**THE USA JOURNALS**
THE AMERICAN JOURNAL OF ENGINEERING AND TECHNOLOGY (ISSN – 2689-0984)
**VOLUME 06 ISSUE08**

**PUBLISHED DATE: - 23-08-2024**
**DOI: -** https://doi.org/10.37547/tajet/Volume06Issue08-08 **PAGE NO.: - 63-72**

**RESEARCH ARTICLE** **Open Access**

# UTILIZING A SCALABLE AI/ML-BASED DATA ANOMALY DETECTION TOOL TO IMPROVE DATA QUALITY IN VIDEO STREAMING SERVICES

**Alexander Motylev**
Director, Data Test Engineering at Paramount Global / PlutoTV, Miami, Florida, United States

**Abstract**
In today's world of online cinemas and streaming platforms such as Netflix and YouTube, data quality plays a key role in ensuring high user satisfaction. A scalable AI/ML anomaly detection tool is used to improve data quality and enhance the reliability of video streams. This paper examines various approaches to anomaly detection, including supervised and unsupervised learning, as well as deep learning methods such as autoencoders and recurrent neural networks. In addition, the application of AI/ML for predicting user behavior and optimizing the resources of video streaming services is analyzed. The introduction of such technologies allows not only to improve the quality of video streams but also to reduce the cost of content moderation and network resource management. The prospects for the development of these technologies and their impact on the video-streaming industry are discussed.

**Keywords** Video streaming, data anomaly detection tools, AI-based data anomalies, Ml-based data anomalies, video streaming services.

## INTRODUCTION

Modern video streaming services, such as Netflix, Hulu, and YouTube, play a key role in delivering content to millions of users worldwide. With the constant growth in data volume and the increasing number of users, maintaining high data quality becomes critically important for ensuring satisfactory user experience and service reliability. The relevance of the data quality issue lies in the fact that poor-quality data can lead to service disruptions, which, in turn, negatively affect users' perception of the platform.

To timely detect anomalies, two main approaches exist: supervised and unsupervised. Supervised methods require labeled data, while unsupervised methods, on the contrary, do not require labeled data and are based on the assumption that anomalies statistically differ from normal samples.

Additionally, deep learning methods, such as autoencoders and recurrent neural networks (RNNs), have shown high effectiveness in detecting anomalies in complex and high-dimensional data. However, their use is associated with high computational costs and the need for significant data to train the models. These limitations underscore the need for developing new approaches that are more efficient and cost-effective.

This work aims to explore AI/ML-based tools for anomaly detection in data to improve data quality in video streaming services.

## APPLICATION OF AI AND ML IN VIDEO STREAMING DATA ANALYSIS

Modern streaming platforms actively integrate artificial intelligence (AI) to optimize their operations and create new opportunities for their audience. Currently, services like Netflix, Hulu, Spotify, YouTube, TikTok, and many others employ AI in their operations. AI algorithms meticulously study users' viewing history, preferences, and interactions with the platform to suggest the most suitable content. The primary goal of AI technologies is to increase the time users spend on the platform and enhance their viewing satisfaction[1].

Actor recognition on screen has become a popular feature among viewers. Users can find out an actor's name and filmography simply by pausing the video. Developers use neural networks that analyze movies and create tags with time points of actors' appearances and facial outlines. Super Resolution technology is used to improve the quality of old films. Neural networks analyze frames, increasing resolution and adding missing details. This allows SD films to be converted to Full HD and 2K films to 4K, enhancing viewers' experience [2].

In terms of anomaly detection, it involves identifying rare events that significantly differ from other data. Machine learning methods are widely used for anomaly detection and can be implemented through:

1. Supervised Anomaly Detection: This method uses a labeled dataset that includes both normal and anomalous examples. The model is trained on this data to classify new observations. Popular algorithms for this task include supervised neural networks, support vector machines, and k-nearest neighbor classifiers.

2. Unsupervised Anomaly Detection: This method does not require labeled data. It is based on two assumptions: only a small percentage of data is anomalous, and anomalies statistically differ from normal samples. Data is clustered based on similarity, and instances that deviate significantly from the majority are classified as anomalies [3].

Deep learning, as a major area of artificial intelligence, has immense potential in the field of computer vision. This machine learning approach, based on the principles of artificial neural networks, allows for modeling complex functions and processing large volumes of data. The process of object detection and analysis using deep models includes several key stages:

Initially, the data containing images with specified classes and object boundaries need to be annotated. The data may require preprocessing, such as resizing, normalization, and augmentation. Then, the model is trained, involving the generation of candidate proposals for objects and their classification. First, the model proposes candidate objects that might be present in the image, then classifies them and accurately localizes the boundaries. After training, the model is tested on a test dataset, evaluating metrics such as detection accuracy, recall, localization precision, and mean average precision (mAP) [4]. Below in Figure 1 is the architecture of video classification.

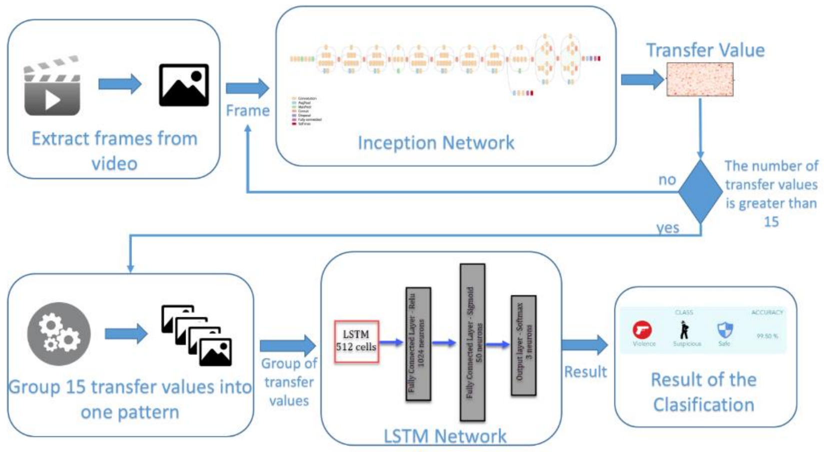
Extract frames from video
Frame
Inception Network
Transfer Value
no
The number of transfer values is greater than 15
yes
Group 15 transfer values into one pattern
Group of transfer values
LSTM 512 cells
Fully Connected Layer - Relu 1024 neurons
Fully Connected Layer - Sigmoid 50 neurons
Output layer - Softmax 3 neurons
LSTM Network
Result
CLASS
ACCURACY
Violence
Suspicious
Safe
99.50 %
Result of the Clasification

Fig.1. Video classification architecture [5]

Finally, the screen can display real-time video classification: every three seconds, a segment of the video is classified as safe, suspicious, or criminal activity.

## PREDICTIVE ANALYTICS AND VIEWERSHIP DATA FORECASTING

Predictive analytics, often referred to as predictive analysis, is a method for forecasting future events based on the analysis of historical data. The process of predictive analytics relies on processing large datasets that traditional analysis methods cannot effectively handle [6]. Predicting user behavior in video streaming involves several key stages:

In the first stage, data is collected about users and their interactions with the platform. This can include data on views, likes, comments, watch time, visit frequency, etc. This data is gathered from server logs, analytical systems, and other sources. Data preprocessing involves cleaning the data from noise and anomalies, normalizing, and transforming it into suitable formats for further analysis. This may include eliminating missing values, removing duplicates, and aggregating data for analysis over different time intervals.

Next, feature engineering takes place, where features that will be used in machine learning models are extracted.

The subsequent stage involves selecting and training machine learning models [7]. The main approaches to predicting user behavior in video streaming include the use of machine learning models such as neural networks, recurrent neural networks (RNNs), and graph neural networks (GNNs). These models help analyze large volumes of user behavior data, including viewing history, time spent on the platform, and interactions with various content. For instance, the STAMP (Short-Term Attention/Memory Priority Model) is used to predict user preferences based on short-term memory and attention [8].

Modern methods of viewership data forecasting are based on the use of machine learning and statistical analysis. Classical statistical models, such as autoregressive integrated moving averages (ARIMA), are widely used for time series analysis. However, with the advancement of technology and the growth of data volume, more complex methods such as neural networks, including recurrent neural networks (RNNs) and long short-term memory (LSTM) networks, have come to the forefront [9]. ARIMA (AutoRegressive Integrated Moving Average) models are powerful tools for analyzing and forecasting time series. The ARIMA model includes three main components:

- Autoregression (AR): Using dependencies between observations and their previous values.

- Integration (I): Differencing the data to eliminate non-stationarity.

- Moving Average (MA): Modeling the dependence of the current value of the series on the errors of previous forecasts [10, 11].

Machine learning (ML) is one of the most powerful tools for viewership data forecasting. Due to its ability to process large volumes of data and identify patterns, ML methods significantly improve forecast accuracy compared to traditional statistical methods [12]. However, to achieve the greatest efficiency, hybrid models are often used. These models are combinations of various methodologies, leveraging the strengths of each applied method. The main idea is to compensate for the weaknesses of one model by utilizing the advantages of another [13, 14].

## PRACTICAL IMPLEMENTATION OF ANOMALY DETECTION AND FORECASTING SYSTEMS FOR VIDEO STREAMING

Anomaly detection and forecasting in video streaming are critical tasks that can significantly improve the quality of service and reliability of video streaming systems. Solving this task without specialized software is practically impossible. These programs must possess high flexibility, performance, and accuracy to adapt to changing data and requirements [15].

First to consider is Apache Kafka, which has become the de facto standard for processing streaming data in recent years. Previously, RabbitMQ, ActiveMQ, and other message queue systems were used, utilizing various message distribution patterns to distribute data between consumers from producers. However, the scale of these tools was limited. Kafka Connect provides a variety of connectivity options, allowing Kafka to integrate with any data sources.

Apache Flink stands out for its ability to process continuous data streams on a scalable basis. Flink's unique architecture allows it to work with both batch and streaming data, making it a versatile tool for various tasks. Flink's integration with Kafka is highly seamless, ensuring smooth interaction and supporting exactly-once delivery semantics. This means that each event will be processed exactly once, even in case of system failures [16].

TensorFlow, developed by Google, is one of the most popular libraries for machine learning and deep learning. In the context of video streaming, TensorFlow provides flexible and powerful tools for processing and analyzing large volumes of video data. PyTorch, created by Facebook, is also an essential tool for developing machine learning and deep learning models. PyTorch supports dynamic computational graphs, allowing for more intuitive model development and testing. In anomaly detection and forecasting tasks in video streaming, PyTorch is used to create and train models capable of analyzing time series video data and identifying deviations. Its intuitive interface and powerful visualization tools facilitate rapid prototyping and model optimization.

Scikit-learn is a machine-learning library for Python that focuses on simplicity and efficiency. While scikit-learn is not designed for deep learning, it offers a wide range of algorithms and tools for traditional machine learning that can be useful for anomaly detection in video streaming. For instance, clustering algorithms such as K-means and outlier detection methods such as Isolation Forest can be used to analyze video data and detect anomalies. Scikit-learn also provides tools for data preprocessing and model evaluation, making it easier to integrate these components into video streaming systems [17].

Prophet, developed by Facebook, is one of the leading tools for time series forecasting. Prophet is designed with an emphasis on ease of use and high prediction accuracy. It excels particularly with time series that contain seasonal variations and trends, making it ideal for video streaming analysis where such patterns may be present. Prophet uses additive models for forecasting, which can account for various components of the time series, such as seasonality, trends, and holidays. This capability allows for effectively modeling complex time series and accurately predicting future values.

StatsModels is another powerful tool for time series analysis and statistical modeling. StatsModels offers a wide range of capabilities for statistical analysis and model building, including methods for working with time series such as autoregressive integrated moving average (ARIMA), seasonal autoregressive integrated moving average (SARIMA), and exponential smoothing. These models enable detailed analysis of time series and accurate forecasts based on historical data. In the context of video streaming, this means the ability to predict server load, potential failures, and changes in user behavior [18].

For reliable and scalable storage, as well as for conducting advanced data analysis, tools such as Elasticsearch and Kibana are often used. These tools provide powerful capabilities for indexing, searching, and visualizing data, which is especially important when working with large volumes of video data. Elasticsearch is ideal for storing and searching large volumes of data, making it indispensable for video streaming systems that generate vast amounts of logs and metadata.

Kibana provides a user-friendly interface for creating dashboards and visualizing data stored in Elasticsearch. Kibana allows users to quickly create graphs, charts, and maps that help better understand the system's state and identify key trends and anomalies. Kibana's interactive capabilities enable detailed data analysis and sharing of results with other team members [19]. These tools allow for the automation of workflows, efficient resource management, and high performance, ultimately improving user service quality and enhancing the stability of video streaming platforms [20, 21].

Next, we will consider the process of integrating components and data flows in video streaming systems. In modern video streaming architectures, components such as AI/ML models (e.g., TensorFlow, PyTorch, scikit-learn), predictive analytics tools (Prophet, StatsModels), data storage and analysis systems (Elasticsearch, Kibana), as well as orchestration and scaling tools (Apache Airflow, Kubernetes), and monitoring tools (Prometheus, Grafana) play a crucial role. Proper integration of these components allows for the creation of an efficient and scalable video streaming system [22]. Specific examples of creating such a system will be presented next.

1. Customizing Prometheus and Grafana

```
global:
  scrape_interval: 15s
scrape_configs:
  - job_name: 'app_metrics'
    static_configs:
      - targets: ['localhost:8000']
```

Docker Compose for Prometheus and Grafana (docker-compose.yml):

```
version: '3.7'

services:
  prometheus:
    image: prom/prometheus
    volumes:
      - ./prometheus.yml:/etc/prometheus/prometheus.yml
    ports:
      - "9090:9090"

  grafana:
    image: grafana/grafana
    ports:
      - "3000:3000"
    environment:
      - GF_SECURITY_ADMIN_PASSWORD=admin
    volumes:
      - grafana-storage:/var/lib/grafana

volumes:
  grafana-storage:
```

2. Configuring Apache Airflow

Docker Compose for Apache Airflow (docker-compose-airflow.yml):

```
version: '3.7'

services:
  postgres:
    image: postgres:13
    environment:
      POSTGRES_USER: airflow
      POSTGRES_PASSWORD: airflow
      POSTGRES_DB: airflow

  webserver:
    image: apache/airflow:2.1.0
    environment:
      AIRFLOW__CORE__SQL_ALCHEMY_CONN:
postgresql+psycopg2://airflow:airflow@postgres/airflow
      AIRFLOW__CORE__EXECUTOR: LocalExecutor
    volumes:
      - ./dags:/usr/local/airflow/dags
    ports:
      - "8080:8080"
    depends_on:
      - postgres
    command: webserver

  scheduler:
    image: apache/airflow:2.1.0
    environment:
      AIRFLOW__CORE__SQL_ALCHEMY_CONN:
postgresql+psycopg2://airflow:airflow@postgres/airflow
      AIRFLOW__CORE__EXECUTOR: LocalExecutor
    volumes:
      - ./dags:/usr/local/airflow/dags
    depends_on:
      - postgres
    command: scheduler
```

3. Example DAG for Apache Airflow

ETL DAG (dags/etl_dag.py):

```
from airflow import DAG
from airflow.operators.python_operator import PythonOperator
from airflow.utils.dates import days_ago
import pandas as pd

def extract_data(**kwargs):
    # Пример извлечения данных
    data = {'time': ['2024-07-17 10:00:00', '2024-07-17 10:01:00'],
            'cpu_usage': [20, 30]}
    df = pd.DataFrame(data)
    df.to_csv('/tmp/data.csv', index=False)
```

```
def transform_data(**kwargs):
    df = pd.read_csv('/tmp/data.csv')
    df['cpu_usage'] = df['cpu_usage'] * 1.1  # Пример трансформации данных
    df.to_csv('/tmp/data_transformed.csv', index=False)

def load_data(**kwargs):
    df = pd.read_csv('/tmp/data_transformed.csv')
    # Пример загрузки данных в Elasticsearch
    # es = Elasticsearch(['http://localhost:9200'])
    # for _, row in df.iterrows():
    #     es.index(index='metrics', body=row.to_dict())

default_args = {
    'owner': 'airflow',
    'start_date': days_ago(1),
}

with DAG(
    dag_id='etl_dag',
    default_args=default_args,
    schedule_interval='@daily',
) as dag:
    extract = PythonOperator(
        task_id='extract_data',
        python_callable=extract_data,
    )

    transform = PythonOperator(
        task_id='transform_data',
        python_callable=transform_data,
    )

    load = PythonOperator(
        task_id='load_data',
        python_callable=load_data,
    )

    extract >> transform >> load
```

These examples demonstrate the basic integration of components for video streaming, including monitoring with Prometheus and Grafana, and orchestrating ETL processes with Apache Airflow. Working together, they enable efficient data management and high performance of the video streaming system [23].

Having reviewed the examples, it is necessary to move on to the study of implementing AI/ML models for anomaly detection and forecasting. There are numerous methods for anomaly detection, which can be divided into statistical methods, machine learning-based methods, and deep learning-based methods. Statistical methods include time series analysis and distribution analysis, which allow for identifying deviations from normal behavior. Machine learning-based methods include clustering and classification, where models are trained to distinguish between normal and anomalous data. Deep learning-based methods, such as autoencoders and recurrent neural networks (RNNs), enable efficient processing of complex and high-dimensional data [24].

Deployment and scaling of the system. Initially, the deployment process involves installing and configuring software on target servers or in cloud infrastructure. At this stage, it is crucial to ensure the correct operation of all system components, their interaction, and compliance with stated requirements. Various automation tools, such as Ansible, Chef, Puppet, and others, can be used for this purpose. These tools allow standardizing the deployment process, reducing the likelihood of errors, and accelerating the implementation of new software versions.

Scaling the system is aimed at ensuring its stable operation under increasing load. Scaling can be horizontal or vertical. Horizontal scaling involves adding additional servers or containers, while vertical scaling involves increasing the power of existing servers by adding resources such as RAM or CPU cores. Both approaches have their advantages and disadvantages, and the choice between them depends on specific operating conditions and system architecture [25].

Containerization and cloud technologies represent revolutionary approaches in software development, deployment, and operation [26]. Containerization is based on the concept of isolating applications and their dependencies into containers that can run independently of the environment in which they are executed. The main containerization tool is Docker, which allows developers to create, test, and deploy applications in standardized containers. Containers provide ease of transfer and environment reproducibility, significantly simplifying the software development and deployment process.

Cloud technologies provide the infrastructure and platforms for deploying and managing containerized applications. Cloud providers, such as Amazon Web Services (AWS), Microsoft Azure, and Google Cloud Platform (GCP), offer a wide range of services, including computing resources, data storage, and container management tools such as Kubernetes. Kubernetes, in particular, has become the de facto standard for container orchestration, providing mechanisms for automatic deployment, scaling, and management of containerized applications [27].

## CONCLUSION

This research examined the problem of improving data quality in video streaming services using scalable AI/ML-based anomaly detection tools. The focus was on analyzing existing methods, such as supervised and unsupervised learning, as well as deep learning, and identifying their limitations. The proposed solution includes the use of advanced machine learning methods for effectively detecting anomalies in data and predicting user behavior.

The developed anomaly detection system has significant potential for improving data quality in video streaming services. It not only efficiently detects and eliminates anomalies but also predicts user behavior, which enhances user experience and service reliability. This, in turn, will lead to increased user satisfaction and audience retention, which is critically important for the success of streaming platforms.

The practical significance of the proposed solution lies in its ability to significantly reduce data processing costs and network resource management. Implementing AI/ML tools allows automating content moderation processes and managing video stream quality, leading to more efficient resource utilization and reduced operational costs. These advantages make the proposed solution attractive to a wide range of video streaming platforms seeking to improve their service quality and increase competitiveness.

The future development prospects of this technology include integration with new data sources, such as social networks and user reviews, allowing for even more accurate and timely anomaly detection.

## REFERENCES

1. How AI simplifies and automates the work of streaming services. [Electronic resource] Access mode: https://www.comnews.ru/content/230555/2023-12-07/2023-w49/1013/kak-ii-uproschaet-i-avtomatiziruet-rabotu-strimingovykh-servisov (accessed 07/12/2024).
2. Anitha Kumari K, Dharani D. A Survey on Anomaly Detection using Unsupervised Learning Techniques. [Electronic resource] Access mode: https://www.researchgate.net/publication/343318335_A_Survey_on_Anomaly_Detection_using_Unsupervised_Learning_Techniques (accessed 12.07.2024).
3. Popova I.A. Detection of anomalies in a data set using machine learning algorithms without a teacher isolation forest and local outlier factor // StudNet. 2020. No.12. pp.1460-1470.
4. Chernikov A.D. Prediction and recognition of objects in a video stream using deep learning // Bulletin of Science. 2023. No.9 (66). pp.209-215.
5. Video Analysis to Detect Suspicious Activity Based on Deep Learning. [Electronic resource] Access mode: https://dzone.com/articles/video-analysis-to-detect-suspicious-activity-based (accessed 12.07.2024).
6. Kalytyuk I.S., Frantsuzova G.A., Gunko A.V. On the issue of choosing methods for predictive analysis of social media data // Automation and software Engineering. 2019. No.4 (30). pp.9-17.
7. User behavior: how to understand and predict user behavior using click modeling. [Electronic resource] Access mode: https://fastercapital.com/ru/content/Поведение-пользователей--как-понять-и-спрогнозировать-поведение-пользователей-с-помощью-моделирования-кликов.html (accessed 07/12/2024).
8. Denis Khryashchev,Alexandru Papiu, Jiamin Xuan, Olivia Dinica, Kyle Hubert, Huy Vo Who Watches What: Forecasting Viewership for the Top 100 TV Networks // Computational Data and Social Networks. 2019. pp.163-174.
9. Lyndon Nixon Predicting Your Future Audience's Popular Topics to Optimize TV Content Marketing Success // Conference: Proceedings of the 2nd International Workshop on AI for Smart TV Content Production, Access and Delivery. 2020. p.5.
10. Sizikov D.O. Time series analysis method and its mathematical model in software // International Scientific Research Journal. 2024. No.3 (141) . [Electronic resource] Access mode: https://research-journal.org/archive/3-141-2024-march/10.23670/IRJ.2024.141.5 (accessed 07/12/2024).
11. Python | ARIMA Model for Time Series Forecasting. [Electronic resource] Access mode: https://translated.turbopages.org/proxy_u/en-ru.ru.8f7a0cad-66981a60-3df9103b-74722d776562/https/www.geeksforgeeks.org/python-arima-model-for-time-series-forecasting / (accessed 12.07.2024).
12. Dozhdikov Anton Valentinovich Forecasting the results of film distribution using machine learning // VTE. 2023. No.4. pp. 93-114.
13. Hybrid Machine Learning Approach to Popularity Prediction of Newly Released Contents for Online Video Streaming Service. [Electronic resource] Access mode: https://translated.turbopages.org/proxy_u/en-ru.ru.859ed0ea-66981acf-db75a817-74722d776562/https/paperswithcode.com/paper/hybrid-machine-learning-approach-to (accessed 12.07.2024).
14. Morozova V. I. Forecasting by machine learning method // Young Scientist. 2022. No. 21 (416). pp. 202-204.
15. Kusakina D. S. Development and research of a method for detecting anomalies in a video stream based on the analysis of the context of a

scene // Young Scientist. 2023. No. 19 (466). pp. 15-18.

**16.** Building architectures for real-time data processing using Apache Kafka, Flink and Druid. [Electronic resource] Access mode: https://habr.com/ru/companies/timeweb/articles/783758 / (accessed 12.07.2024).

**17.** Anomaly detection with TensorFlow. [Electronic resource] Access mode: https://translated.turbopages.org/proxy_u/en-ru.ru.41c286df-66981b41-8a53e5ad-74722d776562/https/www.geeksforgeeks.org/anomaly-detection-with-tensorflow / (accessed 12.07.2024).

**18.** Orlova I.V. Using the Prophet package in forecasting time series // Fundamental Research. 2021. No. 3. pp. 94-102.

**19.** ELK Stack Tutorial: Get Started with Elasticsearch, Logstash, Kibana, & Beats. [Electronic resource] Access mode: https://translated.turbopages.org/proxy_u/en-ru.ru.2192893c-66981b77-65adc37a-74722d776562/https/www.tutorialspoint.com/elk-stack-tutorial-get-started-with-elasticsearch-logstash-kibana-amp-beats (accessed 12.07.2024).

**20.** How to use the Kubernetes Executor in Airflow in production. [Electronic resource] Access mode: https://www.restack.io/docs/airflow-kubernetes-executor (accessed 12.07.2024).

**21.** Monitoring of WebRTC streams using Prometheus and Grafana. [Electronic resource] Access mode: https://habr.com/ru/companies/flashphoner/articles/568784 / (accessed 07/12/2024).

**22.** Prometheus vs Grafana: Top Differences. [Electronic resource] Access mode: https://translated.turbopages.org/proxy_u/en-ru.ru.973f9ca9-66981ba6-c413aef2-74722d776562/https/www.geeksforgeeks.org/prometheus-vs-grafana / (accessed 12.07.2024).

**23.** Integration of third-party video analytics. [Electronic resource] Access mode: https://flussonic.ru/doc/integrate-external-video-analytics / (accessed 12.07.2024).

**24.** Machine Learning for Anomaly Detection. [Electronic resource] Access mode: https://translated.turbopages.org/proxy_u/en-ru.ru.7c9cebae-66981bc9-0aab5e33-74722d776562/https/www.geeksforgeeks.org/machine-learning-for-anomaly-detection / (accessed 12.07.2024).

**25.** Kubernetes for DevOps. [Electronic resource] Access mode: https://github.com/rustamgarifulin/books/blob/master/files/Kubernetes_%D0%B4%D0%BB%D1%8F_DevOps_%D1%80%D0%B0%D0%B7%D0%B2%D0%B5%D1%80%D1%82%D1%8B%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5%2C_%D0%B7%D0%B0%D0%BF%D1%83%D1%81%D0%BA_%D0%B8_%D0%BC%D0%B0%D1%81%D1%88%D1%82%D0%B0%D0%B1%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5_%D0%B2.pdf (accessed 12.07.2024).

**26.** Immersion in containerization. [Electronic resource] Access mode: https://habr.com/ru/companies/otus/articles/767884 / (accessed 12.07.2024).

**27.** CI/CD for Machine Learning: What it is & Benefits in 2024. [Electronic resource] Access mode: https://research.aimultiple.com/ci-cd-machine-learning / (accessed 12.07.2024).