Kathmandu University Journal of Science, Engineering and Technology, Vol. 20 No. 1, April 2026

Kathmandu University
Journal of Science, Engineering and Technology

News aggregation and recommendation using clustering algorithms and graph theory

Mukul Aryal*[a], Bigya Vijay Dhungana[a], and Harish Chandra Bhandari[b]

[a]Department of Computer Science and Engineering, Kathmandu University, Nepal.
[b]Department of Mathematics, Kathmandu University, Nepal.

Abstract

The proliferation of online news sources creates significant challenges in managing redundant articles and providing personalized recommendations, especially without user data. This study proposes a methodology that addresses these issues through a two-stage process. We first apply Agglomerative Hierarchical Clustering (AHC) to a corpus of 8,142 articles scraped from prominent Nepali news websites, reducing content duplication by grouping articles based on semantic similarity. The resulting unique articles are then modeled as nodes in a weighted, undirected graph. A content-based recommendation engine generates curated news lists by applying Dijkstra's algorithm to this graph, leveraging a composite edge weight that accounts for semantic similarity, publication recency, and category. Our evaluation shows this approach effectively reduced article redundancy by 15.9%. Furthermore, the recommendation system demonstrated high performance, generating 10 recommendations in 1.14 seconds on average, and a user study (N=192) found the recommendations to be highly relevant (40% potential CTR). This work validates a computationally efficient, graph-based framework for news aggregation and recommendation that entirely bypasses the training overhead and 'cold start' limitations of traditional machine learning models.

Keywords: News aggregation; Clustering; Graph theory; Recommendation system; Content based filtering.

1. Introduction

In the modern digital era, the proliferation of online news sources has led to an unprecedented volume of information. While this provides users with diverse perspectives, it also creates significant challenges, namely information overload and content redundancy. News aggregation platforms aim to mitigate this by compiling articles from various outlets into a single, accessible interface. However, a persistent problem is the presence of duplicate or near-duplicate articles reporting on the same event, which clutters the user experience and offers little additional value.

Furthermore, personalizing the news consumption experience through recommendation systems is critical for user engagement. The industry standard for personalization often relies on Collaborative Filtering (CF). However, CF models face the *"cold start"* problem. This occurs when a new user enters the system; since they have no history of clicks, likes, or shares, the system cannot infer their preferences, leading to poor initial recommendations. Additionally, for systems with a rapidly changing set of items like news, maintaining up-to-date collaborative models can be computationally expensive.

This paper proposes a solution to these challenges by moving away from user-history dependency and focusing on the relationships between the articles themselves. We present a two-stage approach:

1. **Deduplication via Clustering:** Using Agglomerative Hierarchical Clustering (AHC) to group semantically similar news articles, thereby presenting the user with unique stories rather than repetitive headlines.
2. **Graph-Based Recommendation:** Modeling unique articles as nodes in a graph. By calculating weighted edges based on content similarity, recency, and category, we can generate relevant recommendations using shortest-path algorithms.

Research contributions: The major contributions of this study are:

- Development of a lightweight scraping and preprocessing pipeline specifically for Nepali Unicode text.
- Implementation of a clustering mechanism that effectively identifies and groups duplicate news stories across different publishers.
- Formulation of a graph-based recommendation algorithm that operates without user training data, effectively solving the cold-start problem.
- Empirical evaluation of the system's efficiency and relevance using real-world data and human testers.

This study demonstrates that combining text clustering and graph theory creates a powerful, efficient approach for news aggregation, particularly suitable for resource-constrained environments or applications where user data is scarce.

2. Related works

The application of clustering algorithms to document and news deduplication has been extensively explored in prior research. Abdalla [1] provides a comparative analysis of K-means and Agglomerative Hierarchical Clustering (AHC) on small datasets, using TF-IDF weighting and similarity measures like cosine and Euclidean distance. The study evaluates performance via purity and entropy metrics, finding AHC superior in entropy (indicating better cluster homogeneity) and stability, while K-means excels in purity

*Corresponding author. Email: aryalmukul@gmail.com

and computational speed. This aligns with our use of AHC for semantic grouping of news articles, particularly in handling high-dimensional text data.

Similarly, Popat et al. [2] investigate hierarchical document clustering based on cosine similarity, emphasizing its role in measuring likeness between text objects and generating dendrograms for visualization. Their experimental approach highlights the dependency on similarity measures and clustering criteria, which informs our cosine-based distance matrix for daily news processing.

Extending this to duplicate detection, Daneshpour and Barzegari [3] propose a multi-level hierarchical clustering method with relative similarity functions, achieving 90% duplicate detection at 97% accuracy with reduced computation time. This method's focus on feature-based levels complements our deduplication strategy, offering potential enhancements for handling noisy news corpora.

In the domain of news aggregation, studies have examined both systemic architectures and user consumption patterns. Lee and Chyi [4] analyze the rise of online news aggregators like Yahoo News and Google News, using a national U.S. survey to identify predictors such as age and ethnicity, while noting non-competitive dynamics with traditional media. Their integration of uses-and-gratifications theory reveals that users avoid opinion-driven content on aggregators, which resonates with our content-based focus amid information overload.

Paliouras et al. [5] describe the Personalized News System (PNS), a web-based aggregator that employs wrappers for HTML/RSS extraction, categorizes content, and adapts recommendations via machine learning on user interactions and similarities. A user study in their work validates personalization merits, inspiring our graph traversal but differing in our avoidance of ML overhead.

Kiryanov [6] researches content aggregation architectures, detailing web crawling, fuzzy duplicate detection via shingling and blocking, and scalable designs using microservices, event-driven styles, and caching for high availability. His emphasis on reducing update delays through Poisson-based retrieval policies supports our periodic scraping and graph updates.

For Nepali-specific contexts, a Reddit discussion [7] surveys user preferences for aggregators like Setopati, Dekhapadi, and Twitter lists, emphasizing quality and low-noise sources amid elections. This community insight underscores the need for localized solutions, as in our scraping from Nepali portals.

Recommendation systems for news often blend clustering with filtering techniques. Liu et al. [8] introduce a hybrid algorithm combining K-means clustering for content-based grouping with collaborative filtering for user preferences, providing an overview of news recommendation technologies. This approach addresses cold-start issues but incurs training costs, contrasting our graph-based method that prioritizes efficiency and zero-overhead personalization via weighted edges.

3. Methodology

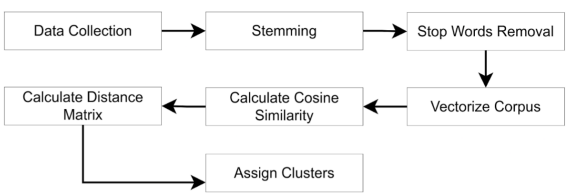

Figure 1: Overview of the proposed text clustering pipeline, illustrating the sequential process from data collection and preprocessing to vectorization, similarity computation, and final cluster assignment.

The proposed methodology, as illustrated in Figure 1, outlines a systematic pipeline for clustering news articles based on textual similarity. The process begins with data collection, followed by stemming and stop words removal to normalize the text. The refined corpus is then vectorized, and cosine similarity is computed to quantify pairwise relationships between articles. Using this similarity information, a distance matrix is constructed, which serves as the basis for the clustering algorithm that groups semantically similar news articles together.

3.1. Dataset collection

The dataset for this study was compiled over a one-month period, from December 23, 2024, to January 25, 2025. A total of 8,142 news articles were scraped from three prominent Nepali news portals:

1. OnlineKhabar [9]: 2,993 articles (36.8%)
2. RatoPati [10]: 3,174 articles (39.0%)
3. SetoPati [11]: 1,975 articles (24.2%)

The collected articles were categorized into several predefined sections: Breaking, National, Finance, Sports, Entertainment, and International. The distribution across these categories highlighted a primary focus on 'Breaking' and 'National' news, reflecting typical reader interest and news cycle priorities.

3.2. Preprocessing

To prepare the raw text data for analysis, a standard Natural Language Processing (NLP) pipeline was implemented. This is a critical step to ensure that the semantic similarity calculations are accurate and meaningful.

Stemming: Nepali words were reduced to their root form. For example, a word like "हुँदैछ" (hudaichha, meaning "is happening") is stemmed to "हुनु" (hunu, meaning "to happen"). This process ensures that different grammatical forms of the same word are treated as equivalent, improving consistency. To make the stemming process easier, we used the *nepali-stemmer* python package [12] which implements a modified Hindi Lemmatizer [13] for Nepali language.

Stop word removal: Common, low-information filler words in the Nepali language (e.g., "ले", "को", "मा" - le, ko, ma) were removed from the articles. This step helps to reduce noise and allows the analysis to focus on the words that carry the most semantic weight.

Corpus vectorization (TF-IDF): The preprocessed text corpus was transformed into a numerical representation using the Term Frequency-Inverse Document Frequency (TF-IDF) model. TF-IDF evaluates the importance of a word in a document relative to the entire collection of documents (corpus). Using TF-IDF with stopwords removal and stemming is a common technique that yields great results [14]. The TF-IDF score for a term t in a document d is calculated as:

$$\text{TF-IDF}(t, d) = \text{TF}(t, d) \times \text{IDF}(t)$$

Where:

$$\text{TF}(t, d)$$

is the Term Frequency, or the number of times term t appears in document d.

$$\text{IDF}(t) = \log\left(\frac{N}{1 + \text{df}(t)}\right)$$

Here, N is the total number of documents in the corpus, and $\text{df}(t)$ is the number of documents containing the term t. This gives

higher weight to terms that are frequent in a single document but rare across the entire corpus.

We chose TF-IDF over modern deep learning methods as TF-IDF can capture similar semantic similarity and is much faster in resource constrained environments [15].

3.3. News clustering

Following preprocessing, the vector representations of the articles were used to group them based on similarity.

3.3.1. Similarity calculation

The cosine similarity between the TF-IDF vectors of every pair of documents was calculated. Cosine similarity measures the cosine of the angle between two vectors, with a value of 1 indicating identical documents and 0 indicating no similarity. The formula is:

$$\cos(\theta) = \frac{A \cdot B}{\|A\|\|B\|}$$

where A and B are the TF-IDF vectors of two news articles.

3.3.2. Distance matrix and conflict handling

To be used with clustering algorithms, the similarity scores were converted into a distance metric. This was achieved by creating a distance matrix where the distance is defined as:

$$\text{Distance} = 1 - \text{Cosine Similarity}$$

We define "near-duplicates" as article pairs where the Distance is below a specific threshold (experimentally set at 0.3 for this dataset). This metric focuses on semantic topic overlap. It is important to note that this method groups articles covering the same event regardless of editorial bias. Therefore, articles with "conflicting interests" (differing viewpoints on the same event) are grouped into the same cluster, which is a desirable feature for a news aggregator aiming to provide a holistic view of a topic.

3.3.3. Agglomerative hierarchical clustering (AHC)

Agglomerative Hierarchical Clustering (AHC) was selected as the primary clustering algorithm. It is a bottom-up hierarchical method that works by initially treating each data point (news article) as its own cluster. It then iteratively merges the two closest clusters until a stopping criterion is met.

The *average* linkage criterion was used, which measures the distance between clusters as the average distance between all pairs of points in the respective clusters. The process was performed daily on new articles, with a dynamic threshold set to determine the final number of clusters, effectively grouping articles about the same event.

3.4. Graph-based recommendation system

The core of our recommendation engine is a graph-based model that leverages the content and metadata of the deduplicated news articles.

3.4.1. Graph construction

A weighted, undirected graph $G = (V, E)$ was constructed where:

- **Nodes (V)**: Each unique news article (post-clustering) is a node in the graph.
- **Edges (E)**: An edge exists between every pair of nodes, representing the relationship between two articles.

3.4.2. Edge weight calculation

The weight of each edge is a crucial component, as it quantifies the similarity and relevance between two connected articles. We devised a composite *EdgeScore* calculated as a weighted sum of four factors:

$$EdgeScore = 2 \times \text{PublishedScore} + 3 \times \text{CosineSimilarityScore} + 2 \times \text{CategoryScore} + \text{ScrapeScore}$$

The coefficients *(2, 3, 2, 1)* were tuned based on performance on a small, held-out validation set of articles, where relevance was manually annotated. These weights prioritize semantic similarity while giving significant consideration to category and recency.

The components are defined as:

- **PublishedScore**: Measures the temporal proximity of two articles.
$$\text{PublishedScore} = \lfloor \cos(\Delta\text{time}) \rfloor$$
where Δtime is the difference in publication time. The absolute cosine value ensures that articles published closer together receive a higher score.
- **CosineSimilarityScore**: The raw cosine similarity score between the articles' TF-IDF vectors. This is the most heavily weighted factor.
- **CategoryScore**: A binary score that is 1 if both articles belong to the same category (e.g., 'Sports') and 0 otherwise.
- **ScrapeScore**: An exponential decay function that models the decreasing relevance of older news.
$$\text{ScrapeScore} = e^{-\frac{|\Delta \text{ScrapeTime}|}{\text{ScrapeFrequency}}}$$

3.4.3. Recommendation generation

To generate recommendations for a given source article, we transform the graph and apply pathfinding algorithms. All the work that included building and exploring the graph was done using the NetworkX [16] Python package which provided an easy to use API along with many built-in functions.

Graph Transformation Since standard shortest path algorithms like Dijkstra's Algorithm [17] find paths with the minimum cumulative weight, we convert our *EdgeScore (where higher is better)* into a distance metric *(where lower is better)*. This is done by defining the edge weight in the graph as:

$$\text{weight} = C - \text{EdgeScore}$$

where C is a constant (e.g., 10) larger than any possible EdgeScore.

Iterative Nearest Neighbor Search The shortest path provides a strong initial set of recommendations. To expand this, we propose the *Iterative Nearest Neighbor Search* algorithm. This algorithm operates in a greedy fashion. It starts with a seed set of nodes and iteratively adds the closest neighbor (based on edge weight) to the current set until the quota is filled.

Algorithm Analysis While this approach is a greedy heuristic and does not guarantee a globally optimal set of recommendations in a mathematical sense, it is highly efficient and feasible for real-time systems. The complexity of selecting K recommendations from a graph with average node degree D is approximately $O(K \cdot D)$. Since our graph is sparse after thresholding low-similarity edges, D remains small, ensuring rapid execution.

Algorithm 1: Iterative Nearest Neighbor Search Algorithm

Input : Graph G, Path P, Desired number of news items N
Output : List of final recommendations

```
finalRecommendations ← P;
while True do
    foreach node N in finalRecommendations do
        c ← closest neighbor of N not in P;
        finalRecommendations.add(c);
    if finalRecommendations.length ≥ N then
        break
return finalRecommendations[1...N]
```

3.5. Illustrative example

To clarify the recommendation logic, consider two articles:

- **Article A (Source):** "Nepal wins cricket match against UAE" (Category: Sports, Time: 10:00 AM)
- **Article B (Candidate):** "Sompal Kami takes 5 wickets" (Category: Sports, Time: 11:00 AM)

1. **Similarity:** TF-IDF Cosine Similarity is calculated as 0.85 (high overlap).
2. **Recency:** Time difference is 1 hour. PublishedScore ≈ 0.99.
3. **Category:** Both are 'Sports', so CategoryScore $= 1$.
4. **Calculation:** EdgeScore $= (2 \times 0.99) + (3 \times 0.85) + (2 \times 1) +$ ScrapeScore ≈ 6.53.
5. **Graph Weight:** Taking $C = 10$, the edge weight is $10 - 6.53 = 3.47$.

This low edge weight signifies a strong connection, making Article B a high-priority recommendation for a user reading Article A.

4. Evaluation and results

The effectiveness of both the clustering and recommendation systems was evaluated through computational metrics and human feedback. Feedback was collected from 192 human testers recruited via a snowball sampling method. The participants represented a diverse range of age groups and educational backgrounds.

4.1. Clustering and deduplication performance

The AHC clustering approach proved effective at identifying redundant news. Out of the initial 8,142 articles, the system identified and grouped duplicates, resulting in a deduplicated set of 6,848 unique story clusters. This corresponds to a *Data Deduplication Ratio (DDR)* of 15.9%.

Human evaluation confirmed the quality of the clusters. When presented with the article groups, 91.77% of participants gave the clusters a relevance score of 8 or higher on a scale of 1 to 10, indicating that the grouped articles were indeed about the same topic. Furthermore, when comparing clusters generated by AHC and DBSCAN, 59% of testers preferred the coherence and quality of the AHC clusters.

4.2. Recommendation system performance

The graph-based recommendation system was evaluated on two key metrics: performance speed and user-perceived relevance.

Speed: The system demonstrated high efficiency. On average, it took only 1.14 seconds on average to generate 10 recommendations for any given news article. This was achieved without any code parallelization, highlighting the inherent speed of the graph traversal approach compared to the inference time of large ML models.

Relevance: The relevance of the recommendations was assessed through a user study (N=192). On average, participants indicated a willingness to click on 4 out of the 10 recommended articles, resulting in a 40% *potential click-through* rate or a *user-reported relevance score* of 4/10.

5. Discussion

The results of this study are promising. A 15.9% data deduplication rate confirms that significant content overlap exists across major news portals, and that clustering is an effective technique to manage this redundancy.

The most significant contribution is the validation of the graph-based recommendation system. A 40% Potential CTR achieved without any user-specific data is a powerful result. It demonstrates that a deep understanding of the content's semantic meaning, temporal relevance, and categorical relationships can be a strong proxy for user interest. The system's main advantage is its efficiency and simplicity. By forgoing the need for ML training pipelines, it eliminates associated costs, complexities, and the "cold start" problem. It is inherently adaptable; adding a new news category or handling a sudden surge in articles only requires updating the graph, not retraining a model.

However, the approach has limitations. The system is purely content-based and does not yet learn from user behavior. While this is an advantage for privacy and simplicity, a hybrid approach could yield even better results. Finally, the snowball sampling method may introduce some bias into the user evaluation.

6. Conclusion and future work

This paper presented a two-stage system for news aggregation and recommendation. By combining Agglomerative Hierarchical Clustering for effective deduplication and a fast, graph-based recommendation engine, we have demonstrated a powerful alternative to traditional, resource-intensive methods. The system achieved a 15.9% reduction in article redundancy and a 40% recommendation CTR, with strong acceptance confirmed by diverse user groups. This approach is ideal for systems with a fixed set of items, limited computational resources, and a need for a robust, easily maintainable recommendation solution.

Future enhancements will focus on three key areas:

- **Performance Optimization:** Parallelizing the graph construction and recommendation algorithms to further decrease response times.
- **Hybrid Modeling:** Incorporating user behavior data (e.g., clicks, reading time) to dynamically adjust edge weights, evolving the system into a hybrid model that combines content-based and collaborative filtering principles.
- **Advanced NLP:** Implementing state-of-the-art deep learning models like BERT [18] or other transformer [19] based embedding models for the text vectorization step. This could provide a more nuanced understanding of semantic similarity, especially for short or complex texts, potentially improving both clustering and recommendation quality.

References

[1] Abdalla H I, A brief comparison of k-means and agglomerative hierarchical clustering algorithms on small datasets, *Proceeding of 2021 International Conference on Wireless Communications, Networking and Applications* (2022) 623–632. https://doi.org/10.1007/978-981-19-2456-9_64.

[2] *Hierarchical document clustering based on cosine similarity measure* (2017). https://doi.org/10.1109/icisim.2017.8122166. URL https://ieeexplore.ieee.org/document/8122166.

[3] Daneshpour N & Barzegari A, A new method for duplicate detection using hierarchical clustering of records, *Signal and Data Processing*, 18 (2022) 3–22. https://doi.org/10.52547/jsdp.18.4.3. URL https://jsdp.rcisp.ac.ir/article-1-1039-en.html.

[4] Lee A M & Chyi H I, The rise of online news aggregators: Consumption and competition, *International Journal on Media Management*, 17 (2015) 3–24. https://doi.org/10.1080/14241277.2014.997383.

[5] Paliouras G, Mouzakidis A, Moustakas V & Skourlas C, Pns: A personalized news aggregator on the web, *Intelligent Interactive Systems in Knowledge-Based Environments* (2008) 175–197. https://doi.org/10.1007/978-3-540-77471-6_10.

[6] Kiryanov D A, Research of the methods of creating content aggregation systems, *Programmnye sistemy i vyčislitel□nye metody* (2022) 9–31. https://doi.org/10.7256/2454-0714.2022.1.37341.

[7] meanadhikari. What is the go to news aggregator site for Nepali news (2022). URL https://www.reddit.com/r/Nepal/comments/ymyqj7/what_is_the_go_to_news_aggregator_site_for_nepali/.

[8] Liu J, Song J, Li C, Zhu X & Deng R, A hybrid news recommendation algorithm based on k-means clustering and collaborative filtering, *Journal of Physics: Conference Series*, 1881 (2021) 032050. https://doi.org/10.1088/1742-6596/1881/3/032050.

[9] Onlinekhabar. https://www.onlinekhabar.com (2025). Accessed: 2025-08-25.

[10] Ratopati. https://www.ratopati.com (2025). Accessed: 2025-08-25.

[11] Setopati. https://www.setopati.com (2025). Accessed: 2025-08-25.

[12] oya163. nepali-stemmer 0.0.2 (2025). URL https://pypi.org/project/nepali-stemmer/.

[13] Paul S, Joshi N & Mathur I. Development of a Hindi lemmatizer (2025). URL https://arxiv.org/abs/1305.6211.

[14] Thapa L B R & Bal B K, Classifying sentiments in Nepali subjective texts. https://doi.org/10.1109/iisa.2016.7785374.

[15] Sazan S A, Miraz M H & Rahman A B M M, Enhancing depressive post detection in Bangla: A comparative study of tf-idf, bert and fasttext embeddings, *SSRN Electronic Journal*. https://doi.org/10.2139/ssrn.4885802.

[16] Hagberg A, Swart P & S Chult D. Exploring network structure, dynamics, and function using networkx (2008). URL https://www.osti.gov/biblio/960616.

[17] Dijkstra E W, A note on two problems in connexion with graphs, *Numerische Mathematik*, 1 (1959) 269–271. https://doi.org/10.1007/bf01386390.

[18] Devlin J, Chang M W, Lee K & Toutanova K. Bert: Pre-training of deep bidirectional transformers for language understanding (2018). URL https://arxiv.org/abs/1810.04805.

[19] Vaswani A, Shazeer N, Parmar N, Uszkoreit J, Jones L, Gomez A N, Kaiser L & Polosukhin I. Attention is all you need (2017). URL https://arxiv.org/abs/1706.03762.