using regression», *Proceedings of COLT*, 2006.

[35] Metzler, D., W.C., y McCallum, A., «Direct maximization of rank-based

metrics for information retrieval», *Proceedings of Technical report, CIIR*, 2005.

[36] Chapelle, O., «Direct Optimization for Web Search Ranking», *Proceedings of SIGIR*, 2009.

[37] Alejo, Oscar J., Fernández, Juan M., and Huete, Juan F., "Estudio métrico de la producción científica sobre 'Learning to Rank' en el período 2002-2012."

[38] Zayas Agüero, Pedro M., *El rombo de las investigaciones de las Ciencias Sociales*. .

[39] Bijarro Hernández, Francisco, *Desarrollo estratégico para la Investigación Científica*. eumed.net.

[40] Moreno, Eleazar y Alejo, Oscar J., «L2RLab: Herramienta Informática para asistir el proceso de experimentación y análisis de modelos de Aprendizaje de la Ordenación.», Trabajo de diploma, Universidad de Cienfuegos, 2012.

[41] Sione, Palu, «Java in Soft Computing», 29-Abr-2002. [Online]. Available: http://www.developer.com/Java-in-Soft-Computing.html.

[42] Alejo, Oscar J., Fernández, Juan M., Huete, Juan F., and Concepción, Eduardo R., "FISHERMAN SEARCH PROCEDURE."

[43] V.-D. Cung, S.L. Martins, C.C. Ribeiro and C. Roucairol (2002) Strategies for the parallel implementation of metaheuristics. In: C.C. Ribeiro and C.C. Ribeiro (eds.), *Essays and Surveys in Metaheuristics.* Kluwer Academic Publishers, pp. 263–308.

[44] Zhang, Min, Kuang, Da, Hua, G., Liu, Y., y Ma, S., «Is learning to rank effective for Web search?», *Proceedings of SIGIR*, 2009.

[45] Xu, J., Tie-Yang, Liu, y Hang Li, «The Evaluation Tool in LETOR». [Online]. Available:https://research.microsoft.com/enus/um/beijing/projects/letor/LETOR3.0/EvaluationTool.zip.

[46] Xu, J., Tie-Yang, Liu, y Hang Li, «The Evaluation Tool in LETOR». [Online]. Available:http://research.microsoft.com/enus/um/beijing/projects/letor/LETOR4.0/Evaluation/Eval-Score-4.0.pl.txt

[47] Cárdenas, J., *Análisis Univariado. Capítulo # 6.- Análisis de varianza multifactorial*, C.-A.d.v.m.D.J.R.C.P.F.d.E. UCLV, Editor. 2003, Facultad de Economía, UCLV.

[48] Sheskin, D.J., *Handbook of Parametric and nonparametric statistical procedures*. 3rd ed. ed. 2004: Chapman & Hall / CRC.

[49] Wilcoxon, F., *Individual comparisons by ranking methods.* Biometrics Bulletin, 1945. **1**: p. 80–83.

[50] Siegel, S., *Estadística no paramétrica.* 1988: Trillas.

[51] Tie-Yan, Liu, “Learning to Rank for Information Retrieval.” 2008.

[52] Pernús, Diana L, “Desarrollo de un nuevo método de Aprendizaje de la Ordenación, basado en el algoritmo de optimización global Supernova.,” Carlos Rafael Rodríquez, 2012.

[53] La Serna, Nora, Román, Ulises, Osorio, Norberto, Benito, Oscar, Espezúa, Jimy, and Vega, Hugo, “ESTUDIO Y EVALUACIÓN DE LOS SISTEMAS DE RECUPERACIÓN DE INFORMACIÓN.” 2004.