Journal of Information Technology Management
ISSN-OnLine: 2980-7972
University of Tehran

Fusion of Graph-Based Ensembles Using Fuzzy Integrals for Persian Learning to Rank

Maryam Piroozmand

Department of Algorithms and Computation, School of Engineering Sciences, College of Engineering, University of Tehran, Tehran, Iran. E-mail: m.piroozmand@ut.ac.ir

Ali Moeini*

*Corresponding author, Professor, Department of Algorithms and Computation, School of Engineering Sciences, College of Engineering, University of Tehran, Tehran, Iran. E-mail: moeini@ut.ac.ir

Mojtaba Mazoochi

Assistant prof., ICT Research Institute, Tehran, Iran. E-mail: mazoochi@itrc.ac.ir

Amir Hosein Keyhanipour

Assistant prof., Computer Engineering Department, Faculty of Engineering, College of Farabi, University of Tehran, Tehran, Iran. E-mail: keyhanipour@ut.ac.ir

Journal of Information Technology Management, 2026, Vol. 18, Issue 3, pp. 47-81
Published by the University of Tehran, College of Management
doi: https://doi.org/10.22059/jitm.2026.412047.4443
Article Type: Research Paper

Received: January 16, 2026
Received in revised form: March 24, 2026
Accepted: May 03, 2026
Published online: July 01, 2026

Abstract

The rapid growth of Persian-language Web content has created an urgent need for effective Learning to Rank (LTR) systems. However, developing such systems for Persian presents significant challenges, including Persian's morphological complexity, its unique Web link topology, and the scarcity of annotated resources relative to dominant languages such as English. Existing methods often rely on hand-crafted features, apply graph techniques in isolation, or employ simple ensemble fusion, lacking a unified framework to integrate structural semantics and handle model uncertainty. This paper proposes PersianRank, a novel LTR framework that bridges these gaps by modeling query-document relationships as a weighted bipartite graph to extract graph-based structural features. It applies deep neural networks to predict these graph features for unseen query-document pairs, enabling feature augmentation during the testing phase. A diverse ensemble of base rankers is then trained on the enriched feature set, and their predictions are fused via the Choquet fuzzy integral, which

performs a non-linear, uncertainty-aware aggregation. Evaluated on the dotIR dataset, PersianRank achieves state-of-the-art performance, improving P@1 by 17.1% relative to the best baseline (MDPRank) and demonstrating notable top-rank accuracy through effective graph-based feature space expansion and uncertainty-aware model fusion. The framework is especially effective for top-rank retrieval, a critical aspect in user-centric search systems.

Keywords: Learning to Rank, Persian Web Retrieval, Graph-Based Features, Ensemble Fusion, Fuzzy Integral, Uncertainty-Aware Ranking

Introduction

Persian has become one of the fastest-growing languages on the Web. As of late 2025, it accounts for over 2% of published pages and ranks as the 13th most common content language among the top 10 million websites (Wikipedia, 2026), a large portion of which is hosted under the .ir domain. This expansion, coupled with over 110 million Persian speakers globally (Eberhard et al., 2025), makes accurate and effective Web search essential. As an established framework for training models on rich feature sets, LTR has significantly enhanced the performance of Web search engines. However, developing high-performance LTR systems for Persian faces major challenges, including the language's morphological complexity, its unique Web link topology, and the relative scarcity of large-scale, annotated Persian benchmark datasets compared to the available resources for English and other dominant languages.

Research on Persian Web ranking has evolved along three primary methodological trajectories: LTR with feature engineering, graph-based and link analysis methods, and cross-lingual or language-specific NLP techniques. Prior LTR approaches for Persian have often relied on hand-crafted features, click-through data, reinforcement learning, or genetic programming. Graph-based methods have been applied in isolation to tasks such as entity linking and keyword extraction, while NLP techniques have addressed linguistic processing challenges. However, a significant research gap persists: these methods seldom integrate graph structural semantics into a unified LTR feature expansion framework. Specifically, while graph methods exist, they rarely model query-document relationships as explicit bipartite structures to derive topological features that augment traditional content-based representations. Furthermore, existing ensemble fusion strategies for multiple relevance estimators typically employ fixed or linear combinations, lacking a principled mechanism to handle the inherent uncertainty and conflicting evidence among different base models. This is particularly problematic in Persian Web retrieval, where feature sparsity and linguistic variability amplify model disagreement. The Choquet fuzzy integral provides a mathematically grounded mechanism for modeling interactions and redundancies among estimators, moving beyond naïve averaging or weighted sums. Our proposed algorithm, PersianRank, directly addresses these limitations by leveraging graph theory to systematically

expand the feature space, leveraging deep learning to estimate these graph-based features, and applying fuzzy integral operators to perform an uncertainty-aware fusion of multiple machine learning models. Thus, this work investigates three core research questions: (RQ1) Can graph-based structural features derived from query-document bipartite graphs provide complementary ranking signals beyond traditional content-driven and link-based features? (RQ2) Does the Choquet fuzzy integral offer superior fusion performance compared to linear, fixed-weight, or other non-linear ensemble fusion strategies in Persian LTR? (RQ3) Can the integrated approach of feature space augmentation and uncertainty-aware fusion achieve state-of-the-art performance on the Persian datasets?

PersianRank operates through four cohesive stages to bridge the identified gaps. First, it employs a query-aware strategy to preprocess and partition the dataset, thereby preventing data leakage. Second, it models query-document relationships as a weighted bipartite graph, from which 22 structural features (11 for queries and 11 for documents) that capture centrality, connectivity, and higher-order patterns are extracted. Subsequently, a dedicated deep neural network is trained to predict each graph feature for unseen test data, effectively augmenting the original feature space with latent relational knowledge. Third, a diverse ensemble of base rankers is trained on this enriched set. The predictions of the base rankers are then aggregated using the Choquet fuzzy integral, which employs an optimized fuzzy measure to perform a context-sensitive, uncertainty-aware fusion that accounts for model interactions and redundancies. The framework is rigorously evaluated on the dotIR dataset, a comprehensive Persian Web corpus containing 5,000 query-document pairs with human relevance judgments and 56 finely-graded original features.

The primary contributions of the current work are threefold:

- We introduce a novel graph-based feature augmentation pipeline for LTR that models query-document pairs as a bipartite graph, extracting 22 structural features to capture latent relational patterns beyond traditional content and link metrics.
- We propose an uncertainty-aware fusion mechanism using the Choquet fuzzy integral, which learns an optimal measure to non-linearly aggregate diverse base model predictions, effectively handling model interdependencies and conflicting evidence.
- We present a comprehensive empirical evaluation on the Persian dotIR dataset, demonstrating that PersianRank achieves state-of-the-art performance, particularly in top-rank accuracy.

The remainder of this paper is structured as follows: Section 2 surveys related works in Persian Web ranking. Section 3 details the proposed PersianRank algorithm. Section 4 presents the experimental setup and evaluation results. Section 5 provides an analytical discussion on the experimental findings, and Section 6 concludes the paper and suggests future directions.

Literature Review

Research in ranking and retrieval for Persian Web content has evolved along several interconnected methodological paths. Broadly, existing works can be categorized into three primary groups: (1) Learning to Rank and Feature Engineering Approaches, which focus on developing and optimizing ranking models using various machine learning techniques and engineered features; (2) Graph-Based and Link Analysis Methods, which leverage the structural properties of documents, entities, or Web graphs for ranking; and (3) Cross-Lingual and Language-Specific NLP Techniques, which address the challenges of Persian language processing and cross-lingual information retrieval.

The first category encompasses learning to rank and feature engineering approaches. These works primarily develop and apply machine learning models to rank documents, often incorporating user feedback and query-specific features. Baradaran Hashemi et al. explored various ensemble techniques like Ordered Weighted Averaging (OWA) and Support Vector Machines (SVM) for fusing features (Baradaran Hashemi et al., 2010). For Persian Web content, Derhami et al. reformulated ranking as a reinforcement learning problem by introducing the RLRank algorithm (Derhami et al., 2013). Azarbonyad et al. employed LTR to rank translation candidates from multiple resources for the Cross-Language Information Retrieval (CLIR) task (Azarbonyad et al., 2014). Latifi and Nematbakhsh applied LTR with query-independent features for ranking Resource Description Framework (RDF) entities (Latifi & Nematbakhsh, 2014). Rahimi et al. used Ranking SVM to align documents for corpus construction (Rahimi et al., 2016). Ghanbari and Shakery proposed an LTR model tailored per query using cross-lingual features (Ghanbari & Shakery, 2018). Similarly, Azarbonyad et al. mapped monolingual IR features into cross-lingual space using LTR (Azarbonyad et al., 2019). Keyhanipour customized the CF-Rank algorithm with feature transformation and OWA fusion (Keyhanipour, 2019b). The RRLUFF algorithm uses reinforcement learning with user feedback for ranking (Derhami et al., 2019). Keyhanipour introduced click-through features and reinforcement learning in the QRC-Rank algorithm (Keyhanipour, 2019a). Later, he presented the MGP-Rank algorithm through evolving ranking functions with genetic programming to generate and optimize click-through features (Keyhanipour, 2020). Finally, Ghanbari and Shakery introduced a novel listwise loss function for training bilingual ranking models (Ghanbari & Shakery, 2021).

The second category comprises graph-based and link analysis methods, which utilize graph structures for ranking, entity linking, or keyword extraction. Rogers proposed learning from and repurposing the "natively digital" methods embedded in the medium itself—such as analyzing hyperlinks as reputation markers, using search engine results for societal diagnosis, and studying archived Web histories (Rogers, 2010). Rezvani and Hashemi integrated content similarity into Topic-Sensitive PageRank (Rezvani & Hashemi, 2012). Furthermore, Shyam et al. modeled Web navigation as a Markov Decision Process (MDP) on the hyperlink graph

with the Semantic Rank algorithm (Shyam et al., 2013). Zeraatkar et al. enhanced PageRank by incorporating spam scores and page age (Zeraatkar et al., 2019). Asgari-Bidhendi et al. introduced ParsEL, combining graph-based and similarity scores for entity linking (Asgari-Bidhendi et al., 2020). Azarafza and his team applied the TextRank algorithm on a co-occurrence graph for keyword extraction (Azarafza M. et al., 2020). Bostan et al. proposed HybridMaxSim, combining Word2Vec and BERT embeddings via a graph-inspired similarity formula (Bostan et al., 2023). For the web page ranking problem, Keyhanipour constructed feature-similarity graphs to analyze LTR benchmarks (Keyhanipour, 2024).

The third category covers Cross-Lingual and Language-Specific NLP techniques, focusing on Persian language processing, textual understanding, and cross-lingual adaptation. Moosavi evaluated a ranking model for coreference resolution, adapting features for Persian linguistics (Moosavi, 2009). Mirzababaei used an SMT framework with discourse features for spelling correction (Mirzababaei, 2013). Veisi and Shandi developed a rule and NLP-based Question-Answering (QA) system for the medical domain (Veisi & Shandi, 2020). Rahimi and Shamsfard hybridized data-mined rules with BERT for contradiction detection (Rahimi & Shamsfard, 2023). They also built ontologies and combined rules with BERT for textual entailment (Rahimi & ShamsFard, 2024). Recently, Thakur et al. introduced a multilingual RAG benchmark with a surrogate judge model (Thakur et al., 2025).

Table 1 synthesizes these works, highlighting their core ideas, strengths, and limitations, thereby establishing the context and motivation for the proposed PersianRank framework.

Table 1. Summary of related literature

Category	Algorithm	Major Ideas	Strengths	Drawbacks
LTR & Feature Engineering	(Ghanbari & Shakery, 2018)	Local LTR model per query using cross-lingual features.	Adapts ranking to the specific query context.	Computationally intensive for per-query model building.
	(Azarbonyad et al., 2014)	LTR to rank translations from multiple resources for CLIR.	Effectively combines diverse translation resources.	Relies on the quality of underlying translation probabilities.
	(Azarbonyad et al., 2019)	Maps monolingual IR features to cross-lingual space via LTR.	Feature-level combination outperforms simple fusion.	Complex feature engineering pipeline.
	(Rahimi et al., 2016)	Uses Ranking SVM to align documents for corpus building.	Improves corpus quality over heuristics.	Requires parallel data for training simulation.
	(Ghanbari & Shakery, 2021)	Novel listwise loss combining mono- and cross-lingual losses.	Leverages training data in both languages effectively.	Loss function design is non-trivial.
	(Keyhanipour, 2019a)	QRC-Rank uses click-through data & reinforcement learning.	Leverages implicit user feedback for Persian Web.	Dependent on the availability of click-through logs.
	(Keyhanipour,	MGP-Rank uses genetic	Automatically	High computational

	2020)	programming to evolve ranking functions.	generates optimized ranking functions.	cost of genetic programming.
	(Derhami et al., 2013)	RLRank treats ranking as an RL problem on the Web graph.	Combines connectivity-based and content-based (BM25) signals.	RL training can be unstable and data-hungry.
	(Derhami et al., 2019)	RL algorithm using direct user feedback (clicks) for ranking.	Incorporates real user interaction signals.	Requires substantial click data, risk of bias.
	(Keyhanipour, 2019b)	CF-Rank transforms feature space & fuses models via OWA.	Improved performance via specialized models and fusion.	Fusion operator tuning adds complexity.
	(Baradaran Hashemi et al., 2010)	Explores ensemble methods (OWA, SVM) to fuse features.	OWA and SVM fusion improve search accuracy.	Ensemble design requires careful selection.
	(Latifi & Nematbakhsh, 2014)	Ranks RDF entities using query-independent graph/Web features.	Applicable to any SPARQL query over the knowledge graph.	May not capture query-specific nuances.
Graph-Based & Link Analysis	(Asgari-Bidhendi et al., 2020)	Unsupervised entity linking using graph-based and similarity scores.	High F1-score on social media data; uses FarsBase KG.	Performance depends on knowledge graph coverage.
	(Azarafza M. et al., 2020)	Applies TextRank on the co-occurrence graph for keyword extraction.	Unsupervised; identifies bursty topics from microblogs.	Primarily suited for extracting noun and adjective keywords.
	(Keyhanipour, 2024)	Constructs a feature-similarity graph to compare LTR datasets.	Reveals structural differences between benchmarks.	Analysis is descriptive rather than prescriptive.
	(Bostan et al., 2023)	HybridMaxSim combines Word2Vec & BERT embeddings for ranking.	Achieves high nDCG; leverages static and dynamic embeddings.	Computationally expensive to compute HybridMaxSim.
	(Shyam et al., 2013)	Models Web navigation as MDP on hyperlink graph (Semantic Rank).	Outperforms PageRank; refines list for ad relevance.	Assumes reward is inversely proportional to out-degree.
	(Rezvani & Hashemi, 2012)	Biases PageRank jump probability with content similarity.	Improves topical relevance for Persian Web content.	Requires topic distribution as input.
	(Zeraatkar et al., 2019)	Enhances PageRank by integrating spam score and page age.	Improves ranking accuracy and spam resilience.	Requires a reliable spam detection mechanism.
	(Rogers, 2010)	Advocates digital methods (e.g., hyperlinks as reputation markers).	Grounds social analysis in native digital data.	Methodological contribution rather than a ranking algorithm.
Cross-Lingual & Language-Specific NLP	(Rahimi & ShamsFard, 2024)	Combines causality/contradiction ontologies with fine-tuned BERT.	Integrates commonsense knowledge for implicit relations.	Ontology construction is manual and domain-limited.
	(Rahimi & Shamsfard, 2023)	Hybrid method combining data-mined rules and the BERT model.	Handles multiple contradiction categories; F-measure >80%.	Rule mining may not cover all linguistic patterns.

	(Mirzababaei et al., 2013)	Uses the SMT framework with discourse features for spell checking.	Significant recall improvement for Persian/English.	Relies on the quality of SMT candidate generation.
	(Thakur et al., 2025)	Multilingual RAG benchmark with heuristic & LLM-based evaluation.	An efficient surrogate judge approximates costly LLM rankings.	Surrogate judge performance depends on training data.
	(Veisi & Shandi, 2020)	Rule-based and NLP-based QA system for the Persian medical domain.	Achieves 83.6% accuracy on the domain-specific dataset.	Primarily rule-based, may lack generalization.
	(Moosavi, 2009)	Evaluates the MaxEnt ranking model for Persian pronoun resolution.	Adapt features for Persian linguistic challenges.	Decision trees outperformed the ranking.

While existing works provide a solid foundation, several research gaps remain evident. Many LTR approaches for Persian Web content rely heavily on hand-crafted or click-through features, which may not fully capture the complex latent relationships between documents and queries. Graph-based methods are often applied in isolation—for link analysis, entity linking, or keyword extraction—but are seldom integrated into a unified feature expansion framework for LTR. Furthermore, the integration of multiple relevance estimation models typically relies on fixed or linear fusion strategies, lacking a principled mechanism to handle inherent uncertainty and conflicting evidence across ranking models. The proposed algorithm addresses these gaps by leveraging graph theory to expand the feature space systematically, using deep learning to estimate these graph-based features, and applying fuzzy integral operators to perform an uncertainty-aware fusion of multiple machine learning relevance estimators. It creates a more holistic and adaptive ranking framework specifically designed for the complexities of Persian Web content.

Methodology

The Proposed Algorithm

This paper introduces PersianRank, a novel LTR framework that integrates graph theory with ensemble learning and uncertainty-aware fusion to improve ranking accuracy. At its core, PersianRank models query-document relationships as a bipartite graph, extracts advanced structural features such as centrality and connectivity metrics, and incorporates them into a multi-stage ranking pipeline. The proposed method comprises four key stages: (1) query-aware dataset preprocessing and partitioning, (2) feature space augmentation through graph-based feature extraction, (3) training diverse base ranking models, and (4) uncertainty-aware fusion of model predictions using the Choquet fuzzy integral operator. By capturing not only intrinsic document-query features but also latent relational structures within the dataset, PersianRank delivers an effective ranking function. The complete workflow of this framework is illustrated in Figure 1.

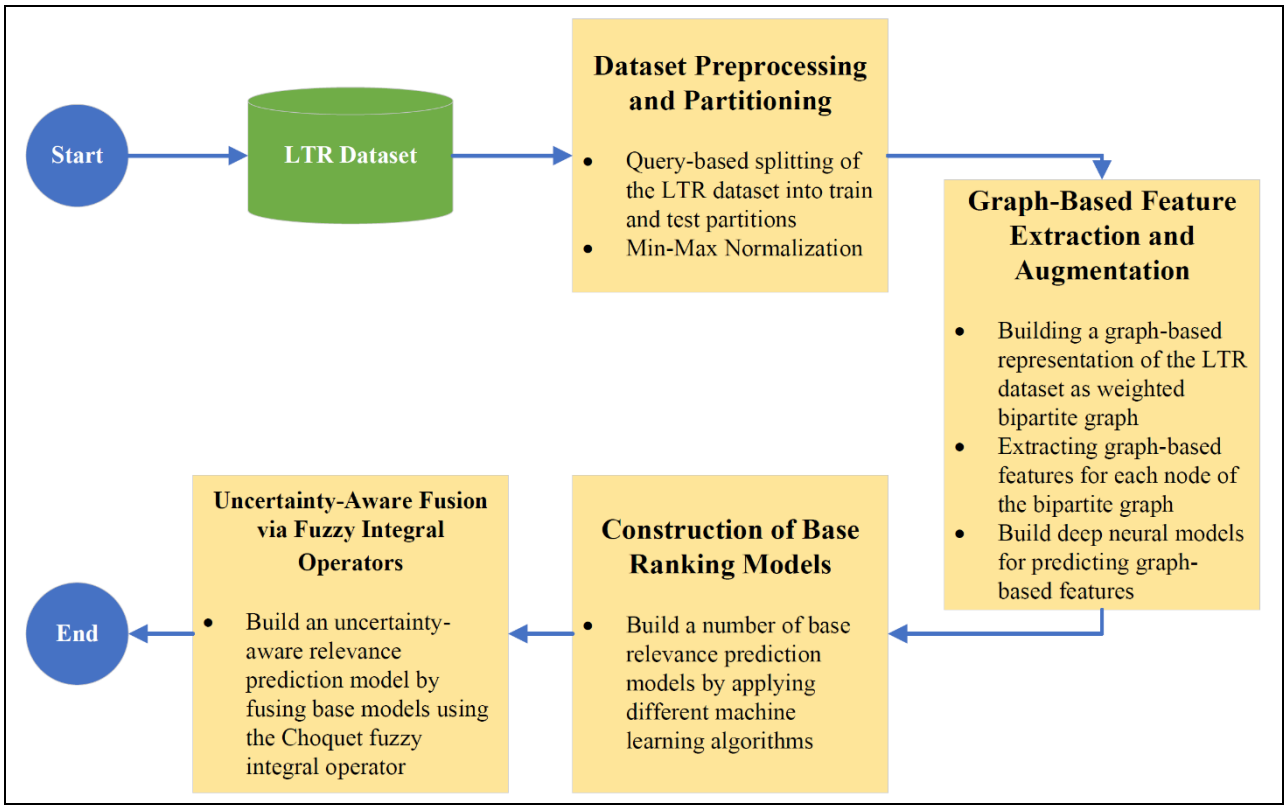

Figure 1. Flowchart of the Proposed LTR Framework

A detailed breakdown of the above-mentioned key steps of the proposed algorithm is as follows:

Step 1: Dataset Preprocessing and Partitioning

The initial phase prepares the LTR dataset for subsequent graph-based processing and evaluation. The dataset D consists of tuples:

$$(q_i, d_j, \mathbf{f}_{ij}, r_{ij}) \tag{1}$$

where q_i is a query, d_j is a document, $\mathbf{f}_{ij} \in \mathbb{R}^n$ is the feature vector containing n features, and r_{ij} is the relevance label. First, the dataset is partitioned into training and test sets using a query-aware splitting strategy. Instead of random splitting, documents belonging to the same query are kept together, and a stratified sample of documents per query is allocated to the test set. This preserves the query-group structure and prevents data leakage. Let Q be the set of all unique queries. For each query $q \in Q$, let D_q be its associated documents. A proportion τ of documents from D_q is randomly selected for the test set T_q, and the remainder form the training set S_q. The final training and test sets (S and T) are:

$$S = \bigcup_{q \in Q} S_q \text{ and } T = \bigcup_{q \in Q} T_q \quad (2)$$

Subsequently, all features—including the original features and the newly derived graph features—are normalized using min-max scaling to the $[0,1]$ range to ensure rigorous model training.

Step 2: Graph-Based Feature Extraction and Augmentation

This step constructs a graph-based representation of the training dataset and extracts sophisticated topological features to augment the original feature set. The query-document relationships are modeled as a weighted bipartite graph $G = (V, E, w)$, where the vertex set V is partitioned into query nodes V_q and document nodes V_d. An edge $e = (q_i, d_j) \in E$ exists if the document d_j is associated with the query q_i in the training data. Its weight $w(e)$ is set to the corresponding relevance score r_{ij}. If $r_{ij} = 0$, a small constant is used instead to maintain connectivity. This graph encodes the latent relational structure of the ranking problem.

From this graph, we compute 11 graph-driven features for each node type, yielding 22 features in total (11 query features and 11 document features) for each query-document pair. For each node, these features capture its importance, centrality, and connectivity within the corresponding graph. The 11 features per node type were selected based on their established utility in a network analysis context (centrality, cohesion, clustering) and their potential to capture distinct relational aspects in bipartite LTR graphs. We conducted a preliminary correlation analysis to ensure low redundancy among selected features. Features with intercorrelation exceeding 0.85 were excluded to maintain diversity in the structural signal. A full listing of these features and their formulas is provided in Table 2 and Equations (3)-(13). These features are derived from several established graph algorithms and metrics, each providing a unique perspective on a node's role. The computed features are then mapped to the original training data: each query-document pair (q_i, d_j) is augmented with the graph features of q_i and d_j, resulting in an enriched feature vector. For the unseen test set, these graph features cannot be computed directly from the test data's isolated graph. Therefore, a dedicated predictive model is trained to infer each graph-based feature for test queries and documents using only the standard 56 features of those pairs as input. This approach assumes that the relational patterns captured by graph features in the training set generalize to unseen pairs via their original feature representations. The use of Deep Neural Networks (DNNs) allows for non-linear mapping from content/link features to structural graph properties. Specifically, we train 22 separate DNNs: one for each of the 11 query-specific graph features and one for each of the 11 document-specific graph features. We selected this decoupled design after empirical comparison with a multi-output alternative. In preliminary experiments, a single DNN predicting all 22 features simultaneously—using a shared trunk of [128, 64]

layers followed by task-specific heads—produced inferior imputation accuracy (average MAE of 0.073 vs. 0.041 for the independent models). The performance gap was most pronounced for structurally complex features such as the Betweenness Centrality (MAE: 0.098 vs. 0.045), indicating negative task interference among heterogeneous graph properties. The separate DNNs also converged more rapidly (35 vs. 62 epochs) and generalized better on test data. Each DNN takes the original 56-dimensional feature vector of a query-document pair as input and outputs the corresponding graph feature value. Each DNN is a fully connected network with hidden layer sizes [64, 32, 16], ReLU activations, dropout rates between 0.3 and 0.5, and trained using the Adam optimizer with mean squared error loss. Early stopping is used to prevent overfitting. Details of the setting configuration parameters of each step of the proposed method are described in Section 4. These models are trained on the training set and then applied to the test set to impute the missing graph-based features, yielding a complete, augmented feature set for the testing phase that effectively transfers the structural knowledge learned from the training graph.

To effectively capture the latent relational structure within the LTR dataset, we model query-document relationships as a weighted bipartite graph and extract a comprehensive suite of 11 graph-based features. These features are designed to quantify different aspects of a node's (query or document) position, connectivity, and role within this graph. For clarity and analytical insight, we categorize these features into three categories. The first category, 'Centrality & Importance', comprises features that measure a node's overall prominence and influence within the global network. The second, 'Local Connectivity & Cohesion', focuses on the density and interconnectedness of a node's immediate neighborhood, revealing local community structure. The third, 'Higher-Order Structural Patterns', captures derived relationships and clustering among nodes of the same type, uncovering latent similarity groups. Table 2 details each feature within its category, describing its conceptual foundation and its specific interpretation for ranking relevance in the LTR context.

Table 2. Categorization of graph-based features extracted from the query-document bipartite graph

Category	Feature	Main Idea	LTR Interpretation
Centrality & Importance	Weighted Degree	Sum of connection strengths to neighbors	Total relevance associations; identifies central documents or queries
	PageRank	Global importance via recursive influence	Network-wide authority and influence of nodes
	Eigenvector Centrality	Importance of connections to important nodes	Membership in influential clusters or communities
	Betweenness Centrality	Role as a bridge on the shortest paths	Connector between different topics/domains
	Closeness Centrality	Average distance to all reachable nodes	Efficiency in information access and propagation
Local	Bipartite Clustering	Likelihood neighbors share	Local community density and topical

Connectivity & Cohesion		other connections	coherence
	Average Neighbor Degree	Average connectivity of neighbors	Association with network hubs and visibility
	Common Neighbors Ratio	Proportion of neighbor-pairs with shared connections	Basic local interconnectedness
	Weighted Common Neighbors	Average strength of shared connections	Intensity of local community bonds
Higher-Order Structural Patterns	Projection Degree	Total strength in the co-occurrence network	Overall similarity to peers of the same type
	Projection Clustering	Transitivity among similar peers	Membership in tightly-knit similarity clusters

This subsection provides a detailed description of the extracted graph-driven features. Let G be the constructed bipartite graph, $N(u)$ as the set of neighbors of the node u, and w_{uv} as the weight of the edge (u, v). The following features are computed for each node that could be either a query or a document:

1. Weighted Degree: The sum of weights of edges incident to a node. It measures the total strength of a node's connections as:

$$C_{deg}(u) = \sum_{v \in N(u)} w_{uv} \tag{3}$$

2. PageRank: A measure of node importance based on the random surfer model, adapted for weighted edges. It recursively defines a node's score based on the scores of its neighbors. Formally, for node u, it could be defined as:

$$PR(u) = \frac{1-\alpha}{|V|} + \alpha \sum_{v \in N(u)} \frac{w_{vu} \cdot PR(v)}{\sum_{k \in N(v)} w_{vk}} \tag{4}$$

where α is the damping factor.

3. Eigenvector Centrality (Bipartite-Adapted): Measures a node's influence based on the influence of its neighbors (Newman, 2018). For bipartite graphs, it is computed via projections. A document-document graph is created where documents are connected if they share a common query, with edge weights reflecting the strength of shared connections. In the same way, the query-query graph is built. Eigenvector centrality is then calculated on such a projection, as follows:

$$\mathrm{Ax} = \lambda \mathrm{x} \tag{5}$$

where **A** is the adjacency matrix of the projection graph, and the centrality score is the corresponding eigenvector element.

4. Betweenness Centrality: The fraction of all shortest paths in the graph that pass through a given node. It identifies nodes that act as bridges. Formally, for node u, this measure is defined as:

$$C_{bet}(u) = \sum_{s \neq u \neq t \in V} \frac{\sigma_{st}(u)}{\sigma_{st}} \tag{6}$$

where σ_{st} is the total number of shortest paths from s to t, and $\sigma_{st}(u)$ is the number of those paths passing through u.

5. Closeness Centrality. The inverse of the average shortest path distance from a node to all other reachable nodes. It measures how quickly a node can interact with others (Newman, 2018). Formally, for node u we have:

$$C_{close}(u) = \frac{|R(u)|}{\sum_{v \in R(u)} d(u,v)} \tag{7}$$

where $R(u)$ is the set of nodes reachable from u, and $d(u,v)$ is the shortest-path distance.

6. Bipartite Clustering Coefficient (Dot mode): This metric measures the local connectivity density among a node's neighbors in the bipartite graph, excluding connections through the node itself (Newman, 2018). For node u, this coefficient quantifies the likelihood that two of its neighbors are themselves connected through an alternative common neighbor, excluding u itself. This metric captures the extent to which the node's immediate network neighborhood forms an interconnected community. Specifically, for node u with neighbor set $N(u)$, it calculates the proportion of neighbor pairs sharing at least one common neighbor other than u. Formally:

$$CC_{dot}(u) = \frac{\sum_{v_i, v_j \in N(u), i<j} \left| N(v_i) \cap N(v_j) \backslash \{u\} \right|}{\binom{|N(u)|}{2}} \tag{8}$$

where $\binom{|N(u)|}{2}$ is the total number of unique neighbor pairs. The term $N(v_i) \cap N(v_j) \setminus \{u\}$ computes the intersection of the neighbor sets of two distinct neighbors v_i and v_j of u, then removes u from this intersection to count only other common neighbors. The operator $\setminus$ denotes the set difference, ensuring that the trivial connection through u itself is not counted. If $|N(u)| < 2$, the coefficient is defined as zero, as no pairs exist. This coefficient is computed symmetrically for both query and document nodes, reflecting the density of alternative connections within each node's neighborhood.

7. Average Neighbor Degree: For a specified node, the average weighted degree of all neighboring nodes is:

$$AvgND(u) = \frac{1}{|N(u)|}\sum_{v\in N(u)} C_{deg}\,(v) \tag{9}$$

8. Common Neighbors Ratio: This feature quantifies the local connectivity structure between a node's neighbors (Barabási & Pósfai, 2016). For a given node u, we examine all unique pairs of its neighboring nodes (n_i, n_j). The metric calculates the proportion of such neighbor-pairs that have at least one additional common neighbor other than u itself in the bipartite graph. A high ratio indicates that the node's neighbors tend to be interconnected through other parts of the network, suggesting the node resides within a densely interconnected local module or community. Formally, for node u with neighbor set $N(u)$, the ratio is defined as:

$$R_{cn}(u) = \frac{\left|\{(n_i, n_j) \in N(u)\times N(u): i<j, \left|N(n_i)\cap N(n_j)\setminus\{u\}\right|>0\}\right|}{\binom{|N(u)|}{2}} \tag{10}$$

where $N(n_i)\cap N(n_j)\setminus\{u\}$ represents the set intersection of the neighbors of n_i and n_j, then removing u if present. The set difference operator $\setminus$ removes element u from the intersection set. The denominator $\binom{|N(u)|}{2}$ is the total number of unique neighbor-pairs. If $|N(u)|<2$, the ratio is defined as zero.

9. Weighted Common Neighbors Score: This is a refined, continuous version of the Common Neighbors Ratio that incorporates edge weight information to measure the strength of shared connections (Newman, 2018). For a node u, we again consider all unique pairs of its neighbors (n_i, n_j). For each such pair, we identify their common neighbors v (excluding u) and compute a contribution based on the geometric mean of the edge weights connecting v to n_i and n_j: $\sqrt{w_{vn_i}\cdot w_{vn_j}}$. The contributions from all common neighbors v are summed to yield a pairwise connection strength score. The final feature value for node u is the average of these pairwise scores across all neighbor pairs. This score captures not only the existence of shared connections but also their intensity. Formally:

$$S_{wcn}(u) = \frac{1}{\binom{|N(u)|}{2}}\sum_{i<j}\left(\sum_{v\in(N(n_i)\cap N(n_j)\setminus\{u\})}\sqrt{w_{vn_i}\cdot w_{vn_j}}\right) \tag{11}$$

Here, $N(n_i)\cap N(n_j)\setminus\{u\}$ again uses set intersection followed by set difference to ensure u is excluded from the common neighbors. The outer summation is over all unique pairs $i<j$ of neighbors of u. If $|N(u)|<2$, the score is defined as zero

10. Projection Degree: This metric quantifies the total strength of a node's indirect connections within a projected network representation derived from the original bipartite graph (Newman, 2018). For a node u belonging to one partition of the bipartite graph (either query or document nodes), we construct a unipartite projection graph $G^{proj} = (V^{proj}, E^{proj}, w^{proj})$ where V^{proj} is the set of all nodes of the same type as u. Two nodes in V^{proj} are connected by an edge if they share at least one common neighbor of the opposite type in the original bipartite graph. The weight of a projected edge between nodes u and v, denoted w_{uv}^{proj}, is computed as the sum over all their common neighbors z (of the opposite type) of the product of the original edge weights w_{uz} and w_{vz}. Formally, the projection degree for the node u is given by:

$$PD(u) = \sum_{v \in V^{proj}, v \neq u} w_{uv}^{proj} \tag{12}$$

where $w_{uv}^{proj} = \sum_{z \in N(u) \cap N(v)} w_{uz} \cdot w_{vz}$. In this equation, $N(u)$ represents the set of neighbors of the node u in the original bipartite graph G, which consists exclusively of nodes of the opposite type due to the bipartite constraint. The intersection $N(u) \cap N(v)$ yields the set of common neighbors shared by u and v. The product value $w_{uz} \cdot w_{vz}$ captures the joint strength of connection through a common neighbor z. This measure effectively aggregates the intensity of all co-occurrence relationships between u and other nodes of the same type across the entire network.

11. Projection Clustering Coefficient: This feature assesses the local transitivity and cohesiveness within the neighborhood of a node in the projected graph (Barabási & Pósfai, 2016). It measures the extent to which nodes that are strongly co-associated with a given node u also tend to be strongly associated with one another. For node u in the projection graph G^{proj}, the weighted clustering coefficient is computed as the proportion of closed weighted triplets centered at u. Specifically, it compares the actual weighted triangles involving u to the maximum possible weighted triangles that could exist among its neighbors. The formal expression for the node u is:

$$PC(u) = \frac{\sum_{v_i, v_j \in N^{proj}(u), i<j} \sqrt{w_{uv_i}^{proj} \cdot w_{uv_j}^{proj}} \cdot \mathbb{I}(v_i, v_j) \cdot w_{v_i v_j}^{proj}}{\sum_{v_i, v_j \in N^{proj}(u), i<j} \sqrt{w_{uv_i}^{proj} \cdot w_{uv_j}^{proj}}} \tag{13}$$

In this formulation, $N^{proj}(u)$ denotes the set of neighbors of node u in the projection graph G^{proj}, which consists of nodes of the same type as u. The term w_{ab}^{proj} represents the edge weight between nodes a and b in the projection graph, as defined previously. The indicator function $\mathbb{I}(v_i, v_j)$ takes the value 1 if an edge exists between v_i and v_j in G^{proj} (i.e.,

if $w_{v_i v_j}^{proj} > 0$), and 0 otherwise. The geometric mean $\sqrt{w_{uv_i}^{proj} \cdot w_{uv_j}^{proj}}$ serves as a weighting factor for each potential triangle, assigning greater importance to triangles where u has strong connections to both v_i and v_j. When an edge exists between v_i and v_j, the product $\sqrt{w_{uv_i}^{proj} \cdot w_{uv_j}^{proj}} \cdot w_{v_i v_j}^{proj}$ contributes to the numerator; otherwise, the contribution is zero. The denominator normalizes by the total weight of all possible triplets centered at u, ensuring the coefficient ranges between 0 and 1. A value close to 1 indicates that nodes sharing strong projected connections with u also form a densely interconnected cluster, suggesting cohesive community structure in the projection space.

Before augmenting the original feature set, all extracted graph-based features undergo Min-Max normalization to the range [0, 1]. This ensures that features with inherently large numerical ranges (e.g., weighted degree) do not dominate the learning process when combined with the original normalized features. The normalization is performed separately for query and document features using statistics computed from the training set.

Step 3: Construction of Base Ranking Models

Using the augmented feature set, the next step trains multiple diverse base ranking models. The goal is to create a pool of predictors that capture different aspects of the relationship between features and relevance. We employ a suite of machine learning algorithms, each with its unique inductive bias, to predict the relevance score r_{ij} for a query-document pair. The models include: K-Nearest Neighbors (KNN), which predicts the class label of each item based on local similarity (Syriopoulos et al., 2025); Random Forest (RF), an ensemble of decision trees that captures complex, non-linear interactions (Breiman, 2001); Gradient Boosting Machines (GBM), which sequentially corrects the errors of previous models (Friedman, 2001); and LightGBM (LGBM), a highly efficient gradient boosting framework (Ke et al., 2017). Each model is trained on the enriched training data S. Their hyperparameters (e.g., number of neighbors for KNN, tree depth for RF, and GBM) are optimized via grid search (see Section 4) to maximize ranking performance metrics like P@1 on a validation set or via cross-validation. This yields a set of trained base models $\mathcal{M} = \{M_1, M_2, \ldots, M_K\}$, each capable of producing a relevance score estimate $\hat{r}_{ij}^{(k)} = M_k(\mathbf{x}_{ij})$, where $\mathbf{x}_{ij}$ is the augmented feature vector.

Step 4: Uncertainty-Aware Fusion via Fuzzy Integral Operators

The final step aggregates predictions from the heterogeneous base models into a unified ranking model. Rather than using simple averaging, we employ the Choquet fuzzy integral, which accounts for interactions and redundancies among the models. The Choquet integral requires a fuzzy measure $\mu: 2^{\mathcal{M}} \rightarrow [0,1]$, which assigns an importance weight not only to

individual models but also to every possible coalition of models. The measure $\mu(S)$ for a subset of models $S \subseteq \mathcal{M}$ reflects the combined significance of that group.

The fuzzy measure μ is initialized based on the individual model performances on a validation set. We empirically validated the choice of P@1 as the base performance metric by comparing against alternative initialization strategies using NDCG@1, NDCG@3, and P@3. Across three independent validation runs, the P@1-initialized measure consistently produced the highest final ranking performance (average NDCG@1 of 0.818 vs. 0.806 for NDCG@1-initialization and 0.809 for P@3-initialization). The superiority of P@1 initialization is attributable to the fuzzy integral's sensitivity to the top-ranked position—the same cutoff where PersianRank demonstrates its largest gains—and the fact that P@1 provides a simpler, less ambiguous signal than graded metrics like NDCG@1, which can conflate partial relevance effects. While NDCG@1-initialization yielded comparable results (within 1.5%), the P@1 strategy was retained for its combination of strong empirical performance and conceptual clarity. For a singleton model subset $\{M_k\}$, we set $\mu(\{M_k\}) = P@1(M_k)/\sum_i P@1(M_i)$. For a non-singleton subset of models $S \subseteq \mathcal{M}$, we initialize its measure as the maximum individual performance among its members: $\mu(S) = max\{M_k \in S\}.\mu(\{M_k\})$, where $\mu(\{M_k\})$ is the normalized P@1 performance of the model M_k on the validation set.

We optimize the fuzzy measure μ via stochastic gradient descent with a learning rate of 0.01. The objective is to minimize a pairwise hinge loss over document pairs that have differing relevance labels: $L = \sum_{(i,j)\in\mathcal{P}} max\left(0{,}1 - \left(C_\mu(\hat{r}_{(i)}) - C_\mu(\hat{r}_{(\mathrm{j})})\right)^2\right)$, where $\mathcal{P}$ is the set of preference pairs from training data. Optimization runs for 100 epochs with early stopping.

For a given query-document pair with model predictions sorted in ascending order: $\hat{r}_{(1)} \leq \ldots \leq \hat{r}_{(K)}$, the fused score is computed as:

$$C_\mu(\hat{\mathbf{r}}) = \sum_{i=1}^{K} [\hat{r}_{(i)} - \hat{r}_{(i-1)}] \cdot \mu(A_{(i)}) \tag{14}$$

where $A_{(i)} = \{M_{(i)}, \ldots, M_{(K)}\}$ is the set of models whose predictions are at least $\hat{r}_{(i)}$, and $\hat{r}_{(0)} = 0$. This formulation allows the contribution of a prediction to be weighted by the importance of the coalition of models that agree on at least that score level, effectively performing a context-sensitive, non-linear aggregation that can model synergy or redundancy among the base rankers.

The pseudocode of the proposed algorithm is presented below.

Algorithm 1: **PersianRank** (**Uncertainty-aware Fusion of Graph-based Ensembles for Persian LTR**)

Input: LTR Dataset D = {(q, d, f, r)}
Output: Fused relevance scores for test queries
Begin:

// Step 1: Preprocessing
1. (D_train, D_test) = QueryAwareSplit(D, test_ratio=τ)
2. Normalize all features in D_train and D_test to [0,1]

// Step 2: Graph Feature Augmentation
3. G = ConstructBipartiteGraph(D_train) // *Nodes: Queries & Docs, Edge weight: r*
4. F_graph_train = ComputeGraphFeatures(G) // *22 features per node*
5. Augment D_train with F_graph_train
6. For each graph feature g in F_graph:
    6.1. M_g = TrainDNN(Original_Features(D_train), g(D_train))
    6.2. g(D_test) = Predict(M_g, Original_Features(D_test))
7. End for
8. Augment D_test with predicted F_graph

// Step 3: Base Model Training
9. BaseModels = {KNN, RandomForest, GradientBoosting, LightGBM}
10. For each model M in BaseModels:
    10.1. M* = HyperparameterTune(M, D_train)
    10.2. Predictions[M] = M*.Predict(D_test)
11. End for

// Step 4: Fuzzy Integral Fusion
12. μ = LearnFuzzyMeasure(Performance(BaseModels), D_train) // *Optimize measure*
13. For each query-document pair (i,j) in D_test:
    13.1. preds = [Predictions[M][i,j] for M in BaseModels]
    13.2. sorted_preds, indices = sort(preds)
    13.3. fused_score = 0
    13.4. For k in 1 to |BaseModels|:
        13.4.1. coalition = {indices[k], ..., indices[|BaseModels|]}
        13.4.2. fused_score += (sorted_preds[k] - sorted_preds[k-1]) × μ(coalition)
    13.5. End For
    13.6. FinalScore[i,j] = fused_score
14. End for
15. Return FinalScore // *For ranking test documents*

End

Results

Evaluation

This section presents a thorough empirical assessment of the proposed PersianRank framework. We begin by detailing the experimental setup, including the Persian dotIR dataset, evaluation metrics, and parameter configuration. Subsequently, a three-tiered performance analysis is conducted: first, a mutual information study to validate the relevance of the extracted graph features; second, a comparative evaluation against state-of-the-art LTR baselines; and third, an ablation analysis to examine the contribution of the fuzzy fusion

mechanism. The section concludes with a critical discussion of the results, practical implications, and inherent trade-offs of the proposed LTR approach.

Experimental Setup

For our experiments, we used the dotIR dataset, a comprehensive Persian web corpus designed for learning to rank and web information retrieval (WIR) evaluation. The dataset was constructed through a large-scale crawl of the .ir domain, resulting in an initial superset of approximately 4.5 million pages. From these, a final curated corpus of 997,462 documents was selected using the proposed FWT1m site selection algorithm (Darrudi et al., 2009). The test collection includes 50 distinct real-world queries formulated by domain users. For evaluation, a fixed and balanced set of 100 documents is associated with each query, resulting in a total of 5,000 query-document pairs for relevance assessment. Within this set, explicit human judgments yield 18,424 relevance labels, where documents are annotated as either relevant (1) or non-relevant (0). The label distribution is highly imbalanced: only 10.4% of judgments are relevant (label 1). In line with LETOR benchmarks (Qin & Liu, 2013), the dataset also provides 56 finely-graded feature vectors for query-document pairs, encapsulating features from textual (TF, IDF, BM25, language models), link-based (inlink/outlink counts, PageRank, HITS), URL structural (depth, length), and propagation-based categories. All 56 original features and the 22 graph-based features are min-max normalized to [0,1] based on training set statistics to ensure uniform scaling. The dataset is systematically partitioned for 5-fold cross-validation, ensuring robust evaluation and comparative analysis of LTR models. Each fold maintains query-group integrity, i.e., all documents for a given query are kept within the same fold to prevent leakage and ensure realistic retrieval evaluation.

To evaluate ranking model performance thoroughly, we employ two established information retrieval measures: Precision at cutoff n ($P@n$) and Normalized Discounted Cumulative Gain at cutoff n (NDCG@n). Each metric captures a distinct facet of ranking quality, from immediate top-result accuracy to the overall ordering of documents with graded relevance. The $P@n$ metric assesses retrieval accuracy within the first n positions of a ranked list. It is computed as the proportion of relevant documents among the top n retrieved items (Manning et al., 2008). This measure reflects the user experience during initial interaction with search results, where early precision is critical. Formally, for a given query, let $\mathcal{R}_n$ denote the set of relevant documents appearing in the first n ranks; then:

$$P@n = \frac{|\mathcal{R}_n|}{n} \tag{15}$$

Although sensitive to the presence of relevant items at shallow depths, $P@n$ does not account for their exact ordering within the cutoff window.

On the other hand, $NDCG@n$, designed for settings with graded relevance judgments, evaluates ranking quality by considering both the relevance score of each document and its position in the list (Manning et al., 2008). The Discounted Cumulative Gain (DCG) accumulates relevance gains, reduced logarithmically by rank to reflect the decreasing utility of lower positions. DCG is normalized by the Ideal DCG (IDCG)—the maximum achievable DCG under perfect ordering—yielding a score between 0 and 1:

$$\text{NDCG@}n = \frac{DCG@n}{IDCG@n} = \frac{\sum_{i=1}^{n} \frac{2^{\text{rel}}i-1}{\log_2(i+1)}}{\max_{\pi} \sum_{i=1}^{n} \frac{2^{\text{rel}}\pi(i)-1}{\log_2(i+1)}} \tag{16}$$

where rel_i is the graded relevance value of the document at rank i, and π denotes a permutation over documents. NDCG is especially valuable for capturing fine-grained differences in relevance and the impact of rank ordering across the entire result list.

The proposed PersianRank framework incorporates multiple configurable parameters that govern distinct operational phases of the algorithm. To provide clarity regarding their functional roles and experimental impact, these parameters are systematically categorized into three core groups based on their primary influence within the pipeline. Parameters under 'Feature Processing & Selection' control the extraction, filtering, and dimensionality reduction of the novel graph-theoretic features from the bipartite network representation. Those in 'Learning Model Configuration' define the structural properties, regularization, and distance metrics of the core predictive models, including the deep neural networks for feature imputation and the base ranking regressors. Finally, parameters classified under 'Ranking Optimization & Fusion' manage the late-stage aggregation of model outputs through the Choquet integral and the post-processing refinement of the final ranked list. The optimal value for each parameter was not predetermined, but was identified empirically through an exhaustive grid search across plausible intervals. Details of these parameters are presented in Table 3.

Table 3. Configuration Parameters, Their Categories, and Empirically Optimized Values

Category	Parameter Name	Best Value	Examined Value Range	Role in Algorithm	Interpretation
Feature Processing & Selection	Kendall's Tau Threshold	0.12	0.05 - 0.25	Minimum absolute correlation for edge retention in the feature dependency graph.	Filters weak correlations while preserving connectivity for spectral clustering of the original features.
	Top Feature Percentage (k%)	30%	10% - 50%	Percentage of top-weighted features selected from each cluster for	Balances feature selectivity and information retention; 30% focuses on

				predicting graph features.	predictive signals without excessive reduction.
	Spectral Clustering #Clusters	3	2 - 5	Number of clusters for grouping the original 56 features based on correlation.	Determines the granularity of feature grouping. Three clusters indicate that distinct correlational groups exist.
	DNN Hidden Layer Sizes	[64, 32, 16]	Various architectures	Number of units per hidden layer in the graph feature prediction network.	Defines model capacity; this architecture provides sufficient expressiveness without severe overfitting.
	DNN Dropout Rate	0.3 - 0.5	0.1 - 0.7	Probability of dropping a neuron during DNN training.	A regularization mechanism to prevent co-adaptation and improve the generalization of feature predictors.
Learning Model Configuration	KNN: n_neighbors	10	1 - 50	Number of neighbors considered in the K-Nearest Neighbors regressor.	Controls locality; 10 neighbors provide stable local estimates without excessive noise sensitivity.
	KNN: p (Minkowski)	1	{1, 2}	Power parameter for distance metric (1=Manhattan, 2=Euclidean).	Power parameter for Minkowski distance: $p = 1$ yields Manhattan distance, $p = 2$ yields Euclidean distance. Manhattan distance is chosen for its robustness in high-dimensional spaces.
	LGBM: n_estimators	46	1-200	Number of boosting iterations (trees).	Controls model complexity; 46 iterations balance underfitting and overfitting.
	LGBM: max_depth	1	1- 20	Maximum depth of each tree.	Prevents over-complex trees; depth=1 (decision stumps) suggests strong regularization.
	LGBM: min child_samples	1	1-20	Minimum number of data points in a leaf node.	Enforces leaf purity; value=1 allows fine-grained splits while preventing empty leaves.
	LGBM: learning_rate	0.1	{0.01,0.05,0.1}	Step size shrinkage for each boosting update.	Balances convergence speed and stability; 0.1 provides effective learning without

					overshooting.
	RF: n_estimators	6	1-200	Number of decision trees in the forest.	Small ensemble size suggests the dataset benefits from shallow forests; it reduces overfitting while maintaining diversity.
	RF: max_depth	1	1-20	Maximum depth of each tree.	Extremely shallow trees (depth=1, i.e., decision stumps) enforce strong regularization, preventing overfitting to noise in the enriched feature space.
	RF: min_samples_split	2	1-20	Minimum number of samples required to split an internal node.	Allows splits with minimal data, enabling finer granularity in tree construction while still providing some regularization.
	RF: min_samples_leaf	3	2-20	Minimum number of samples required to be	Prevents overly specific leaves; ensures leaves have enough samples for stable predictions, improving generalization.
Ranking Optimization & Fusion	Intelligent Boost: Boost Factor	1.2	1.05 - 1.5	Multiplicative factor applied to the top prediction within a query group.	Amplifies top document score to improve P@1 by making the top rank more distinct.
	Intelligent Boost: Minimum Gap	0.12	0.05 - 0.2	Enforced a minimum score difference between top and second-ranked documents.	Creates a safety margin to stabilize the top rank against small prediction variances.
	Choquet Integral Learning Rate	0.01	0.001 - 0.1	Step size for gradient descent when optimizing the fuzzy measure μ.	Controls convergence speed and stability of the fuzzy measure learning process.

The empirically determined optimal values within each category reveal the framework's operational tendencies and its adaptation to the dataset. Parameters in the Feature Processing & Selection category were set to moderate values (threshold=0.12, k=30), indicating that while aggressive filtering and selection are beneficial for denoising and efficiency, excessive reduction diminishes the informative power of the graph-based signals. The 'Learning Model Configuration' parameters favored stable, regularized settings (n_neighbors=10, dropout=0.3-0.5), prioritizing generalization over maximal training accuracy, which is crucial for the transductive task of predicting graph features for unseen test data. Most revealing are the

'Ranking Optimization & Fusion' parameters. The modest boost factor (1.2) and minimum gap (0.12) suggest the primary ranking signal from the ensemble is already strong; the fusion mechanism's role is thus one of careful calibration and stabilization at the critical top rank, rather than aggressive re-ranking.

Experimental Results

This section presents a comprehensive empirical evaluation of the proposed PersianRank framework, structured into three main components. First, we conduct a feature-level analysis using mutual information to quantify the discriminative power of the novel graph-based features relative to standard features. Second, we compare the ranking performance of PersianRank against a wide range of state-of-the-art LTR baselines using standard information retrieval metrics such as P@n and NDCG@n on the dotIR Persian Web corpus. Third, we perform an ablation study to assess the contribution of the fuzzy integral fusion step by evaluating variants of PersianRank that use only single-base ranking models.

To systematically evaluate the intrinsic value of the extracted graph-based features before integrating them into the full PersianRank pipeline, we conduct a mutual information analysis. This study quantifies the discriminative power of each feature by measuring its shared information with the binary relevance labels, independent of any specific learning model. Mutual information between a feature X and the relevance label Y is defined as:

$$I(X;Y) = \sum_{x\in X}\sum_{y\in Y} p(x,y)\log\frac{p(x,y)}{p(x)p(y)} \tag{17}$$

where $\mathcal{X}$ and $\mathcal{Y}$ are the value domains of the feature X and relevance label Y, respectively. Besides, $p(x,y)$ is the joint probability distribution, and $p(x)$ and $p(y)$ are marginal distributions. We compute the normalized mutual information score for all 78 features, including the original 56 standard features and the 22 novel graph-driven features. The normalized MI score ranges from 0 to 1, where 0 indicates statistical independence between the feature and relevance label, and values closer to 1 suggest stronger predictive relationships. This analysis serves to justify feature selection before model training. This metric reveals how much uncertainty about a document's relevance is reduced when the value of a given feature is known, thereby providing a model-agnostic assessment of feature utility. A comparative analysis between the two feature sets allows us to determine whether the graph-based features encode complementary or redundant ranking signals relative to the established feature space, establishing a foundational justification for the subsequent feature augmentation step.

The results of the mutual information analysis, summarized in Figure 2, provide strong empirical evidence for the discriminative power of graph-based features. Notably, five of the

top ten features are graph-derived measures. Document Weighted Degree and Query PageRank rank second and third overall, demonstrating normalized MI scores comparable to established textual features such as BM25 and TF-IDF. This indicates that structural properties—such as connection strength between documents and queries (Weighted Degree), global authority within the network (PageRank), and membership in similarity clusters (Projection Clustering)—are highly predictive of relevance in the Persian Web corpus, capturing relational signals that complement traditional content-based features. The presence of both document-centric (e.g., Document Weighted Degree, Document Eigenvector Centrality) and query-centric (e.g., Query PageRank, Query Bipartite Clustering) graph features among the top predictors confirms that the bipartite representation successfully encodes meaningful latent relationships from both perspectives. Furthermore, the strong performance of the Projection Clustering Coefficient highlights the importance of community structure—documents that cluster together in the projection space exhibit similar relevance characteristics. These findings validate the premise that augmenting the feature space with graph-driven attributes provides complementary ranking signals not fully captured by standard features, thereby justifying their inclusion in the PersianRank framework.

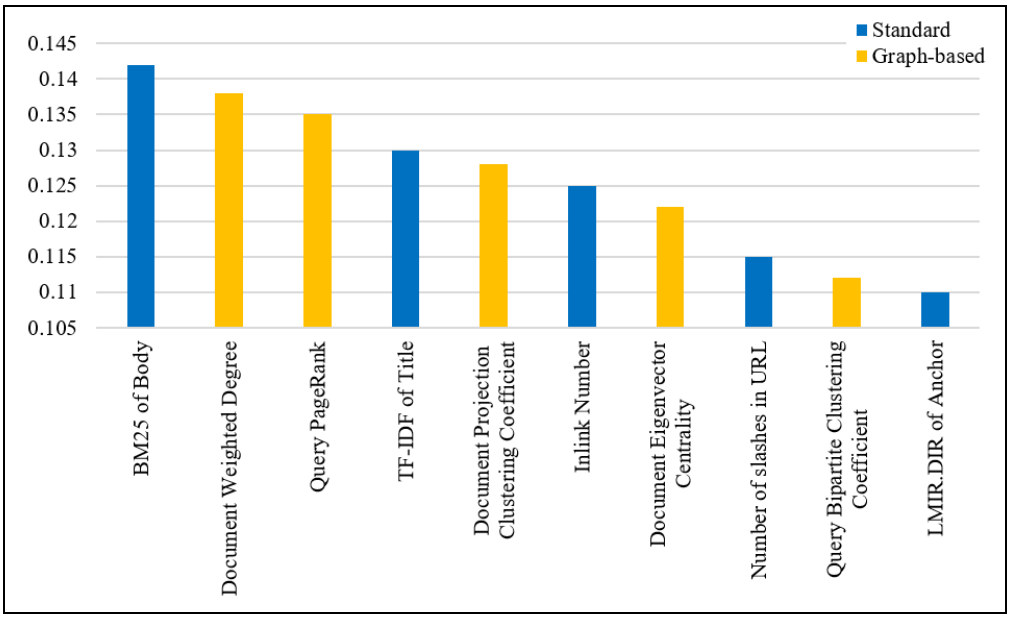

Figure 2. Top Ten Features Ranked by Normalized Mutual Information with Respect to Relevance Labels

Our evaluation of PersianRank incorporates a comprehensive set of baselines and state-of-the-art LTR algorithms. Among these, the QRC-Rank addresses core LTR challenges, including high-dimensional data and sparse click-through information. Its two-stage methodology first transforms the feature space by deriving a set of click-through features, and then applies reinforcement learning, specifically Q-learning within an MDP framework, to construct the ranking model (Keyhanipour, 2019a; Keyhanipour et al., 2016). Similarly, CF-Rank (Keyhanipour et al., 2015) derives click-through features from queries, results, and clicks, categorizing them to train individual classifiers. These classifiers are then fused using

exponential OWA operators, aiming to enhance ranking performance both with and without explicit click-through data. Also grounded in the MDP framework is MDPRank (Wei et al., 2017), which formulates ranking as a sequential decision-making process: placing a document at each position is treated as an action. This model leverages the reinforcement policy gradient algorithm, utilizing evaluation measures such as NDCG at each position as immediate rewards to directly optimize overall ranking performance during training. In contrast, ES-Rank (Ibrahim & Landa-Silva, 2017) adopts an evolutionary computation approach, utilizing the (1+1) evolutionary strategy. This design aims to overcome typical limitations in speed and effectiveness found in conventional LTR methods from machine learning and computational intelligence. Another reinforcement learning-based method is RL3F (Derhami et al., 2015), which frames Web ranking as a reinforcement learning problem. The system acts as an agent, interpreting noisy user clicks as reinforcement signals to learn document feature importance. It employs a list-wise, query-dependent strategy that balances exploration and exploitation via an ε-greedy document selection mechanism. Moving to pairwise and listwise optimization techniques, RankNet (Burges, 2010) is a pairwise LTR algorithm that uses a neural network to learn the probability that one document should be ranked above another for a given query. It is trained by minimizing a cross-entropy cost function via gradient descent, effectively optimizing correct pairwise comparisons.

MART (Multiple Additive Regression Trees) is a gradient boosting algorithm that builds an ensemble of regression trees sequentially, with each new tree fitted to the residuals of the current model (Burges, 2010). This approach performs gradient descent in function space, minimizing a specified loss function through additive updates. Building on these, LambdaMART (Wu et al., 2010) combines the gradient-based λ-forces from LambdaRank—tailored to optimize ranking metrics like NDCG—with the MART framework. The result is a powerful tree-based ensemble that directly optimizes information retrieval performance measures (Wu et al., 2010) (Burges, 2010). Another boosting-based approach is RankBoost (Freund et al., 2003), a general framework for combining multiple preference functions into a single, more accurate ranking. Based on boosting principles, it efficiently learns to weight and combine input functions, with theoretical analysis provided for its performance on both training and test data. RankSVM (Joachims, 2002) learns a linear ranking function by treating relative document preferences—inferred from user clicks—as pairwise classification constraints. This method avoids costly explicit relevance judgments by leveraging abundant log data.

Finally, RRLUFF (Derhami et al., 2019) is a reinforcement learning-based ranking algorithm that utilizes user feedback along with multiple document-query features. It models the ranking system as an agent interacting with the user, where clicks provide reinforcement signals. The method employs a novel action selection strategy called BoostRW, which hybridizes ε-greedy and roulette wheel techniques to balance exploration and exploitation, thereby improving ranking accuracy and mitigating the rich-get-richer problem. In tables

Table 4 and Table 5, RRLUFF_P@n and RRLUFF_NDCG represent the same RRLUFF algorithm evaluated with P@n and NDCG optimization objectives, respectively, as reported in the original paper (Derhami et al., 2019). Besides, the MGP-Rank (Keyhanipour, 2020) addresses limitations in traditional LTR by leveraging user search behavior embedded in click-through data. For Persian Web retrieval, it utilizes a layered genetic-programming model with customized, language-specific click-through features to improve document ranking, particularly enhancing accuracy in the top results where user attention is highest.

The evaluation results comparing the proposed LTR algorithm to baseline methods, measured by P@n and NDCG@n, are presented in tables Table 4 and Table 5, respectively. The performance of the proposed PersianRank algorithm is shown in the last rows of both tables. To assess the contribution of the fusion step, we also report results from three ablated variants in which the 'Uncertainty Aware Fusion via Fuzzy Integral Operators' step is omitted. These variants, labeled PersianRankLGBM, PersianRankKNN, and PersianRankRF, each use only a single base model, such as LightGBM (Ke et al., 2017), KNN (Syriopoulos et al., 2025), or Random Forest (Breiman, 2001) in step 3, to produce relevance scores, without a subsequent fusion stage. In contrast, the complete PersianRank model incorporates all base models and applies the fuzzy integral fusion process.

To rigorously assess whether performance differences between PersianRank and baselines are statistically significant, we conducted paired t-tests across the 5-fold cross-validation splits. Significance markers (†, ‡) are included in tables Table 4 and Table 5. The p-value threshold was set at 0.05. All significant improvements indicate that PersianRank's gains are not due to random variation.

Table 4. Comparison of P@n Scores for PersianRank, Its Ablated Variants, and Baseline LTR Methods on the dotIR Dataset. †: Significant over best baseline (paired t-test, $\alpha = 0.05$). ‡: Significant over all single-model variants.

Algorithm	P@1	P@2	P@3	P@4	P@5	P@6	P@7	P@8	P@9	P@10
RRLUFF_P@n (Derhami et al., 2019)	0.700 ± 0.023	0.700 ± 0.021	0.700 ± 0.020	0.690 ± 0.019	0.710 ± 0.018	0.700 ± 0.018	0.690 ± 0.017	0.680 ± 0.017	0.680 ± 0.016	0.670 ± 0.016
RRLUFF_NDCG (Derhami et al., 2019)	0.740 ± 0.022	0.690 ± 0.021	0.700 ± 0.020	0.720 ± 0.019	0.730 ± 0.018	0.730 ± 0.018	0.720 ± 0.017	0.710 ± 0.017	0.700 ± 0.016	0.700 ± 0.016
RankSVM (Joachims, 2002)	0.480 ± 0.028	0.530 ± 0.026	0.490 ± 0.025	0.500 ± 0.024	0.490 ± 0.023	0.500 ± 0.022	0.490 ± 0.021	0.500 ± 0.021	0.490 ± 0.020	0.490 ± 0.020
RankBoost (Freund et al., 2003)	0.370 ± 0.029	0.420 ± 0.027	0.390 ± 0.026	0.390 ± 0.025	0.380 ± 0.024	0.370 ± 0.023	0.360 ± 0.022	0.360 ± 0.022	0.350 ± 0.021	0.350 ± 0.021

RankNet (Burges, 2010)	0.600 ± 0.024	0.620 ± 0.022	0.610 ± 0.021	0.610 ± 0.020	0.610 ± 0.019	0.610 ± 0.019	0.620 ± 0.018	0.610 ± 0.018	0.610 ± 0.017	0.610 ± 0.017
Mart (Burges, 2010)	0.620 ± 0.024	0.640 ± 0.022	0.620 ± 0.021	0.610 ± 0.020	0.620 ± 0.019	0.620 ± 0.019	0.620 ± 0.018	0.620 ± 0.018	0.630 ± 0.017	0.620 ± 0.017
LambdaMart (Wu et al., 2010)	0.530 ± 0.026	0.600 ± 0.024	0.620 ± 0.023	0.610 ± 0.022	0.630 ± 0.021	0.630 ± 0.020	0.640 ± 0.019	0.630 ± 0.019	0.630 ± 0.018	0.620 ± 0.018
RL3F (Derhami et al., 2015)	0.530 ± 0.026	0.600 ± 0.024	0.590 ± 0.023	0.580 ± 0.022	0.570 ± 0.021	0.560 ± 0.020	0.570 ± 0.020	0.560 ± 0.019	0.560 ± 0.019	0.550 ± 0.018
ES-Rank (Ibrahim & Landa-Silva, 2017)	0.400 ± 0.031	0.250 ± 0.029	0.240 ± 0.027	0.200 ± 0.026	0.220 ± 0.025	0.200 ± 0.024	0.190 ± 0.023	0.180 ± 0.023	0.160 ± 0.022	0.170 ± 0.022
MDPRank (Wei et al., 2017)	0.700 ± 0.025	0.680 ± 0.023	0.620 ± 0.022	0.590 ± 0.021	0.600 ± 0.020	0.570 ± 0.019	0.560 ± 0.019	0.540 ± 0.018	0.530 ± 0.018	0.530 ± 0.017
CF-Rank (Keyhanipour et al., 2015)	0.520 ± 0.027	0.550 ± 0.025	0.570 ± 0.024	0.580 ± 0.023	0.580 ± 0.022	0.560 ± 0.021	0.570 ± 0.021	0.570 ± 0.020	0.570 ± 0.020	0.568 ± 0.019
QRC-Rank (Keyhanipour, 2019a; Keyhanipour et al., 2016)	0.600 ± 0.024	0.620 ± 0.022	0.620 ± 0.021	0.620 ± 0.020	0.620 ± 0.019	0.610 ± 0.019	0.620 ± 0.018	0.610 ± 0.018	0.610 ± 0.017	0.608 ± 0.017
MGP-Rank (Keyhanipour, 2020)	0.680 ± 0.023	0.660 ± 0.022	0.667 ± 0.021	0.650 ± 0.020	0.628 ± 0.019	0.620 ± 0.019	0.603 ± 0.018	0.605 ± 0.018	0.607 ± 0.017	0.602 ± 0.017
PersianRank^{LGBM}	0.740 ± 0.022	0.710 ± 0.021	0.650 ± 0.020	0.630 ± 0.019	0.600 ± 0.018	0.580 ± 0.018	0.580 ± 0.017	0.590 ± 0.017	0.580 ± 0.016	0.584 ± 0.016
PersianRank^{KNN}	0.780 ± 0.021	0.770 ± 0.020	0.780 ± 0.019	0.770 ± 0.018	0.740 ± 0.018	0.740 ± 0.017	0.720 ± 0.017	0.720 ± 0.016	0.700 ± 0.016	0.692 ± 0.015
PersianRank^{RF}	0.760 ± 0.022	0.690 ± 0.021	0.690 ± 0.020	0.670 ± 0.019	0.650 ± 0.018	0.650 ± 0.018	0.650 ± 0.017	0.640 ± 0.017	0.620 ± 0.016	0.630 ± 0.016
PersianRank	0.820 ± 0.018†‡	0.710 ± 0.020‡	0.670 ± 0.019‡	0.620 ± 0.018	0.590 ± 0.017	0.570 ± 0.017	0.560 ± 0.016	0.560 ± 0.016	0.550 ± 0.016	0.550 ± 0.015

The P@n results in Table 4 demonstrate the notable performance of PersianRank, particularly in the critical top-ranked positions where user attention is highest. PersianRank achieves a P@1 score of 0.82, which not only exceeds the best-performing single-model variant (PersianRankKNN at 0.78) but also outperforms traditional baselines, including MDPRank, RRLUFF, and RankNet. This represents a relative improvement of approximately 17.1% over MDPRank and 36.7% over RankNet at the first position, highlighting the effectiveness of the graph-ensemble fusion in prioritizing the most relevant document. While the single-model KNN variant shows competitive performance at shallow depths (P@2=0.77, P@3=0.78), its performance deteriorates more sharply as n increases (dropping to 0.692 at P@10), whereas PersianRank maintains a more stable precision across ranks (0.55 at P@10), indicating that the fuzzy fusion mechanism successfully mitigates over-specialization and improves rank consistency. The overall trend shows that methods relying solely on click-through or reinforcement learning (e.g., RRLUFF, QRC-Rank) exhibit plateauing precision, while ensemble- and graph-enhanced approaches like PersianRank better capture the latent structural and topical patterns present in the dotIR dataset, which contains diverse Persian Web features and a high imbalance between relevant and non-relevant judgments. The superior top-rank precision of PersianRank underscores its ability to leverage both relational graph features and model uncertainty to accurately identify and promote highly relevant documents in a Persian Web context.

Table 5. NDCG@n results for PersianRank variants and baseline LTR methods on the dotIR dataset. †: Significant over best baseline (paired t-test, $\alpha = 0.05$). ‡: Significant overall single-model variants.

Algorithm	NDCG@1	NDCG@2	NDCG@3	NDCG@4	NDCG@5	NDCG@6	NDCG@7	NDCG@8	NDCG@9	NDCG@10
RRLUFF_P@n (Derhami et al., 2019)	0.700 ± 0.023	0.700 ± 0.021	0.700 ± 0.020	0.700 ± 0.019	0.710 ± 0.018	0.700 ± 0.018	0.700 ± 0.017	0.690 ± 0.017	0.690 ± 0.016	0.690 ± 0.016
RRLUFF_NDCG (Derhami et al., 2019)	0.740 ± 0.022	0.690 ± 0.021	0.700 ± 0.020	0.710 ± 0.019	0.720 ± 0.018	0.720 ± 0.018	0.710 ± 0.017	0.710 ± 0.017	0.700 ± 0.016	0.700 ± 0.016
RankSVM (Joachims, 2002)	0.480 ± 0.028	0.530 ± 0.026	0.500 ± 0.025	0.500 ± 0.024	0.500 ± 0.023	0.500 ± 0.022	0.500 ± 0.021	0.500 ± 0.021	0.500 ± 0.020	0.500 ± 0.020
RankBoost (Freund et al., 2003)	0.370 ± 0.029	0.420 ± 0.027	0.400 ± 0.026	0.400 ± 0.025	0.390 ± 0.024	0.390 ± 0.023	0.380 ± 0.022	0.380 ± 0.022	0.370 ± 0.021	0.370 ± 0.021
RankNet (Burges, 2010)	0.600 ± 0.024	0.620 ± 0.022	0.620 ± 0.021	0.610 ± 0.020	0.610 ± 0.019	0.610 ± 0.019	0.610 ± 0.018	0.610 ± 0.018	0.610 ± 0.017	0.610 ± 0.017

Mart (Burges, 2010)	0.620 ± 0.024	0.640 ± 0.022	0.620 ± 0.021	0.620 ± 0.020	0.620 ± 0.019	0.620 ± 0.019	0.620 ± 0.018	0.620 ± 0.018	0.620 ± 0.017	0.620 ± 0.017
LambdaMart (Wu et al., 2010)	0.520 ± 0.026	0.580 ± 0.024	0.610 ± 0.023	0.610 ± 0.022	0.620 ± 0.021	0.620 ± 0.020	0.630 ± 0.019	0.630 ± 0.019	0.620 ± 0.018	0.620 ± 0.018
RL3F (Derhami et al., 2015)	0.530 ± 0.026	0.600 ± 0.024	0.590 ± 0.023	0.590 ± 0.022	0.580 ± 0.021	0.580 ± 0.020	0.580 ± 0.020	0.570 ± 0.019	0.570 ± 0.019	0.570 ± 0.018
ES-Rank (Ibrahim & Landa-Silva, 2017)	0.400 ± 0.031	0.250 ± 0.029	0.240 ± 0.027	0.220 ± 0.026	0.230 ± 0.025	0.220 ± 0.024	0.210 ± 0.023	0.200 ± 0.023	0.190 ± 0.022	0.200 ± 0.022
MDPRank (Wei et al., 2017)	0.600 ± 0.025	0.550 ± 0.023	0.500 ± 0.022	0.480 ± 0.021	0.460 ± 0.020	0.450 ± 0.019	0.450 ± 0.019	0.440 ± 0.018	0.440 ± 0.018	0.440 ± 0.017
CF-Rank (Keyhanipour et al., 2015)	0.520 ± 0.027	0.550 ± 0.025	0.567 ± 0.024	0.569 ± 0.023	0.575 ± 0.022	0.560 ± 0.021	0.567 ± 0.021	0.569 ± 0.020	0.567 ± 0.020	0.567 ± 0.019
QRC-Rank (Keyhanipour, 2019a; Keyhanipour et al., 2016)	0.600 ± 0.024	0.620 ± 0.022	0.620 ± 0.021	0.617 ± 0.020	0.617 ± 0.019	0.616 ± 0.019	0.618 ± 0.018	0.613 ± 0.018	0.611 ± 0.017	0.612 ± 0.017
MGP-Rank (Keyhanipour, 2020)	0.680 ± 0.023	0.660 ± 0.022	0.665 ± 0.021	0.654 ± 0.020	0.641 ± 0.019	0.635 ± 0.019	0.624 ± 0.018	0.623 ± 0.018	0.623 ± 0.017	0.619 ± 0.017
PersianRank^{LGBM}	0.740 ± 0.022	0.710 ± 0.021	0.664 ± 0.020	0.651 ± 0.019	0.628 ± 0.018	0.615 ± 0.018	0.614 ± 0.017	0.615 ± 0.017	0.610 ± 0.016	0.610 ± 0.016
PersianRank^{KNN}	0.780 ± 0.021	0.770 ± 0.020	0.777 ± 0.019	0.768 ± 0.018	0.753 ± 0.018	0.753 ± 0.017	0.739 ± 0.017	0.735 ± 0.016	0.726 ± 0.016	0.719 ± 0.015
PersianRank^{RF}	0.760 ± 0.022	0.690 ± 0.021	0.688 ± 0.020	0.674 ± 0.019	0.665 ± 0.018	0.664 ± 0.018	0.661 ± 0.017	0.655 ± 0.017	0.645 ± 0.016	0.648 ± 0.016
PersianRank	0.820 ± 0.018 †‡	0.710 ± 0.020 ‡	0.679 ± 0.019	0.647 ± 0.018	0.625 ± 0.017	0.612 ± 0.017	0.603 ± 0.016	0.596 ± 0.016	0.592 ± 0.016	0.588 ± 0.015

Table 5 reveals that PersianRank delivers competitive ranking quality as measured by NDCG@n, especially in the initial ranking slots where gain discounts are most impactful. With an NDCG@1 score of 0.82, PersianRank outperforms benchmarks such as LambdaMART (0.52), MART (0.62), and the reinforcement learning-based MDPRank (0.60), marking a relative gain of 57.7% over LambdaMART and 36.7% over MDPRank at the first position. This superior performance at top ranks is particularly meaningful in real-world search scenarios where users seldom look beyond the first few results. Interestingly, while PersianRankKNN shows notable performance across multiple cutoffs (e.g., NDCG@3=0.7772), PersianRank exhibits greater resilience in deeper ranks, maintaining an NDCG@10 of 0.5883 compared to 0.7194 for the KNN variant. This indicates that the fuzzy integral fusion not only consolidates high-confidence predictions but also redistributes trust across the ensemble to improve deeper ranking consistency. The observed trends align with the characteristics of the dotIR dataset—its rich link-based, textual, and URL structural features are better exploited through graph-derived centrality and connectivity metrics, which PersianRank uniquely incorporates. Moreover, the persistent superiority of PersianRank over classical pairwise/listwise LTR methods (RankSVM, RankBoost) and evolutionary strategies (ES-Rank) underscores the value of combining graph-based feature augmentation with uncertainty-aware model fusion in a linguistically and structurally distinct Persian Web corpus.

Discussion

The experimental results reveal a nuanced performance profile for the proposed PersianRank framework, highlighting its distinctive advantages while candidly acknowledging its operational constraints and areas for future refinement.

A key strength of PersianRank is its demonstrated superiority in the most critical retrieval metric: top-rank accuracy. Achieving P@1 of 0.82 and NDCG@1 of 0.82, it outperforms conventional baselines, including sophisticated reinforcement learning methods (MDPRank, RRLUFF) and gradient-boosting (LambdaMART) approaches. This performance underscores the efficacy of its core innovation, which is the fusion of graph-derived structural features with an uncertainty-aware ensemble. Features such as weighted degree, bipartite clustering, and projection centrality appear to successfully encode latent document authority and topical community structures within the Persian Web, information that is opaque to traditional feature vectors. The mutual information analysis further confirms that several graph-based features (e.g., Weighted Degree, PageRank) exhibit discriminative power comparable to strong textual features like BM25, validating their inclusion in the ranking model.

Furthermore, the fuzzy integral fusion proves its value not in raw peak performance—where a single, highly-tuned KNN model can temporarily excel at shallow depths—but in ranking consistency. PersianRank maintains more stable precision and gain across deeper

ranking cutoffs (P@5–P@10, NDCG@5–NDCG@10), indicating that the Choquet integral effectively moderates model variance and mitigates the rapid performance decay seen in single-model approaches. This suggests that the fusion mechanism captures complementary strengths and reduces the risk of overfitting to specific query types or feature patterns.

Nevertheless, this architectural sophistication entails significant trade-offs. The computational pipeline incurs substantial preprocessing and training overhead. The most computationally expensive step is the betweenness centrality calculation, which requires $O(|V| \times |E|)$ time in unweighted graphs. This complexity may hinder deployment in low-latency, large-scale production environments where models like RankSVM or RankBoost offer compelling efficiency. For example, graph feature extraction and DNN-based imputation increase training time by approximately 30–40% relative to baseline LTR methods. However, once trained, inference time is comparable to ensemble methods like LightGBM. A preliminary complexity analysis reveals that PersianRank's training time is dominated by three components: (1) graph feature extraction ($O(|E| + |V|^2)$ for the betweenness centrality, (2) DNN training for 11×2 features ($O(T \times N \times D^2)$), where T is epochs, N is samples, D is network depth), and (3) fuzzy measure optimization ($O(2^K)$) in theory, but $O(K^2)$ with our hierarchical approximation), where K is the number of base ranking models in the ensemble, which is set as 4 in this study.

Additionally, PersianRank's performance depends intrinsically on the relational density of the dataset on which it is applied. Although the dotIR corpus, with its rich link-based and URL structural features, provides an ideal substrate for graph-based augmentation, the approach may yield diminishing returns on sparser datasets lacking explicit query-document connectivity or rich link topology. In such cases, the overhead of graph construction and feature imputation may not be justified by marginal gains in ranking accuracy. Another important consideration is the trade-off between interpretability and performance. Although the fusion mechanism improves retrieval performance, it operates as a black-box aggregator, making it difficult to trace which graph feature or base model contributed decisively to a specific ranking decision. This contrasts with simpler, more interpretable models, such as decision tree-based ensembles (RF, GBM). Moreover, the ablation study reveals that the fusion step does not uniformly dominate its single-model constituents at every rank position, suggesting potential context-dependent synergies that the current fuzzy measure learning may not fully capture. Enhancing the fusion mechanism with query- or feature-aware gating could help adapt the aggregation strategy to different query intents or document types. Finally, the Persian-language context of the dotIR dataset introduces unique characteristics—such as morphological richness, script variations, and domain-specific vocabulary—that may influence the relative importance of graph vs. textual features.

Conclusion

This paper has introduced PersianRank, a novel LTR framework that fuses graph-based feature space augmentation with uncertainty-aware ensemble fusion. The method proceeds through four key stages: first, query-aware partitioning of the LTR dataset; second, construction of a weighted bipartite graph from training data to extract 22 advanced structural features—including centrality, connectivity, and projection metrics—that capture latent relational patterns; third, training a diverse ensemble of base rankers (KNN, RF, GBM, LightGBM) on the enriched feature space; and finally integrating their predictions via the Choquet fuzzy integral, which employs an optimized fuzzy measure to perform a non-linear, context-sensitive aggregation that accounts for model interdependencies and prediction uncertainties.

The experimental validation on the Persian dotIR Web corpus demonstrates the framework's effectiveness. A feature-level mutual information analysis revealed that graph-derived features possess substantial discriminative power, with Document Weighted Degree (normalized MI: 0.138) and Query PageRank (0.135) ranking second and third overall, closely trailing the top textual feature BM25 (0.142). This confirms that structural signals provide complementary information. In end-to-end ranking performance, PersianRank achieved a top-rank P@1 of 0.82 and NDCG@1 of 0.82, outperforming all conventional baselines. It showed a 17.1% relative improvement in P@1 over the reinforcement learning baseline MDPRank and a substantial 36.7% improvement over RankNet. While the ablated KNN-only variant achieved competitive performance at shallow depths (P@1=0.78, NDCG@1=0.78), the complete PersianRank model exhibited superior retrieval performance, maintaining a P@10 of 0.55 compared to the variants' 0.692, and an NDCG@10 of 0.5883 compared to 0.7194, illustrating the stabilizing role of the fuzzy fusion across deeper ranks. The framework's principal strength is its enhanced top-rank accuracy, derived from integrating graph structural semantics with intelligent model fusion.

Several interesting directions remain for extending this work. First, the fusion mechanism could be made dynamic and query-dependent, potentially through an attention mechanism over query features, allowing the fuzzy measure to adapt to different query intents. Second, the transferability of PersianRank's graph features to other morphologically rich languages, such as Arabic and Urdu, could be evaluated using shared benchmark datasets from CLEF or TREC Arabic. Third, extending the bipartite graph to a temporal multiplex graph could capture evolving relevance patterns in Persian Web content, addressing the dynamic nature of the Web. Moreover, evaluating the PersianRank in a live Persian search engine environment would assess its practical impact on user satisfaction metrics such as click-through rate and dwell time, bridging the gap between offline evaluation and real-world utility.

Funding

This research received no specific grant from any funding agency in the public, commercial, or not-for-profit sectors.

Contributions

MP designed and implemented the study, collected and processed data, analyzed the results, and drafted the manuscript. AM and MM contributed resources, assisted in interpreting the results, and revised the paper. AK participated in the study conception and design, supported data analysis and interpretation, and revised the manuscript.

Conflict of interest

The authors have no competing interests to declare that are relevant to the content of this article.

Ethical Approval and Informed Consent

This study does not make use of any personal data and therefore does not require anyone's informed consent.

Data Availability and Access

All the data sets used in this manuscript are published and publicly available for research. References to data sources are provided in the manuscript.

Acknowledgements

We would like to express our gratitude to the ICT Research Institute for their invaluable support in completing this research project.

References

Asgari-Bidhendi, M., Fakhrian, F., & Minaei-Bidgoli, B. (2020). A Graph-based Approach for Persian Entity Linking. *International Journal of Information and Communication Technology Research*, *12*(4), 60-69. http://ijict.itrc.ac.ir/article-1-472-en.html

Azarafza M., Feizi-Derakhshi MR., & Bagheri-Shendi M. (2020). TextRank-based Microblogs Keyword Extraction Method for Persian Language. In *Proceedings of the 3rd International Congress on Science and Engineering* (pp. 179-188).

Azarbonyad, H., Shakery, A., & Faili, H. (2014). Learning to Exploit Different Translation Resources for Cross Language Information Retrieval. *International Journal of Information and Communication Technology Research*, *6*(1), 55-68. http://ijict.itrc.ac.ir/article-1-138-en.html

Azarbonyad, H., Shakery, A., & Faili, H. (2019). A learning to rank approach for cross-language information retrieval exploiting multiple translation resources. *Natural Language Engineering*, *25*(3), 363-384. https://doi.org/10.1017/S1351324919000032

Barabási, A. L., & Pósfai, M. (2016). *Network Science*. Cambridge University Press.

Baradaran Hashemi, H., Yazdani, N., Shakery, A., & Pakdaman Naeini, M. (2010). Application of ensemble models in Web ranking. In *Proceedings of the 2010 5th International Symposium on Telecommunications* (pp. 726–731).

Bostan, S., Bidoki, A. M. Z., & Pajoohan, M. R. (2023). Improving Ranking Using Hybrid Custom Embedding Models on Persian Web. *Journal of Web Engineering*, *22*(5), 797-820. https://doi.org/10.13052/JWE1540-9589.2253

Breiman, L. (2001). Random Forests. *Machine Learning*, *45*(1), 5-32. https://doi.org/10.1023/A:1010933404324

Burges, C. J. C. (2010). From ranknet to lambdarank to lambdamart: An overview. *Technical Report MSR-TR-2010-82*. http://www.ccs.neu.edu/home/vip/teach/MLcourse/4_boosting/materials/msr-tr-2010-82.pdf

Darrudi, E., Hashemi, H. B., AleAhmad, A., Zare Bidoki, A. M., Habibian, A., Mahdikhani, F., & Rahgozar, M. (2009). dotIR collection for Persian Web retrieval. *Technical Report UT-DBRG-402*.

Derhami, V., Khodadadian, E., Ghasemzadeh, M., & Zareh Bidoki, A. M. (2013). Applying reinforcement learning for Web pages ranking algorithms. *Applied Soft Computing*, *13*(4), 1686-1692. https://doi.org/10.1016/J.ASOC.2012.12.023

Derhami, V., Paksima, J., & Khajah, H. (2015). Web pages ranking algorithm based on reinforcement learning and user feedback. *Journal of AI and Data Mining*, *3*(2), 157–168. https://doi.org/10.5829/idosi.jaidm.2015.03.02.05

Derhami, V., Paksima, J., & Khajeh, H. (2019). RRLUFF: Ranking function based on Reinforcement Learning using User Feedback and Web Document Features. *Journal of AI and Data Mining*, *7*(3), 421–442. https://doi.org/10.22044/JADM.2019.3547.1814

Eberhard, D. M., Simons, G. F., & Fennig, C. D. (2025). Languages of the World. In *Ethnologue* (28th ed.). SIL International. https://www.ethnologue.com

Freund, Y., Iyer, R., Schapire, R. E., Singer, Y., & Dietterich, T. G. (2003). An efficient boosting algorithm for combining preferences. *The Journal of Machine Learning Research*, *4*, 933–969. https://doi.org/10.5555/945365.964285

Friedman, J. H. (2001). Greedy function approximation: A gradient boosting machine. *The Annals of Statistics*, *29*(5), 1189–1232. https://doi.org/10.1214/AOS/1013203451

Ghanbari, E., & Shakery, A. (2018). Query-dependent learning to rank for cross-lingual information retrieval. *Knowledge and Information Systems 2018 59:3*, *59*(3), 711–743. https://doi.org/10.1007/S10115-018-1232-8

Ghanbari, E., & Shakery, A. (2021). A Learning to rank framework based on cross-lingual loss function for cross-lingual information retrieval. *Applied Intelligence 2021*, *52*(3), 3156–3174. https://doi.org/10.1007/S10489-021-02592-Z

Ibrahim, O. A. S., & Landa-Silva, D. (2017). ES-rank: Evolution strategy Learning to Rank approach. In *Proceedings of the ACM Symposium on Applied Computing* (pp. 944–950).

Joachims, T. (2002). Optimizing search engines using clickthrough data. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (pp. 133–142).

Ke, G., Meng, Q., Finley, T., Wang, T., Chen, W., Ma, W., Ye, Q., & Liu, T. Y. (2017). LightGBM: A Highly Efficient Gradient Boosting Decision Tree. In *Proceedings of the Advances in Neural Information Processing Systems* (Vol. 30, pp. 3149–3157).

Keyhanipour, A. H. (2019a). Application of Learning to Rank on Persian Web Content. In *Proceedings of the 5th IEEE International Conference on Web Research* (Vol. 5, pp. 1–14).

Keyhanipour, A. H. (2019b). Effective Learning to Rank for the Persian Web Content. *Journal of Information Technology Management*, *11*(4), 92–109. https://doi.org/10.22059/JITM.2019.284726.2377

Keyhanipour, A. H. (2020). Learning to Rank for the Persian Web Using the Layered Genetic Programming. *Journal of Information and Communication Technology*, *10*(37), 45–70. https://jour.aicti.ir/en/Article/8179/FullText

Keyhanipour, A. H. (2024). Graph-based comparative analysis of learning to rank datasets. *International Journal of Data Science and Analytics*, *17*(2), 165–187. https://doi.org/10.1007/S41060-023-00406-8/METRICS

Keyhanipour, A. H., Moshiri, B., & Rahgozar, M. (2015). CF-Rank: Learning to rank by classifier fusion on click-through data. *Expert Systems with Applications*, *42*(22), 8597–8608. https://doi.org/10.1016/j.eswa.2015.07.014

Keyhanipour, A. H., Moshiri, B., Piroozmand, M., Oroumchian, F., & Moeini, A. (2016). Learning to rank with click-through features in a reinforcement learning framework. *International Journal of Web Information Systems*, *12*(4), 448–476. https://doi.org/10.1108/IJWIS-12-2015-0046

Latifi, S., & Nematbakhsh, M. (2014). Query-independent learning to rank RDF entity results of SPARQL queries. In *Proceedings of the 4th International Conference on Computer and Knowledge Engineering* (pp. 297–301).

Manning, C. D., Raghavan, P., & Schütze, H. (2008). *Introduction to Information Retrieval*. Cambridge University Press.

Mirzababaei, B., Faili, H., & Ehsan, N. (2013). Discourse-aware Statistical Machine Translation as a Context-Sensitive Spell Checker. In *Proceedings of Recent Advances in Natural Language Processing* (pp. 475–482).

Moosavi, N. S. G.-S. G. (2009). A Ranking Approach to Persian Pronoun Resolution. *Advances in Computational Linguistics*, *41*, 169–180.

Newman, M. (2018). *Networks* (Second Edition). Oxford University Press.

Qin, T., & Liu, T. Y. (2013). *Introducing LETOR 4.0 Datasets*. arXiv. http://arxiv.org/abs/1306.2597

Rahimi, R., Shakery, A., Dadashkarimi, J., Ariannezhad, M., Dehghani, M., & Esfahani, H. N. (2016). Building a multi-domain comparable corpus using a learning to rank method†. *Natural Language Engineering*, *22*(4), 627–653. https://doi.org/10.1017/S1351324916000164

Rahimi, Z., & Shamsfard, M. (2023). A Neuro Symbolic Approach for Contradiction Detection in Persian Text. *Journal of Universal Computer Science*, *29*(3), 242–264. https://doi.org/10.3897/jucs.90646

Rahimi, Z., & ShamsFard, M. (2024). *A Knowledge-Based Approach for Recognizing Textual Entailments with a Focus on Causality and Contradiction*. Available at SSRN 4526759. https://doi.org/10.21203/RS.3.RS-3826973/V1

Rezvani, M., & Hashemi, S. M. (2012). Enhancing accuracy of topic sensitive PageRank using Jaccard index and cosine similarity. In *Proceedings of the 2012 IEEE/WIC/ACM International Conference on Web Intelligence (WI 2012)* (pp. 620–624).

Rogers, R. (2010). Internet Research: The Question of Method—A Keynote Address from the YouTube and the 2008 Election Cycle in the United States Conference. *Journal of Information Technology & Politics*, *7*(2–3), 241–260. https://doi.org/10.1080/19331681003753438

Shyam, K. P., Jagannathan, S., & Rajavel, M. (2013). A New Technique for Ranking Web Pages and Adwords. *International Journal of Computer Applications*, *82*(12), 32–36. https://doi.org/10.5120/14171-2378

Syriopoulos, P. K., Kalampalikis, N. G., Kotsiantis, S. B., & Vrahatis, M. N. (2025). kNN Classification: a review. *Annals of Mathematics and Artificial Intelligence*, *93*(1), 43–75. https://doi.org/10.1007/s10472-023-09882-x

Thakur, N., Kazi, S., Luo, G., Lin, J., & Ahmad, A. (2025). MIRAGE-Bench: Automatic Multilingual Benchmark Arena for Retrieval-Augmented Generation Systems. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies* (pp. 274–298).

Veisi, H., & Shandi, H. F. (2020). A Persian Medical Question Answering System. *International Journal on Artificial Intelligence Tools*, *29*(6), 205-219. https://doi.org/10.1142/S0218213020500190

Wei, Z., Xu, J., Lan, Y., Guo, J., & Cheng, X. (2017). Reinforcement learning to rank with Markov decision process. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval* (pp. 945–948).

Wikipedia. (2026, June 6). Languages used on the Internet. *Wikipedia*. https://en.wikipedia.org/wiki/Languages_used_on_the_Internet#cite_note-w3techs_historical_trends-1

Wu, Q., Burges, C. J. C., Svore, K. M., & Gao, J. (2010). Adapting boosting for information retrieval measures. *Information Retrieval*, *13*(3), 254–270. https://doi.org/10.1007/S10791-009-9112-1/FIGURES/4

Zeraatkar, A., Mirvaziri, H., & Ahsaee, M. G. (2019). Improvement of Page Ranking Algorithm by Negative Score of Spam Pages. *Webology*, *Volume 16*(2), 43–56. https://doi.org/10.14704/WEB/V16I2/A187

Bibliographic information of this paper for citing:

Piroozmand, Maryam; Moeini, Ali; Mazoochi, Mojtaba & Keyhanipour, Amir Hosein (2026). Fusion of Graph-Based Ensembles Using Fuzzy Integrals for Persian Learning to Rank. *Journal of Information Technology Management*, 18 (3), 47-81. https://doi.org/10.22059/jitm.2026.412047.4443