DOI: 10.17323/2587-814X.2025.1.7.21

Recommendation system model based on technical events

Kirill I. Pashigorev
E-mail: kipashigorev@sberbank.ru

Andrei O. Reznikov
E-mail: aoreznikov@sberbank.ru

PJSC Sberbank, Moscow, Russia

Abstract

Recommendation systems are widely used in the commercial field. The algorithms and architectures of recommendation systems are similar in various fields of application and have proven their effectiveness. Recommendations are based on the user's profile, the manner of his behavior on various IT (Information Technology) resources, as well as on similar users. At the same time, the use of recommendation systems in specialized areas is not widespread. Technology divisions are a promising new area of application for recommendation systems, and IT experts themselves will be the users. The purpose of this article is to consider a combination of a recommendation system, machine learning (ML) and LLM (Large Language Model) and to design these tools in a single system. Data volumes are currently measured in petabytes (10^{15} bytes) and exabytes (10^{18} bytes). In order to process even technical information (metadata/technodata) from the surrounding IT landscape, from the IT systems used by experts, AI (Artificial Intelligence) agents are needed. This article provides a literature review regarding the use of recommendation systems in combination with LLM applications, and suggests an application architecture model that generates human-readable news from technical event logs. The system is designed for a group of users who work with big data (ML engineers, data analysts, and data researchers). It is a combination of recommendation system technologies, LLM, and machine learning models. The article also provides the first results of the research that was carried out.

Keywords: recommendation system, matrix factorization, prompt engineering, LLM, AI agents

Citation: Pashigorev K.I., Reznikov A.O. (2025) Recommendation system model based on technical events. *Business Informatics*, vol. 19, no. 1, pp. 7–21. DOI: 10.17323/2587-814X.2025.1.7.21

Introduction

Today we are witnessing the widespread creation and use of AI agents. Tools such as recommendation systems, expert systems and voice assistants have become commonplace. But to effectively solve practical problems, the isolated use of individual tools is no longer effective, so system architectures are becoming more complex, and organizations are designing integrated systems to solve complex problems. The cases of designing complex systems will be discussed later in section 1.

In terms of the combination of technologies, we can single out currently popular AI agents - digital assistants for performing a set of tasks: several AI agents will be able to interact with each other and autonomously perform complex tasks without human intervention [1, 2], and be his unconditional assistant. The implementation of AI agents in all spheres of life, including human processing of large amounts of disparate data using smart assistants, is an urgent task for the coming years.

Recommendation systems have become a universal tool used in various fields, including those unrelated to the Internet: healthcare, education, logistics, etc. The introduction of recommendation systems in Technology departments is promising. The emergence of powerful language models based on the transformer architecture (LLM) has opened up a new approach to solving the problem of processing large amounts of data. Using LLM, it became possible to extract relevant information from large amounts of technical data, which was previously a task requiring significant human resources; identify target audiences among employees of the Technology department who may be interested in specific events or information; work with various types of professional information, including metadata, technological events, news from disparate channels and chats, articles, professional meetings, etc., and combine them all in a single interface. By focusing on the professional interests and competencies of employees, the system so designed will be able to provide more accurate recommendations and focus on relevant technical events without distracting the employee with uninteresting events that are outside the scope of his professional activity.

The use of LLM for log analysis was discussed in [3] and [4]. The authors consider a model for reducing the number of anomalies by analyzing the logs. At the same time, in the approach considered by the authors, there is a limitation on the composition of fields – the so-called weak adaptivity property, and, as a result, the need to retrain LLM in case of changes in input data structures. That is, technical data were used for analysis and forecasting and have limitations on input data structures. With the development of LLM, new scenarios for working with technical data appear, namely, notifying the end user about relevant events in the IT landscape. The authors also explicitly point out another drawback of the models working with logs – this is insufficient interpretability. When log-based anomalies are detected, interpreted results are crucial so that the administrator and analysts can trust and act on the automated analysis. The system model proposed in this article will allow LLM to be integrated into the industrial development process, ensuring that information is not lost and will be delivered to the recipient in a timely manner, as well as unambiguously interpreted. This approach is a new way of applying LLM to existing technical data.

The proposed system model also assumes the formation of a user's portrait for the correct selection of recommendations. To do this, concepts such as a user's portrait or user profile are introduced into the terminology. The users of the system are data experts – ML engineers, data analysts, and data researchers who are located within a diverse IT landscape, hundreds and thousands of systems of which are regularly updated, and in which tens of thousands of events occur regularly every month that need to be monitored.

1. Recommendation systems in large dynamic IT landscapes for metadata filtering

Recommendation systems are used in various fields, which include both commercial activities – e–commerce, distance education and entertainment platforms – these systems contribute to the selection of goods, services and content that match individual preferences and behavioral characteristics of users. They are increasingly being used to solve technical problems in the fields of security and construction, as a tool generalization of large texts, etc.

The review article [5] states that the key element used in modern information technology is data analysis using artificial intelligence technologies. The author considers machine learning, neural networks and natural language processing as the main elements of artificial intelligence that are used for data analysis.

In the article [6], the author describes hybrid systems in which various approaches and algorithms are combined to achieve more accurate personalized recommendations within the framework of building recommendation systems. It also describes the advantage of this combination over collaborative filtering and content-based recommendation models. At the same time, the author claims that hybrid systems solve the problems of cold start and aggregation of information from various sources. In the article [7], the authors additionally classify hybrid systems into monolithic, mixed, and ensemble models, defining monolithic recommender systems as a set of parts of various types of recommendation algorithms, and mixed recommender systems as a combination of the results of all the recommender systems included in it. At the same time, the complexity of developing such systems is highlighted, since significant resources and efforts are required.

In the article [8], the authors also use generalization methods for the purposes of their research and apply them to the input stream of social network messages numbering in the millions. The authors consider the case that the length of a generalized topic is a controlled variable, and further express the opinion that automatic generalization is the task of creating a consistent abbreviated version of the document, which outlines its main provisions. In this case, depending on the chosen use case, the target length of the final result can be selected relative to the length of the input document or can be limited.

The authors of the article [9] consider the use of LLM applications in construction to compile automated reports based on technical information, and also cite resource optimization as one of the results obtained. The authors use the term "intellectualization of construction inspection" in their work, and also point out the shortcomings in research that currently construction inspection mainly relies on manual execution and analysis.

The authors of the article [10] consider the use of a combination of machine learning, LLM and generative AI technologies. Similar to previous works, the authors pursue the goal of optimizing the working hours of experts in a specific field, and achieve it using modern technologies. The authors conduct research using the GPT-4 model (Generative Pre-trained Transformer 4) and mention the risk of hallucinations as LLM responses, as well as identify non-determinism (different responses during different sessions) as an additional level of complexity and unpredictability in the process of generating responses.

The authors of the article [11] also consider a combination of LLM applications and RecSys (Recommender Systems) to solve their problem and especially focus on the problem of cold start, considering various options. The authors call their system A-LLM-Rec (All-round LLM-based Recommender system), because the main idea is to allow LLM to directly use the collective knowledge contained in a pre-trained modern recommendation system based on collaborative filtering (CF-RecSys, Collaborative Filtering), so that new LLM features can be shared, as well as high-quality representations of users and products that are already trained in the modern CF-RecSys system. But the experiment is still being conducted not on technical data, but with human-readable headings and descriptions.

The authors of the article [12] position their approach as innovative. They describe in detail the application of an LLM-based multi-agent architecture, which uses the following chain of agents: Perceiver – Learner – Performer – Critic – Thinker. The architecture uses a Learn – Act – Criticize cycle and a reflection mechanism to increase the effectiveness of user interaction. As in the previously reviewed articles, the authors of this article focus on cold start. At the same time, the focus is on the balance between the accuracy of recommendations and user satisfaction. The experiment was also conducted on human-readable data. In general, it is the LLM component (consisting of small agents/modules) that is presented as innovation.

In the sources considered, much attention is paid to large language models, the architecture of which can be found in detail in [13]. It should also be noted that the creation of intellectual assistants in various subject areas is actively considered not only by foreign authors and researchers, examples of which are given in sufficient detail in this section, but also by domestic authors and researchers [14, 15]. The difference between the system being designed in this study is that the system we designed does not have business goals such as audience retention or increased content consumption, but rather aims to notify the user of changes in the company's information infrastructure that are relevant to their specifics and objectives. For example, the release of a new data product related to the user's projects, changes in the data relevant to him, the release of releases of systems relevant to the user, changes in metadata, etc. Let's highlight two problems that were not fully solved in [4] – weak adaptability of models and insufficient interpretability of results, and further in this article we will consider ways to solve these problems, among others.

2. Architecture of the Techno Events Smart News feed system

The primary information about an event is technical data generated by other systems (event logs), which are difficult for humans to perceive. Our task is not only to find an event that is relevant to the user, but also to bring it to a form that is easily perceived by humans. Due to the wide range of events being processed, a universal solution for bringing event information to a user-friendly form is to implement LLM to summarize technical data in a news notification containing all the key aspects of the event. In addition, to better match users to a specific event, we will use tagging to highlight key properties.

To define the context of the study, we will introduce the term "technodata": These can be thousands of events taking place in hundreds of IT landscape systems. *Table 1* shows a simple example of the structure of such events.

To solve the research problem, an intelligent system for processing a large volume of events was designed, the architecture of which is shown in *Fig. 1.*

Let's consider the main elements of this system.

- Events generated in IT landscape systems are stored in the DBMS.

Table 1.

Example of an event table

id	Date	Event's type	Description	...	Author	Source
uuid	Date	String	Text	...	Varchar	Varchar

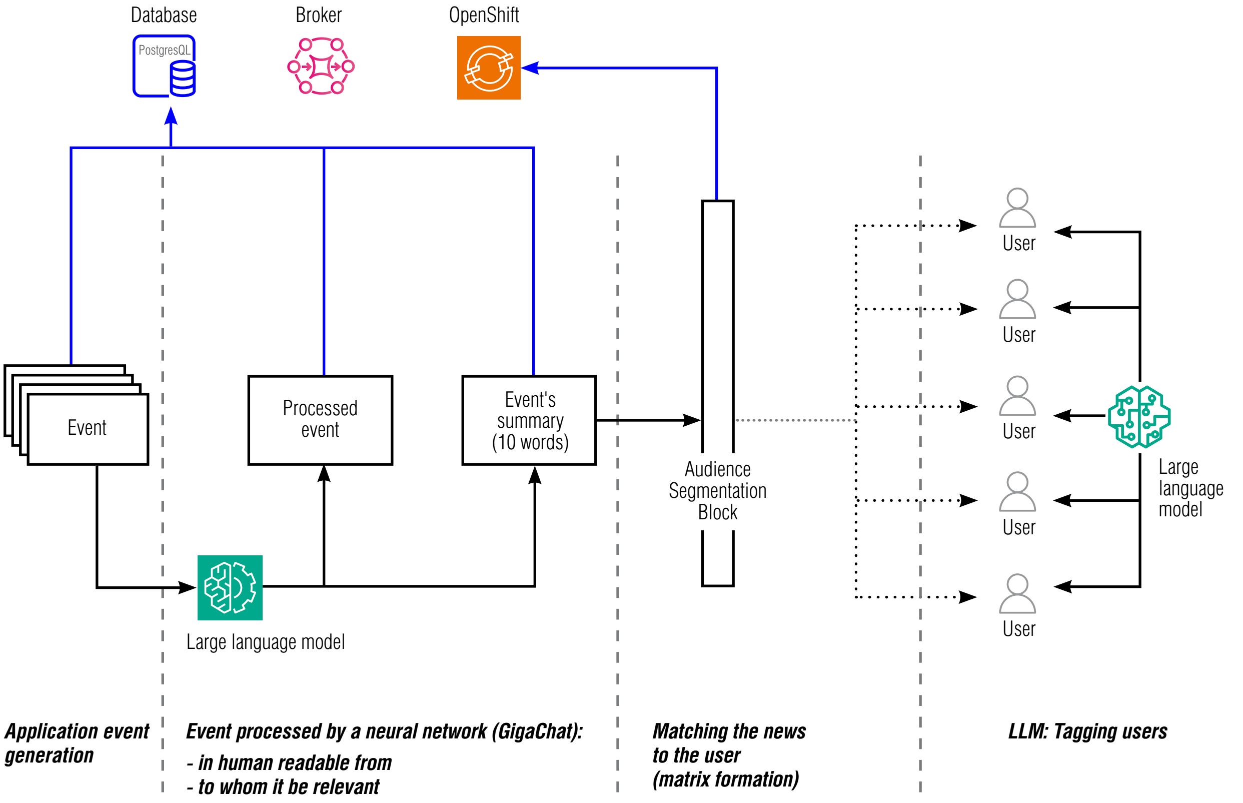

Fig. 1. The main elements of an intelligent system for processing a large volume of events.

- The Giga-intgr microservice sends technical event data to LLM. The response from LLM (processed events) in the form of generated news and assigned tags is recorded in the DBMS.
- The Jira-intgr microservice requests a list of user tasks in the Task Accounting system (Jira) and transmits it to Giga-intgr using a message broker. Giga-intgr transmits the received message to LLM. The response from LLM (user tagging) in the form of tags for each user is recorded in the DBMS.
- The SegmentOfAuditory-serv service generates a matrix of user-news correspondence using a model.
- The homepage service receives news for display in the feed from the DBMS via the API.

The overall result of using LLM in the presented architecture model is the ability to solve the following key tasks:

- Tagging users, i.e., assigning them appropriate tags reflecting their interests and preferences.
- News tagging, which allows you to classify news materials into various categories and topics.
- Forming a short news content based on technical information so that the user can quickly familiarize himself with the content of the message without having to read the entire text.
- The recommendation system has to solve the problem of selecting interesting news for each specific user based on his personal interests and preferences.
- The system architecture presented is able to take

into account a large number of the following parameters:

- A large number of information systems as sources, which can be measured in the range from one to several thousand.
- A large number of events with different structures, which can also vary depending on the event or the source system, as well as over time; the number of events can be measured in hundreds of thousands, and the number of attributes in the structure of these events can vary from two to several hundred.
- A large number of user roles and an even larger number of users.

The architecture of the system is shown in *Fig. 2*, and a description of the technical components is given in *Table 2*.

The architecture is designed in a micro-service style and assumes placement in a dynamic infrastructure on an OpenShift cluster in order to increase the level of horizontal scaling.

3. Optimizing work with technical events

3.1. Setting the task of recommending news

In order to preserve the coherence of the presentation and to formalize the problem, we present a

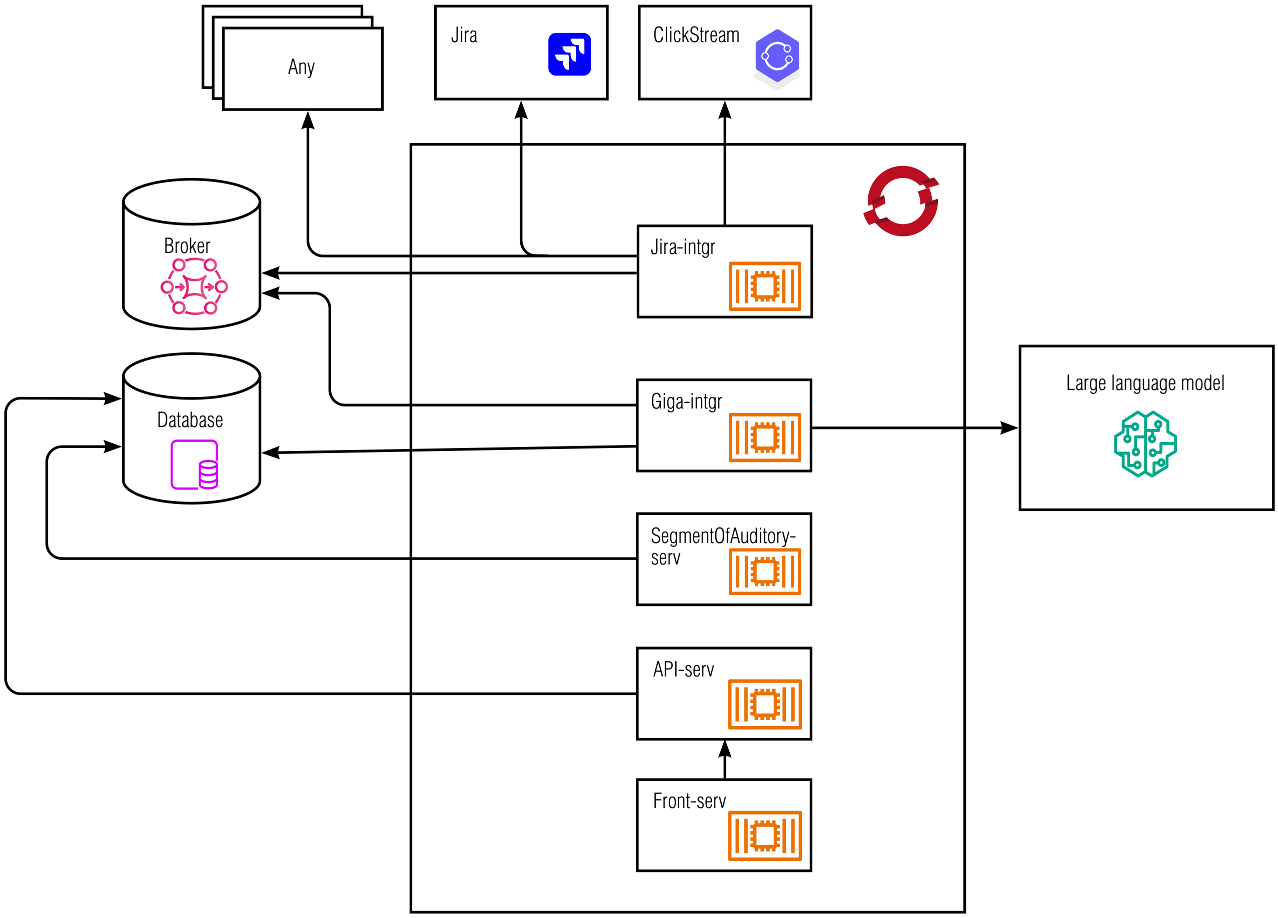

Fig. 2. Component diagram.

Table 2.

System components

Component	Description
Large language model	The Giga Chat API technology will be used as a means of accessing the large language model [16].
Any, Jira, ClickStream	These components are external data source systems. Any arbitrary system can act as a source.
DBMS	A stateful component for storing data in a structured form.
Kafka Broker	A stateful component for implementing interaction between microservices.
Giga-intgr	A component (microservice) for implementing interaction with a large language model. Implemented in Python.
Jira-intgr	A component (microservice) for implementing data acquisition from the surrounding landscape. Implemented on NodeJS.
SegmentOfAuditory-serv	This component (microservice) is used to select, transform, combine, and other ways to prepare data to find useful patterns, as well as to extract patterns from the data obtained.
API-serv	The API for the main page of the feed is implemented in NodeJS.
Front-serv	The main page of the feed (GUI) is implemented in JavaScript – React.

description of the matrix factorization method, which was repeatedly and in detail discussed in [17–19], and is a decomposition of the source matrix into the product of two other matrices of lower rank. In the study described in this paper, the final matrix R is decomposed into matrices A and B: a set of employees is considered $U = \{u_1, u_2, ..., u_m\}$, multiple tasks $T = \{t_1, t_2, ..., t_n\}$ and multiple news $N = \{n_1, n_2, ..., n_k\}$. Employee task matrix $A = (a_{ij}) \in \{0;1\}^{m \cdot n}$ is defined in such a way that $a_{ij} = 1$, if the u_i employee is engaged in the task t_j, and $a_{ij} = 0$ otherwise. News matrix $B = (b_{ij}) \in \mathbb{R}^{n \cdot k}$ represents the relevance of the n_i news to the t_j task. The purpose of the study is to find the recommendation matrix $R = (r_{ij}) \in \mathbb{R}^{m \cdot k}$, where r_{ij} – is the degree of relevance of n_j news to the u_i employee.

Assessment of the relevance of news for tasks is performed by calculating the degree of relevance of b_{ij} for each task t_j and news n_i using BERT text analysis methods. The relevance of news for a u_i employee is assessed by calculating the degree of relevance of r_{ij} for each u_i employee and n_j news using the employee's task matrix A and the news matrix B. The ranking of news for each u_i employee is based on the degree of relevance of r_{ij} in descending order.

This approach can be represented using the following formula:

$$r_{ij} = \sum_{l=1}^{n} a_{il} \cdot b_{jl},$$

where a_{il} is the degree of participation of the u_i employee in the t_l task, and b_{jl} is the degree of relevance of the n_j news to the t_l task.

Restrictions are introduced on the research task: a limit on the maximum number of recommendations L for each employee and the establishment of a minimum degree of relevance r_{min}, below which the news is not recommended to the user.

The following criteria are used to evaluate the model used: accuracy (Precision), completeness (Recall) and F1-score. The goal is to maximize these metrics to provide the most relevant recommendations for each employee. The combination of evaluation criteria is presented in *Table 3.*

This assessment will determine how boldly the model should select recipients. The initial assessment is first performed by a human.

It is important to note that when training the model in the course of the study, an assumption will be made about the confidence interval: if there are no errors of the first kind and there is an acceptable number of errors of the second kind, then the same picture will be preserved for the entire set (sample). Errors should be aggregated by the reviewer and a decision is made whether further training of the model and correction of the error are required. Errors of the second kind are corrected by promptness, errors of the first kind require additional training.

It is also necessary to take into account the need to solve the "cold start" problem [20], which occurs once when the system is started.

At the presented stage of the study, the assumption is made that technical news is offered to users randomly. To evaluate the quality of the predictions, the set of estimates V was divided into separate sets V_{train} for training and V_{test} for testing. The tests for the model will be performed on quantitative indicators, which are presented in *Table 4.*

The characteristics presented in the *Table 4* are relevant to the specifics of this work:

Table 3.

The criteria matrix

The news should be recommended and indeed got into the recommendation (TP)	The news should be recommended, but it doesn't actually make it into the recommendation (FP)
The news should not be recommended, but it actually gets into the recommendation (FN)	The news should not be recommended, and indeed does not get into the recommendation (TN)

Table 4.

Quantitative characteristics of the experiment

Characteristic	Value	Characteristic	Value
Number of "DP Creation" events	40000	Training sample	75%
Number of "MD Change" events	10000	Test sample	25%
Number of DM application releases	10–1500, including child tasks	The rank of the matrices	the value of X will be accepted
Number of DSM application releases	10–1500, including child tasks	Factorization steps	the value of Y will be accepted
Number of users	50	Range of values of the original matrix	{0, 1} – implicit feedback

- DP (data product) is a type of information asset that is a structured description of a set of industrial data (type and composition of data, data sources and delivery methods) available for ordering and receiving through industrial data distribution paths for use in the interests of the Bank's products and the Ecosystem.
- CAP (Corporate Analytical platform) is the general name of the solution responsible for obtaining data, processing it, and providing the processed data to interested parties. A centralized platform for data collection (loading), delivery (unloading), processing, integration, research and analysis. The CAP consists of a Core and a User Space.
- MD (Data Manager) is an employee of the Bank, assigned to the role in accordance with the Data Management Policy (depending on the destination).
- DM App (Data Map) is a system responsible for managing data products: their attributes, structure and accessibility to distribution in the Data Supermarket App.
- DSM App (Data Supermarket) is a single portal for interaction with the CAP which allows you to study and order data from the CAP. It is intended for distributing data in the CAP from replicas of industrial sources.

At the initial stage of the experiment, more than 95% of the cells of the matrix V will be unfilled, i.e. the matrix will be very sparse.

The standard deviation (*RMSE*) will be used as a metric [21]. *RMSE* will allow you to estimate the proportion of news that should have been shown to a specific group of users and were shown:

$$RMSE = \sqrt{\frac{1}{N-d}\sum_{i=1}^{N}\left(M_i - O_i\right)^2},$$

where

M_i is the predicted value for the i-th observation in the dataset;

O_i is the observed value for the i-th observation in the dataset;

N is the sample size;

d is the degree of freedom in the model configuration. For a linear fit $d = 2$.

3.2. LLM and prompt engineering

For the purposes of our research, we will use the already trained GigaChat language model [22], and focus on prompt engineering to achieve the desired result without further training the model. The meaning of these terms is presented in [23]: prompt engineering is the process of creating efficient and accurate tools for working with large language models (LLM). Prompt is a text description of a task that needs to be completed using an AI model. A query that is set by the model to generate text, images, code, or other types of content.

The following elements will be used to transfer it to a large language model:

- Summary type prompt – highlighting the main theses from the text, keywords for these types of prompt: reduce, summarize;
- Prompt with the type "Generation", keywords: write, compose, invent;
- Role – for acting as a certain character;
- The Zero-shot method does not contain any response examples, the response of a large language model will be received in free form.

A preliminary list of event log attributes to be transmitted to a large language model based on which the news will be generated:

- FB tribe;
- Tribe;
- DP name;
- Data product;
- Category;

- Data product type;
- Cluster;
- Schema Platform;
- Database schema;
- Number of tags;
- Tag name;
- Tub. MD DP number;
- UUID;
- Latest status;
- MD assignment date;
- Publication flag;
- Name of the receiving system;
- Distribution platform;
- SLA and others.

A tribe is a group of cross-functional teams working on a product or service area. Each team includes specialists of all the necessary profiles to create a turnkey product.

Thus, based on the attributes of the "Date Product Change" event:

- the news should appear: *"The W showcase for the online assistant has been launched in Block Bl"*;
- the following tags should be assigned to the news: *{"Focus Group": "Block Bl"}, {"Topic": "Showcase W for online Assistant"}, {"Category": "Block Bl News"}.*

When preparing prompt, it is important to make sure that there is no hallucination effect [16, 24]. Also, a mandatory point in the preparation of the prompt is the task of mandatory passage of the Censor's check. There may be cases when the Censor regards a correct request to a large language model as undesirable for discussion and returns a response about the inappropriateness of this topic with a request to change it to a correct one.

Based on the results of the primary study, the following results were obtained, presented in *Table 5.*

Table 5 shows a comparison of the distances between objects and the results of generation based on them.

- Prompt 1: *"Generate a clear and concise news article summarizing the key details of the event described in the technical data provided. Focus on presenting factual information in a human-readable format, avoiding any speculation, assumptions or excessive embellishments. Make sure that the article has a clear and logical structure, uses the correct grammar and syntax of sentences. Use a neutral tone and avoid sensationalistic language.*

Table 5.

Results of the primary research

The distance between the prompts	Prompt 2	Prompt 3
Prompt 1	0.03732394628917002	0.095708435241878
Prompt 2	0	0.0855076122459778
The average distance between the news generated using these prompts	AI-news of prompt 2	AI-news of prompt 3
AI-news of prompt 1	0.15939979460493073	0.1789409953210833
AI-news of prompt 2	0	0.13067957637939245

The goal is to provide an informative and objective overview of the event, making it easy for readers to understand."

- Prompt 2: *"Generate a short and informative news article summarizing the key details of the event described in the provided technical data. Focus on presenting factual information in a neutral tone, avoiding any speculation, speculation or emotional language. Use a clear and logical structure and avoid sensational or attention-grabbing language. The goal is to provide an objective summary of the event so that readers can quickly understand what happened."*
- Prompt 3: *"As an expert in making informative notifications, make up a short message no more than 10–12 words long that will provide key information about the event."*

From the data in *Table 5*, it can be seen that even products that are similar in structure and content can produce results with significant deviations from each other. This behavior can be compared with the behavior of a rigid system of ordinary differential equations (ODES). Based on the fact that the project uses an already trained LLM without the global need for further training, as well as on the basis of the results obtained, it is concluded that a clear formulation of the plan is necessary to obtain a stable result.

Based on the study of the behavior of the model with various prompts, three prompts were selected, the data on which were given in *Table 5*. With the help of these data, a test sample of news was generated, which was provided for markup on the TagMe platform [25] to the target group of users of this product, in order to select the optimal informative and easy-to-understand version of the generated news. As a result, the second prompt was selected for use in the pilot version of the product.

3.3. Cold start and first results

Considering that the users of this service will be employees of our company, it is advisable to think about how to "warm up" the cold start for new users in order to reduce the funnel of convergence of the system to the recommendation of the most relevant news. Since our target audience exists in the same IT landscape, it is possible, in addition to collecting technical data, to also collect digital traces of user work. The first and most obvious trace is the tasks that were started by the user, and which the user started himself. Based on this, we can immediately hypothesize that the tasks associated with the user reflect the type of his professional activity, and therefore, comparing the news in proximity to the user's tasks in Jira, we can assume how relevant they are to the user. Having received the user's tasks and having constructed embeddings for the tasks that the user was engaged in, and embeddings for news and technical data, we can build a matrix of proximity (cosine distances) between tasks and news. This way we can get primary information about which news is most likely to overlap with the user's professional activities.

After constructing the correlation matrix, the task arises of displaying it in a ranked list of news for recommendation to the user. The simplest approach would be to calculate the average correlation for each news item and sort it by the values obtained. However, in the course of the study, another method was chosen, the algorithm of which looks like this: for each news item, the top N *(*$N = 5$*)* news items are selected for each task, each news item is assigned a weight equal to *(*$N - i$*)*, where i is the number of news in the top for each of the tasks; next, all the weights for each news item are summed up and sorted according to the values obtained. The peculiarity of this approach is that not all news gets into the final ranking, however, the distribution of news occurs in a more "honest" way due to the fact that some tasks are poorly correlated with both relevant news for the user and other user tasks. This distribution allows, firstly, to reduce the impact of outliers and make news ranking more resistant to abnormal user activity, while maintaining relevant news for abnormal activity, and, secondly, to exclude irrelevant news from the ranked sample, which in the future will allow more complex models to be used more confidently on such a filtered sample news. But at the moment of the cold start, we can simply output K (the number of news recommended to a particular user) of the first news from this list.

Thus, when the user logs in for the first time, the system already has primary data on the specifics of the user's work, based on his tasks obtained from the task accounting system. The system calculates embedding based on the user's tasks which will later be used to find the cosine distance with news embeddings. Next, the user is invited to read the recommendations that the proposed model has calculated for him in the news feed.

The advantage of using the proposed system model is the exclusion of the use of classical conditional algorithms to determine the target groups of users. In the proposed system model, this function will be performed by the recommendation system using the cosine similarity algorithm and the overlap matrix. The way a large number of event types are handled by classical methods using conditional operators does not produce the desired result. Meanwhile, using a language model to categorize events and users allows you to process these types of events quickly and optimally. It is also worth considering that the same type of event, but with different input characteristics, such as, for example, a logging object, may be relevant for a specific user in one case, but in another case it will no longer be relevant with a different logging object. Processing all possible combinations of input parameters using conditional operators is an expensive operation from a technical point of view.

Conclusion

This article presents the concept of an event management system architecture and a model for routing events in the news format to specific users, as well as defines criteria for evaluating the application of the model. This approach uses LLM to convert raw technical data into short news, which is then delivered to users through a recommendation system. Such an intelligent system design combines neural network technologies, recommendation systems and machine learning to minimize the effect of spam and notify users in a timely manner. The result of automating the news generation process will be a reduction in the time spent by an expert searching for information, and as a result, the risk of critical incidents is minimized. The proposed architecture of the software system implements the interaction of unrelated components, combining them into a single AI agent, minimizing the flow of news to a single user. The proposed architecture also allows for further development of the system at the lowest cost by integrating speech recognition components into the system, which will make the system a full-fledged AI assistant. The combination of these technologies is an indispensable assistant to support experts in their daily data-related tasks.

Based on the results of the first application of the proposed model to the technical data that had been accumulated historically, the first result was obtained: usually the problem of incident detection was solved on such data. But in the new realities, when the rate of change in the IT landscape is regularly increasing, we have to look for new ways to apply LLM to technical data.

Using the example of the system model presented in this article, more than 0.5% of useful data was extracted from the total volume of messages, and more than 1% of relevant consumers of this data were found. This result also reduces the time spent by employees on tracking changes in the IT landscape. Unlike classical approaches to solving the problem of detecting anomalies based on logs presented in [3] and [4], the approach proposed in this article focuses not on log analysis, but on anticipating potential incidents due to sensitive changes in the IT landscape, in products consumed by user systems, due to notifying users about this in a timely manner. The correctness of our chosen direction is also confirmed by the hypothesis from [4] that a high percentage of false positives can lead to missing important failures in the system, and a high percentage of false passes can lead to a waste of developers' efforts. Our proposed approach allows us to prevent failures and costs for developers' efforts.

We also obtained a metric of the final relevant amount of information in the amount of 0.05–0.15% for the test sample of employees and a reduction in the volume of targeted information while maintaining informativeness to no more than 10% of the initial volume. The average volume of incoming material decreased from 8629 characters to 189–538 characters, which reduced the amount of information consumed by more than 23 times while maintaining 96% of the semantic load. ■

Acknowledgments

The authors would like to thank the Sber Department of Research and Innovation for hosting the Techno-Idea pitch session and the opportunity to conduct research, the Sber Department of Data Dissemination and the Department of Artificial Intelligence and Machine Learning Technology Development for providing resources for research, as well as Associate Professor of the HSE Faculty of Computer Science, Candidate of Pedagogical Sciences S.A. Videnin for methodological recommendations when writing the article.

References

1. Microsoft (2024) *What are AI agents?* Available at: https://learn.microsoft.com/ru-ru/azure/cloud-adoption-framework/innovate/best-practices/conversational-ai (accessed 20 July 2024).

2. Sber (2024) *A platform for launching autonomous AI agents is presented*. Available at: https://ai.sber.ru/en/post/predstavlena-platforma-dlya-zapuska-avtonomnyh-ai-agentov (accessed 20 July 2024).

3. Shah A., Pasha D., Zadeh E., Konur S. (2022) Automated log analysis and anomaly detection using machine learning. *Frontiers in Artificial Intelligence and Applications*, vol. 358: Fuzzy Systems and Data Mining, pp. 137–147. https://doi.org/10.3233/FAIA220378

4. Chen Z., Liu J., Gu W., et al. (2021) Experience report: Deep learning-based system log analysis for anomaly detection. *arXiv:2107.05908*. https://doi.org/10.48550/arXiv.2107.05908

5. Mokshanov M.V. (2024) The use of artificial intelligence in data analysis: an overview of the current state and future directions. *Universum: technical sciences: electronic scientific journal*, no. 5(122) (in Russian). https://doi.org/10.32743/UniTech.2024.122.5.17513

6. Eremin O.Y. (2023) Methods of implementation of hybrid recommendation systems. *E-Scio*, no. 3(78) (in Russian).

7. Kurennykh A.E., Sudakov V.A. (2022) Approach to the development of hybrid recommendation systems. *Bulletin of Science and Practice*, vol. 8, no. 11 (in Russian).

8. Völske M., Potthast M., Syed S., Stein B. (2017) TL;DR: Mining Reddit to Learn Automatic Summarization. Proceedings of the *Workshop on New Frontiers in Summarization, Copenhagen, Denmark, 2017*, pp. 59–63. Association for Computational Linguistics. https://doi.org/10.18653/v1/W17-4508

9. Pu H., Yang X., Li J., Guo R. (2024) AutoRepo: A general framework for multimodal LLM-based automated construction reporting. *Expert Systems with Applications*, vol. 255, part B, article 124601. https://doi.org/10.1016/j.eswa.2024.124601

10. Sivakumar M., Belle A.B., Shan J., Shahandashti K.K. (2024) Prompting GPT–4 to support automatic safety case generation. *Expert Systems with Applications*, vol. 255, part C, article 124653. https://doi.org/10.1016/j.eswa.2024.124653

11. Kim S., Kang H., Choi S., et al. (2024) Large Language Models meet Collaborative Filtering: An efficient all-round LLM-based recommender system. *arXiv:2404.11343.* https://doi.org/10.48550/arXiv.2404.11343

12. Shu Y., Zhang H., Gu H., et al. (2023) RAH! RecSys-Assistant-Human: A human-centered recommendation framework with LLM agents. *arXiv:2308.09904.* https://doi.org/10.48550/arXiv.2308.09904

13. Vaswani A., Shazeer N., Parmar N., et al. (2017) Attention is all you need. *arXiv:1706.03762.* https://doi.org/10.48550/arXiv.1706.03762

14. Morozevich E.S., Korotkov V.S., Kuznetsova E.A. (2022) Development of a model for the formation of individual educational trajectories using machine learning methods. *Business Informatics*, vol. 16, no. 2, pp. 21–35. https://doi.org/10.17323/2587-814X.2022.2.21.35

15. Palchunov D.E., Yakobson A.A. (2024) Development of an intelligent assistant for the selection of goods in the process of dialogue with the user. *Business Informatics*, vol. 18, no. 1, pp. 7–21. https://doi.org/10.17323/2587-814X.2024.1.7.21

16. Amenitsky A.V., Rukhovich I.V., Amenitskaya L.A., et al. (2024) Side effects of hallucinations of artificial intelligence. *Science, innovation, education: current issues and modern aspects*, pp. 224–235. Penza, 2024 (in Russian).

17. Strömqvist Z. (2018) *Matrix factorization in recommender systems: How sensitive are matrix factorization models to sparsity?* Uppsala University Publications. Available at: https://uu.diva-portal.org/smash/get/diva2:1214390/FULLTEXT01.pdf (accessed 22 July 2024).

18. Moisyuk-Dranko P.A., Revotyuk M.P. (2020) Methods of matrix factorization for recommendation systems. Proceedings of the international scientific conference *Information technologies and systems 2020 (ITS 2020)*, pp. 193–194. Minsk: BGUIR (in Russian). Available at: https://libeldoc.bsuir.by/bitstream/123456789/41339/1/Moysyuk_Dranko_Metody.pdf (accessed 22 July 2024).

19. Kuznetsov I.A. (2019) *Methods and algorithms of machine learning for preprocessing and classification of weakly structured text data in scientific recommendation systems*. Moscow: MEPhI (in Russian). Available at: https://ds.mephi.ru/documents/90/Кузнецов_И_А_Текст_диссертации.pdf (accessed 22 July 2024).

20 Yuan M., Lin H.-T., Boyd-Graber J. (2020) Cold-start active learning through self-supervised language modeling. *arXiv:2010.09535.* https://doi.org/10.48550/arXiv.2010.09535

21. Liemohn M.W., Shane A.D., Azari A.R., et al. (2021) RMSE is not enough: Guidelines to robust data-model comparisons for magnetospheric physics. *Journal of Atmospheric and Solar-Terrestrial Physics*, vol. 218, article 105624. https://doi.org/10.1016/j.jastp.2021.105624

22. Sber (2024) *GigaChat API* (in Russian). Available at: https://developers.sber.ru/portal/products/gigachat-api (accessed 22 July 2024).

23. Sber (2024) *Prompt engineering* (in Russian). Available at: https://developers.sber.ru/docs/ru/gigachat/prompt-engineering (accessed 26 December 2024).

24. Amenitsky A.V., Rukhovich I.V., Amenitskaya L.A., Amenitsky D.A. (2024) Causes, ethical problems and prevention of hallucination LLM. *Intelligence. Collection of articles of the International Competition of Young Scientists*. Penza, pp. 12–15(in Russian).

25. Sber (2024) *TagMe Data Markup Platform* (in Russian). Available at: https://developers.sber.ru/portal/products/tagme (accessed 09 December 2024).

About the authors

Kirill I. Pashigorev

Head of the Department, SberData, PJSC Sberbank, 25A bld. 6, Warsaw Highway, Moscow 117105, Russia;
E-mail: kipashigorev@sberbank.ru
ORCID: 0009-0008-3478-4874

Andrei O. Reznikov

Chief Development Engineer, SberData, PJSC Sberbank, 25A bld. 6, Warsaw Highway, Moscow 117105, Russia;
E-mail: aoreznikov@sberbank.ru