International Journal of Advanced Engineering Research and Science (IJAERS)
Peer-Reviewed Journal
ISSN: 2349-6495(P) | 2456-1908(O)
Vol-12, Issue-8; Aug, 2025
Journal Home Page Available: https://ijaers.com/
Article DOI: https://dx.doi.org/10.22161/ijaers.128.2

Recommendation System Based on Semantic Analysis and Network Models

N.S. Fedotov

Financial University under the Government of the Russian Federation

Received: 22 Jul 2025,

Received in revised form: 15 Aug 2025,

Accepted: 18 Aug 2025,

Available online: 21 Aug 2025

***Keywords*— recommendation systems, semantic analysis, network analysis, machine learning**

***Abstract*—** *In this work, we implement a hybrid recommendation system for news articles that combines two primary approaches: semantic analysis via TF–IDF vectorization of headlines and Nearest Neighbors search, and network analysis using an article-similarity graph constructed from shared tags. To improve recommendation quality, rare tags were filtered out, and the number of articles per tag was capped to balance the dataset. A weighted combination of semantic and graph-based scores was also employed with parameter tuning. Precision was adopted as the evaluation metric, measuring the proportion of correctly predicted tags in the recommended articles against the ground-truth tags in the test set. Experimental results show that the hybrid model effectively leverages both semantic headline features and network connections between articles. When increasing the per-tag article cap from 1,000 to 3,000, precision rose from 0.168 to 0.187—an 11% improvement—while training time increased from 31.8 s to 506 s. This trade-off confirms the value of expanding the data scope and demonstrates the strength of the hybrid approach. The achieved precision on the test set indicates that integrating semantic and network analyses yields more accurate and well-grounded recommendations tailored to user interests.*

I. INTRODUCTION

The volume of scientific and popular publications is growing exponentially: tens of thousands of new research articles and millions of news items appear each year. At this pace of information accumulation, filtering truly important and relevant content becomes increasingly difficult. Standard search engines return long result lists, whose relevance does not always correlate with content depth or expert assessment. As a result, researchers, journalists, and curious readers spend hours navigating information streams and risk missing key discoveries or valuable insights [1].

Recommendation systems have emerged as an effective tool against information overload. They perform contextual filtering: instead of presenting thousands of irrelevant links, they provide a few personalized suggestions. To do so, these systems analyze document attributes (topics, keywords, semantic vectors) and study user behavior (clicks, reading time, ratings). By combining statistical models with machine learning techniques, modern recommenders can not only reproduce past interests but also propose related topics, thereby broadening users' horizons [2].

The application of recommendation systems has long since extended beyond e-commerce. Scientific databases help researchers quickly locate articles in their specialization, automatically notifying them of newly published reviews or experimental results. News portals use recommendations to keep readers engaged by suggesting reports and analyses similar to those already read. Educational platforms select learning modules and assessments according to a student's proficiency level and progress, making education more flexible and effective [3, 4].

A key objective of such systems is to predict the materials most likely to interest users. This entails generating a ranked list of articles or items with the highest probability of being rated "useful" or "interesting."

The goal of this research is to develop a recommendation system based on semantic analysis and network models.

II. MATERIALS AND METHODS

The research was conducted at the Department of Artificial Intelligence, Financial University under the Government of the Russian Federation. We used the Lenta.ru news dataset (1990–present) [5], and implemented the system in Python with the following libraries: pandas [6], matplotlib [7], collections, random, scikit-learn [8], networkx [9], itertools [10], and numpy [11].

III. RESULTS OF THE RESEARCH

In the initial phase of our research, we analyzed traditional approaches to recommendation system design. This investigation revealed several inherent limitations of such systems.

In particular, the content-based approach relies on analyzing individual document features—identifying keywords, calculating TF–IDF scores, or constructing vector embeddings. While recommendations are generated based on similarity between new and previously viewed content, the resulting articles often exhibit narrow thematic overlap and rarely expand beyond the user's established interests. The system is prone to "self-reinforcement," repeating familiar interest patterns without offering surprising or novel discoveries, which decreases overall diversity. Additionally, it fails to incorporate broader trends and community-wide preferences.

The second approach, collaborative filtering, focuses on collective user behavior—ratings, views, clicks, and other forms of interaction. This method identifies similar user profiles and recommends items that appealed to those with comparable preferences. However, it has significant limitations: during "cold-start" scenarios (involving new users or new items), data sparsity leads to poor recommendation accuracy; high-interaction items tend to dominate the output, crowding out niche or specialized content; and the sparsity of the interaction matrix makes it difficult to detect meaningful correlations. Furthermore, the algorithm cannot anticipate interest in entirely new topics that lack prior ratings.

Neither content-based nor collaborative filtering approaches integrates deep semantic analysis or captures the network structure between articles. Each functions within its own "dimension"—content features or behavioral patterns. Both approaches neglect nuanced semantic relationships, such as overlapping subtopics; they do not employ tag-based network models that reflect community clustering; and they omit social and contextual factors such as authorship, citation impact, or expert evaluations. As a result, their architectures lack flexibility in adapting to shifting interests and emerging themes.

In contrast, the hybrid system proposed in this study combines the strengths of both content-based and collaborative filtering techniques while enhancing them with semantic similarity analysis and a tag-based network model. Semantic analysis of article headlines enables identification of deep thematic relationships between articles, while the graph model encodes linkages via shared tags and social context. This design alleviates cold-start problems by relying on semantic structure—even for new articles; it mitigates popularity bias by prioritizing meaning-based connections; and it improves recommendation diversity and accuracy by blending both textual features and network structure.

Overall, the hybrid architecture significantly enhances article selection and contributes to higher user satisfaction. In the next section, we examine the proposed model in greater detail.

For the semantic component, we used the TfidfVectorizer to transform article headlines into TF–IDF vectors. For each article, the TF–IDF values are calculated per word (Fig. 1), creating a matrix where rows correspond to documents and columns represent terms with their respective TF–IDF scores.

$$TF(t,d) = \frac{number\ of\ times\ t\ appears\ in\ d}{total\ number\ of\ terms\ in\ d}$$

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

$$TF-IDF(t,d) = TF(t,d) * IDF(t)$$

Fig. 1: TF-IDF calculation: t - term; d - document (Karen Spärck Jones, 1972)

Based on the computed semantic representations, a Nearest Neighbors model was constructed to support article-to-article recommendations. Cosine distance was selected as the similarity metric, enabling precise measurement of angular proximity between TF–IDF vectors. Upon querying the system with a target headline, the algorithm returns the top kk articles that exhibit the smallest cosine distance to the query vector, representing the most semantically similar articles.

To build the hybrid recommender system in Python, a comprehensive preprocessing stage was conducted. The

dataset consists of news articles collected from the Lenta.ru news portal, spanning content published since 1990. Each entry includes a tag, a headline, the full article text, and a topic classification.

During implementation, several preprocessing steps were applied. A range of libraries was used (Table 1) alongside custom Python code (Table 2) to remove non-informative tags and discard those appearing infrequently (fewer than 20 times across the dataset). As a result, a more balanced and representative tag distribution was achieved (see Fig. 2).

Table. 1: Python libraries used for data loading and preprocessing

n/n	Python libraries
1	import pandas as pd from collections import Counter

Table. 2: developed code for reading and preprocessing data

n/n	code

1	df = pd.read_csv('lenta-ru-news.csv')
	df = df[['title', 'tags']]
	df['tag_list'] = df['tags'].apply(lambda x: [tag.strip() for tag in x.split(',')] if isinstance(x, str) else [])
	all_tags = [tag for tags in df['tag_list'] for tag in tags]
	tag_counts = Counter(all_tags)
	df = df[df['tags'].str.strip() != 'Все']
	
	def remove_all_tag(tags):
	if pd.isna(tags):
	return ''
	tag_list = [tag.strip() for tag in tags.split(',')]
	tag_list = [tag for tag in tag_list if tag != 'Все']
	return ', '.join(tag_list)
	
	df['tags'] = df['tags'].apply(remove_all_tag)
	all_tags = [tag for tags in df['tag_list'] for tag in tags]
	tag_counts = Counter(all_tags)

	tag_min = 20 popular_tags = {tag for tag, count in tag_counts.items() if count >= tag_min} def is_news_popular(row): tags_ok = any(tag in popular_tags for tag in row['tag_list']) return tags_ok df = df[df.apply(is_news_popular, axis=1)].drop(columns=['tag_list'])

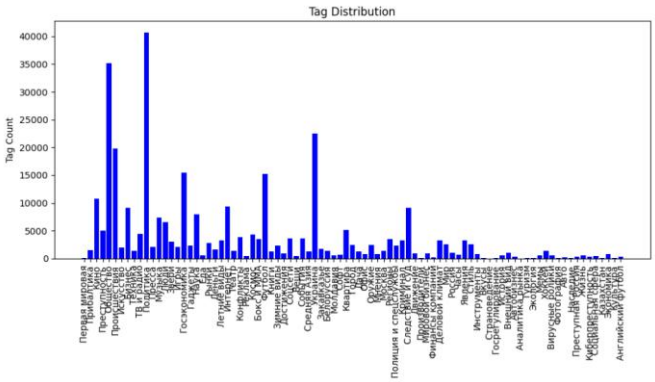

Fig. 2: distribution of tags after preprocessing

The semantic part of the hybrid recommender system was implemented using the Python programming language. To accomplish this, we utilized functions from the scikit-learn library (Table 3), as well as custom code developed during the course of this study (Table 4).

Table. 3: Libraries used in the implementation of the semantic component

n/n	Python libraries
1	from sklearn.feature_extraction.text import TfidfVectorizer from sklearn.neighbors import NearestNeighbors from sklearn.model_selection import train_test_split

Table. 4: Custom Python code for the semantic module

n/n	Custom code
1	df_train, df_test = train_test_split(df1, test_size=0.2, random_state=42) tfidf = TfidfVectorizer(max_features=2500) tfidf_matrix_train = tfidf.fit_transform(df_train['title']) nn_model = NearestNeighbors(n_neighbors=20, metric='cosine') nn_model.fit(tfidf_matrix_train)

In developing the tag-based network model, we relied on constructing an undirected weighted graph based on the tags present in the articles. Each vertex in this graph represents a single article, and an edge connects two articles if their tag sets intersect. The intensity of the thematic connection between articles i and j is quantitatively expressed by the edge weight, which equals the number of common tags:

$$w_{ij} = T_i \cap T_j$$

where T_i and T_j are the tag sets of articles i and j, respectively.

The resulting graph reflects not only the existence but also the strength of thematic overlaps: the more tags two articles share, the denser and more pronounced their connection in the network structure. This model allows identification of clusters of thematically related articles and accounts for the collective distribution of topics via labels assigned by authors or editors.

Subsequently, recommendations can be generated based on this structural component by selecting, for a given article, the nodes most strongly connected to it - i.e., the articles with the highest edge weights. By accounting for network connections, the model complements content analysis and provides a more diverse and well-founded selection of materials.

The network part of the hybrid recommendation system was then implemented in Python. For this purpose, we used a number of libraries (Table 5) and our custom code (Table 6).

Table. 5: libraries used for implementing the network component

n/n	Python libraries
1	from collections import defaultdict import networkx as nx from itertools import combinations

Table. 6: custom code for the network component

n/n	Custom code

1	tag_to_articles = defaultdict(set)
	for idx, row in df_train.iterrows():
	for tag in row['tags'].split(','):
	tag_to_articles[tag.strip()].add(idx)
	
	G1 = nx.Graph()
	for tag, articles in tag_to_articles.items():
	for a1, a2 in combinations(articles, 2):
	if G1.has_edge(a1, a2):
	G1[a1][a2]['weight'] += 1
	else:
	G1.add_edge(a1, a2, weight=1)

This code builds a model that captures connections between articles based on their shared tags. Each article becomes a node in the graph, and an edge is created between two articles if they share at least one tag. The edge weight indicates how many tags they have in common—the higher the weight, the stronger the thematic similarity between the articles. Such a graph enables analysis of the content structure, identification of clusters of similar publications, and can be used for tasks like recommendation or content clustering.

During the development of the hybrid recommender system, we first limited the maximum number of news articles per tag to 1 000 items (Fig. 3). To further improve model quality, we then increased this cap to 3 000 items (Fig. 4).

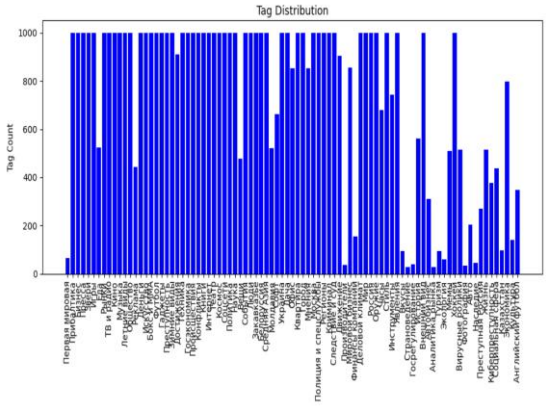

Fig. 3 – Tag distribution after capping the maximum amount of news per tag at 1 000 items

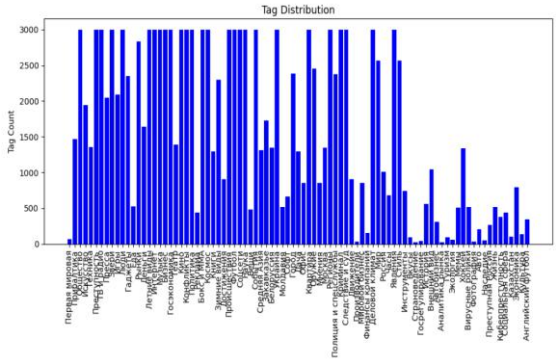

Fig. 4 – Tag distribution after capping the maximum amount of news per tag at 3 000 items

Precision was used as the evaluation metric, measuring the accuracy of predicted tags from recommended articles against the ground-truth tags in the test set. After raising the cap on the number of articles per tag, we observed an increase in precision, although training time also grew (Table 7).

Table. 7:Metrics for the model on different data volumes

Max. news per tag	Training time (s)	Precision
1000	31.8	0.168
3000	506.0	0.187

IV. CONCLUSION

During this work, a hybrid recommendation system for news articles was implemented, employing two key approaches: semantic analysis via headline vectorization (TF–IDF), a Nearest Neighbors search, and network analysis using an article-similarity graph constructed from shared tags. To improve quality, rare tags were filtered out and the number of articles per tag was capped to balance the sample. A combination of semantic and graph-based weights with parameter tuning was also employed.

The conducted experiments demonstrate that the proposed hybrid model effectively utilizes additional information on network connections between articles and semantic features of headlines. When increasing the number of news articles per tag from 1,000 to 3,000, a significant improvement in recommendation accuracy was observed: precision rose from 0.168 to 0.187. This 11% increase, while maintaining reasonable training time (increasing from 31.8 s to 506 s), confirms the high payoff of data expansion and the strength of the hybrid approach.

Thus, the hybrid architecture exhibits a steady increase in recommendation quality with the growth of the input data volume, and the additional training time is compensated by a noticeable gain in accuracy—the trade-off between speed and quality remains justified. The ability to accumulate additional connections in the graph component enhances its capacity to identify thematically relevant articles and reduces the effect of "noisy" matches.

Overall, the achieved precision on the test set confirms that integrating semantic headline analysis and a tag-based network model enables the hybrid system to provide more accurate and well-founded recommendations tailored to user interests.

REFERENCES

[1] Chuhan Wu, Fangzhao Wu, Yongfeng Huang, and Xing Xie. Personalized News Recommendation: Methods and Challenges. ACM Transactions on Information Systems, 41(1), 2023.

[2] Sabrine Ben Abdrbbah. A Novel Recommendation Approach For Groups Based On Aggregating Top-k Lists. Procedia Computer Science, 225:3067–3076, 1 2023.

[3] Lingfei Wu, Yu Chen, Kai Shen, Xiaojie Guo, Hanning Gao, Shucheng Li, Jian Pei, and Bo Long. Graph Neural Networks for Natural Language Processing: A Survey. Foundations and Trends in Machine Learning, 16(2), 2023

[4] Qiang He, Xinkai Li, and Biao Cai. Graph neural network recommendation algorithm based on improved dual tower model. Scientific Reports 2024 14:1, 14(1):1–13, 2 2024.

[5] Corpus of news articles of Lenta.Ru - https://github.com/yutkin/Lenta.Ru-News-Dataset

[6] Pandas - https://pandas.pydata.org/

[7] Matplotlib: Visualization with Python- https://matplotlib.org/

[8] Scikit-learn machine learning in python - https://scikit-learn.org/stable/

[9] NetworkX documentation - https://networkx.org/

[10] Itertools documentation - https://docs.python.org/3/library/itertools.html

[11] Numpy - https://numpy.org/