АННОТАЦИЯ

Выпускная квалификационная работа содержит 74 страницы текста, включая 43 риснука. Работа посвящена разработке системы прогнозирования и обнаружения неисправностей на станках лазерной резки Навигатор КС-12В. Особенностью созданной системы является использованием последних достижений в сфере интеллектуального анализа данных, а также уникальность.

Ключевые слова – интеллектуальный анализ данных, нейронные сети, кластеризация, временные ряды, анализ временных рядов, промышленный интернет вещей.

ANNOTATION

Graduation paper contains 74 pages of text, including 43 pictures. The work is devoted to the development of a system for predicting and detecting faults on machines laser cutting Navigator KS-12V. A feature of the created system is the use of the latest achievements. in the field of data mining, as well as uniqueness.

Keywords – data mining, neural networks, clustering, time series, time series analysis; industrial internet of things.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Основная цель данной дипломной работы – разработка модуля для прогнозирования и обнаружения ошибок и сбоев на станках лазерной резки Навигатор КС-12В компании ВНИТЭП. Проблема прогнозирования неисправностей актуальна не только для станков данной модели, но и для других разнообразных видов производственных устройств, так как не существует общепринятых стандартов для задач подобного рода, а сами задачи решаются экспериментально на основе последний достижений науки.

Мотивацией для создания системы прогнозирования и выявления неисправностей является сохранение ресурсов компании Сеспель, которая использует станки Навигатор, а именно сокращение времени и денежных средств по диагностике и устранению неисправностей.

Для определения источников и типов неисправностей в дипломной работе было дано функциональное и модульное описание станка Навигатор КС-12В. На основе этого описания определялись состояния ошибок и сбоев, а также на основе рекомендаций инженеров компании ВНИТЭП.

Решение поставленной проблемы потребовало определения лучших методов и подходов. Для этого в дипломной работе были исследованы и проанализированы существующие решения, направленные на предсказания и выявление неисправностей устройств. Все решения, описанные в работе, строятся на основе принципов интеллектуального анализа данных и анализа временных рядов. Эти две области также были описаны в работе.

В процессе анализа предметной области были выявлены задачи, которые необходимо было решить для достижения поставленной цели. Все задачи были перечислены.

В процессе анализа были выявлены достоинства и недостатки существующих решений, а также сделан выбор инструментов и методов на основе

анализа, а именно набор библиотек для работы с данными с использованием языка Python и методы для прогнозирования и обнаружения сбоев.

Для прогнозирования временных рядов данных со станка была использована нейронная сеть LSTM. В работе был дан анализ архитектуры данной нейронной сети, а также описаны дополнительные методы для оптимизации результатов работы этой нейроной сети.

Для выявления ошибок и сбоев на станке используются предсказанные значения, а также общие тренды параметров лазера: температуры и мощности. Основные критерии неисправностей – выход за крайние значения температуры и мощности лазера в течение продолжительного времени.

В целях выявления паттернов для состояний станка, был использован алгоритм k-Shape, который направлен на кластеризацию многомерных временных рядов. Данный алгоритм является лучшим для кластеризации временных рядов. В работе описаны принципы алгоритма в сравнении с другими алгоритмами. Также, стоит отметить, что выявленные паттерны используются алгоритмом градиентного бустинга, который также описан в работе.

Разработанный модуль является частью программного комплекса компании Omnicube. В работе было дано описание этой системы, а также принципы интеграции созданного модуля в систему.

1 АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ

1.1 Задача обнаружения и прогнозирования ошибок и сбоев на станке лазерной резки Навигатор

Основным объектом исследования данной работы является станок лазерной резки Навигатор компании ВНИТЭП. Данный станок установлен в компании Сеспель, занимающейся производством автоцистерн и полуприцепов для различных видов перевозок: бензовозы, зерновозы, самосвальные прицепы, цементовозы и т.д.

Опишем функциональный и модульный состав данного станка, а также обозначим проблему обнаружения и прогнозирования ошибок и сбоев для станка. Кроме этого, опишем подход для решения данной задачи через автоматизацию.

1.1.1 Описание комплекса Навигатор КС-12В

Рисунок 1 – Внешний вид комплекса Навигатор КС-12В

Навигатор КС-12В (рисунок 1) представляет собой комплекс обработки листового металла с волоконным лазером, линейным синхронным двигателем и числовым программным управлением (ЧПУ). Разработкой и

поддержкой данного комплекса занимается Российская компания ВНИТЭП. Станок является основным продуктом данной компании.

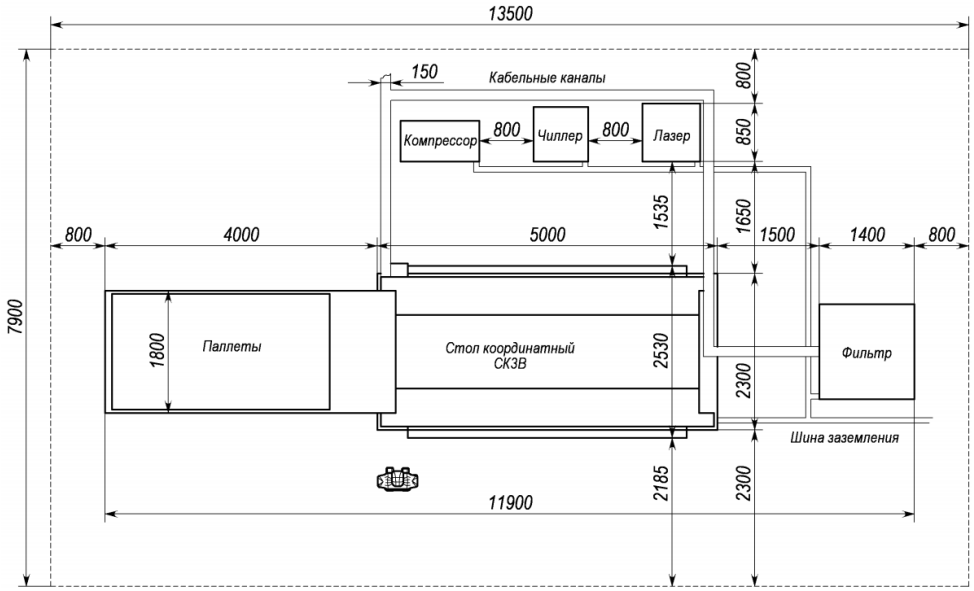


Рисунок 2 – Графическое описание основных частей станка

Комплекс состоит из следующих основные модулей (рисунок 2): координатный стол, иттербиевый лазер, компрессор, установку для фильтрации и вентиляции, чиллер. Координатный стол состоит из пульта ЧПУ с программным обеспечением, линейных моторов, линейных шариковых направляющих, кабельных каналов, предохранительных каналов, паллет для сбора технологических отходов, оптической измерительной системы.

Возможности станка покрывают задачи лазерной резки большинства металлов различной толщины, до 30 мм. Основные используемые металлы: бронированная сталь, нержавеющая сталь, цирконий, латунь.

Для работы со станком оператор загружает программу через интерфейс ЧПУ. Загружаемая программа представляет собой набор инструкций для станка, которые выполняет станок для вырезки определенной детали определенного формата.

Объектом дипломной работы является модель описанного станка. В компании Сеспель используются несколько станков данной модели.

Задача дипломной работы будет решаться в контексте системы компании Omnicube.

Компания Omnicube разрабатывает и предоставляет универсальную платформу для решения разнообразных задач интеллектуального мониторинга на предприятиях различных отраслей: промышленность, интеллектуальные сервисы для зданий, мониторинг персонала, экологический мониторинг, сервисы для предприятий сельского хозяйства, инструменты для медицинских учреждений. [45]

Одним из клиентов компании Omnicube является предприятие Сеспель. Компания Omnicube предоставляет функционал сбора, обработки и отображения данных устройств предприятия: станки, установки, производственные роботы. Сами данные собираются с интегрированных и установленных датчиков этих устройств. Кроме этого, Omnicube предоставляет пользовательский интерфейс для руководства и сотрудников компании Сеспель. На данном интерфейсе отображена статистика по различным параметрам, а также отображены различные статистические показатели.

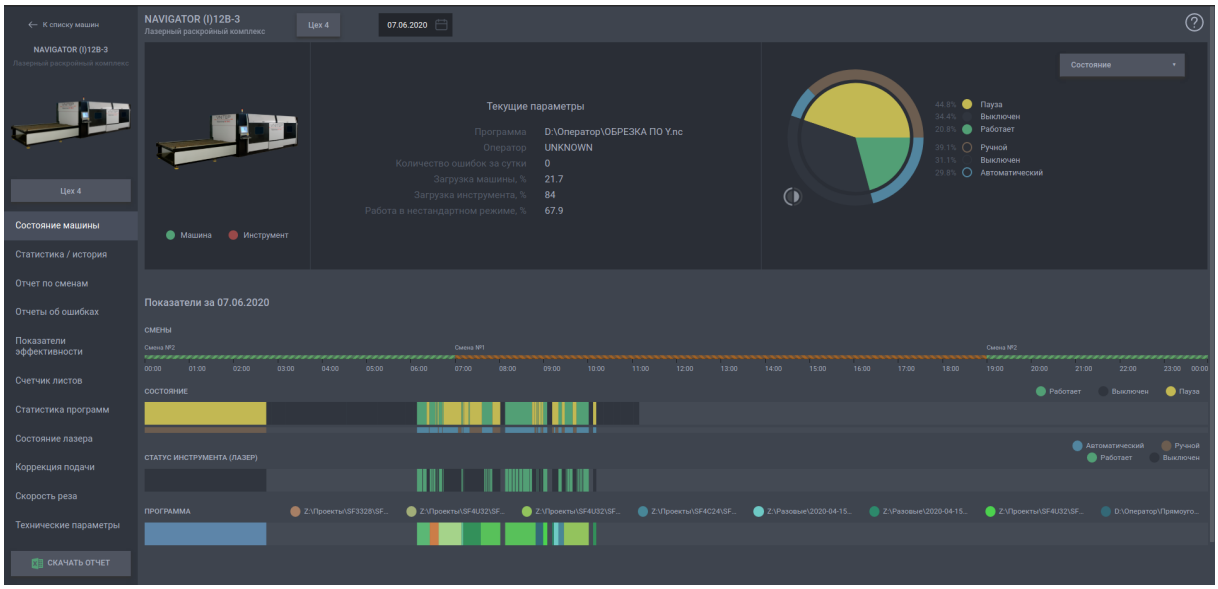


Рисунок 3 – Главные показатели работы станка

Главными показателями (рисунок 3), которые отображаются в интерфейсе являются: состояние, статус лазера, работающая программа. Все по-

казатели записываются относительно рабочий смен сотрудников. Параметр состояния отображает текущий режим работы станка и представляет собой три режима: рабочий (работа), выключенный (выключен), пауза. Показатель статус инструмента (лазера) отображает статистику по работе лазера станка и может иметь режимы: автоматический, ручной, рабочий, выключенный. Рабочий режим представляет собой использование лазера под присмотром оператора. Кроме этого, основным показателем также является статистика используемых программ для станка.

Вся статистика ведется ежедневно, так как работа ведется посменно. Данную статистику можно проверить в разделе статистика-история. Кроме этого, собирается статистика в виде отчета по сменном каждого оператора.

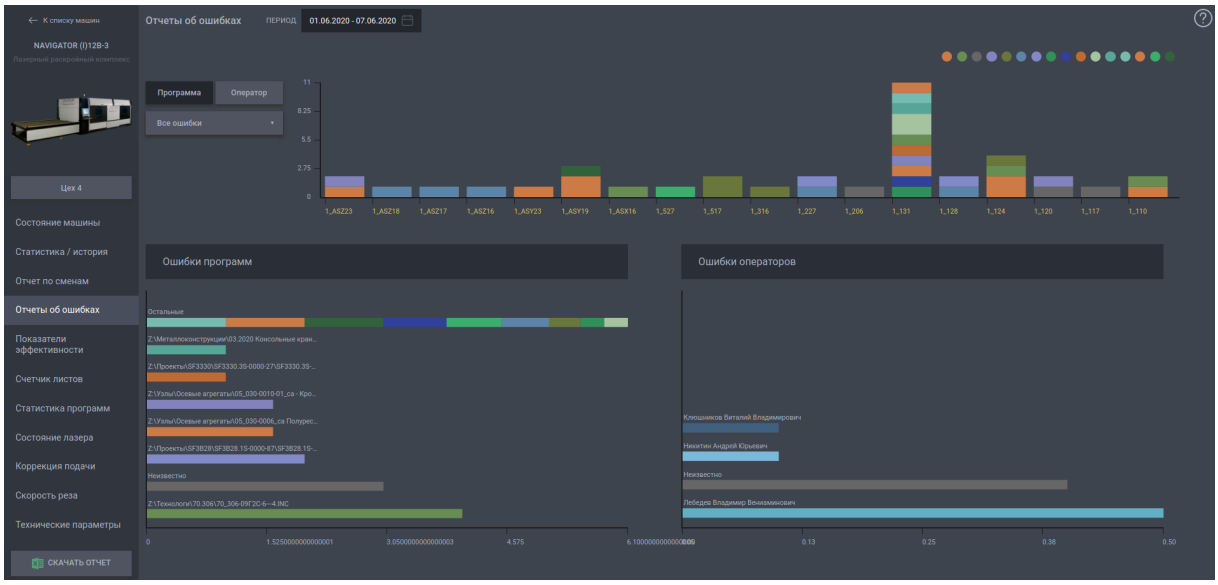


Рисунок 4 – Отчет об ошибках программ и операторов

В разделе «Отчеты об ошибках» (рисунок 4) отображается статистика по неисправностям работающих программ и по ошибкам операторов. В основном графике можно увидеть количество возникновения ошибки (по оси Y), имеющей определенный код (ось X)

Раздел «Показатели эффективности» (рисунок 5) содержит показатели загрузки машины, коэффициента эффективности и коэффициента использования инструмента. Параметр «Загрузка машины» представляет собой вре-

Рисунок 5 – Показатели эффективности

мя полезной работы. «Коэффициент эффективности» – отношение времени полезной работы к длительности включения. «Коэффициент использования инструмента» – отношение времени работы инструмента ко времени полезной работы машины. Все показатели выражены в процентах, либо в часах. Также возможно проверить показатели как по сменам, так и по часам.

Рисунок 6 – Статистика счетчика листов

Раздел «Счетчик листов» (рисунок 6) отображает статистику по количеству загружаемых в станок металлических листов. Для просмотра статистики можно выбрать желаемый период.

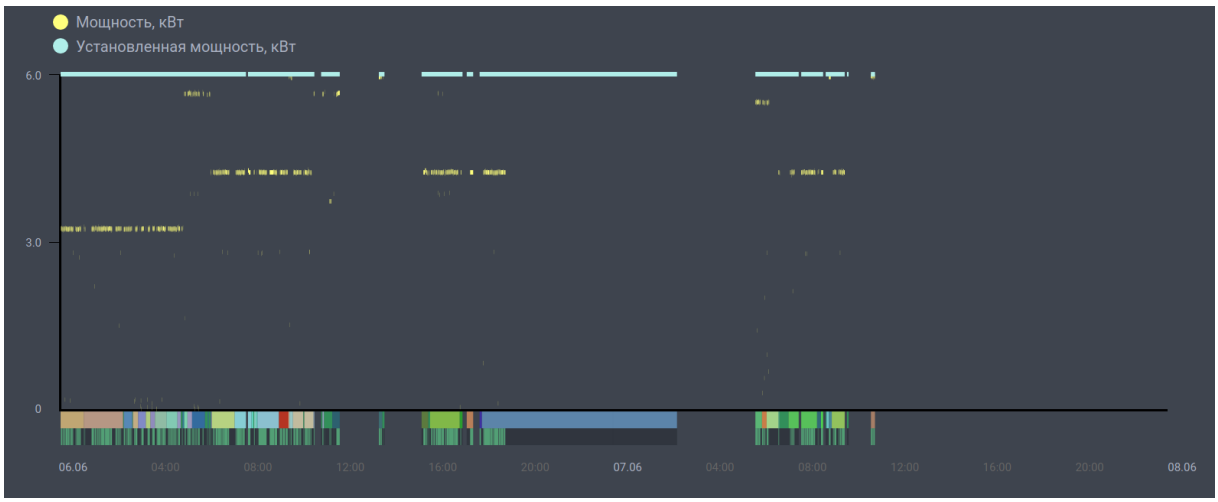


Рисунок 7 – Статистика по используемой мощности лазера

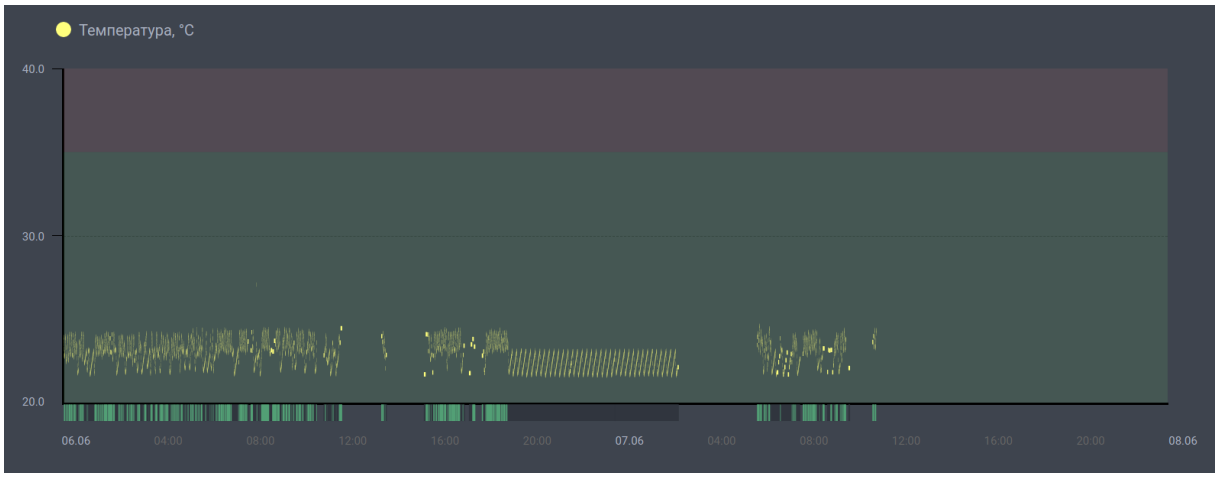


Рисунок 8 – Статистика по температуре источника

Раздел «Состояние лазера» содержит три показателя по состоянию лазера в течение использования: отношение фактической мощности лазера к заданной, мощность фактическая и установленная (рисунок 7), температура источника лазера (рисунок 8).

Кроме этого есть еще разделы: «Статистика программ» – статистика по используемым программам; «Коррекция подачи» – отклонение по программам и операторам; «Скорость реза» содержит данные по скорости

реза, коррекции скорости реза, а также данную статистику относительно программ и операторов. Также можно ознакомиться с основными техническими характеристиками данного станка, а также загрузить отчет в формате .xls, в котором будет содержаться вся статистика по всем показателям за обозначенный промежуток.

1.1.2 Проблема диагностики неисправностей

Эффективная эксплуатация промышленных устройств требует наличия надежных обслуживающих систем, которые должны предоставлять безопасные, отказоустойчивые решения, поэтому предотвращение ошибок и сбоев имеет первостепенное значение.

Обычно, чтобы предотвратить сбои и ошибки, операторы, как правило, выполняют техническое обслуживание на основе рекомендаций производителей, и им приходится регулярно проверять критически важные детали, Для этого требуются опытные специалисты с высоким уровнем квалификации. Данный процесс очень дорог и трудоемок.

Данная проблема присутствует и на станках модели Навигатор. Регулярно возникают ситуации ошибок программного обеспечения и работы операторов, а также возможны критические сбои, которые заключаются в отказе оптической системы (лазера). Также возможен износ оборудования, например, сбой привода оси XYZ.

На данный момент операторам необходимо постоянно следить за состоянием станка, проводить сложный технический анализ, который может потребовать дополнительных специалистов. Как уже говорилось, такой подход может занять много времени, а также потребовать немало денежных ресурсов. Такой подход является единственным для станков модели Навигатор.

1.1.3 Автоматическое обнаружение неисправностей как решение проблемы диагностики неисправностей

Целью данной дипломной работы является создание системы для обнаружения и прогнозирования неисправностей на станках лазерной резки Навигатор. Решение данной задачи представляет собой разработку и внедрение данной системы, а также описание теоретического обоснования.

Решение поставленной задачи принесет положительные результаты как для компании ВНИТЭП, так и для ее клиентов, в том числе для предприятия Сеспель, а именно такое решение может сократить время операторов и инженеров, тем самым буду сэкономлены денежные и другие ресурсы предприятия. Разрабатываемое решение повысить эффективность промышленных процессов предприятия Сеспель.

Основными средствами для решения поставленной задачи будет собранная за период 2017-2019 статистика по различным параметрам различных модулей: ЧПУ, лазер, составные части.

Основными метриками будут:

- Отношение заданной мощности лазера к фактической, которое позволит отслеживать тренд износа головы лазера;
- Температура лазера – параметр, который может также быть признаком износа лазерной головы, а также может быть предупреждающим параметром о возможном отклонении от нормы необходимой температуры, что может классифицироваться как ошибка оператора или ошибка станка в целом;
- Данные с датчиков вибрации – позволяют отследить тренд изнашивания платформы станка;
- Данные с датчиков привода XYZ – позволяют определить ошибки и сбои пространственного приводы лазера.

1.1.4 Описание подхода к решению задачи автоматической диагностики

В последнее время получила свое распространение парадигма промышленного интернета вещей, являющаяся частью более общего понятия – интернета вещей, которая предоставляет идеи объединения промышленного оборудования и датчиков в единую систему. Такая система может позволить проводить автоматизированный мониторинг и анализ критически важных параметров промышленных устройств, без участия операторов, для обнаружения и предсказания ошибок и сбоев [25].

Не смотря на то, что парадигма промышленного интернета вещей предоставляет идеи для создания системы, она не дает описания способов и методов реализации этой системы. Такое положение обусловлено относительно недавним появлением данной парадигмы, из-за чего еще не устоялась методология и принципы разработки.

Так как на данный момент нет устоявшихся концепций, ведутся разработки на основе различных подходов, при этом учитывается специфика промышленных устройств, для которых разрабатывается система.

Проблему разработки системы промышленного интернета вещей можно разбить на две подпроблемы, каждую из которых можно решать отдельно, однако решение второй подпроблемы может зависить от решения первой:

1. Разработка подсистемы сбора, обработки и хранения данных с различных устройств;
2. Разработка подсистемы анализа данных для обнаружения и предсказания ошибок и сбоев.

Решение первой подпроблемы может стать основной для решения второй подпроблемы, так как для анализ данных требует наличие готовой среды, в которой будет происходить анализ.

Если первая подпроблема может быть решена проектированием грамотной архитектуры и выбора подходящих инструментов, то для решения второй задачи необходимо определиться с подходами и методами получения прогнозов ошибок и сбоев, с учетом особенностей данных.

Можно сказать, что для того, чтобы иметь систему обнаружения и прогнозирования ошибок и сбоев устройств, нужно решить две поставленные подпроблемы. Первая подпроблема не является темой данной дипломной работы и выходит за ее рамки, однако, в работе будет описана с учетом преимуществ и недостатков существующая разработанная система, в контексте которой будет решаться вторая подпроблема.

1.1.5 Временные ряды

Проблему предсказания и обнаружения ошибок и сбоев на станках можно охарактеризовать как задачу анализа временных рядов, так как основные данные представляют собой последовательность событий, разграниченных по времени.

Временные ряды – серия точек данных, индексированных во временном порядке. Анализ временных рядов представляет собой совокупность методов анализа данных временных рядов, которые направлены на выявление значимых паттернов [7] .

Временные ряды имеют темпоральную структуру, то есть упорядочены во времени, тем самым анализ временных рядов отличается от перекрестных исследований, в которых обычно используется анализ выборки или выборок из генеральной совокупности в определенный момент времени. Кроме этого, анализ временных рядов отличается от пространственного анализа, который имеет такую же фиксированную основу как и перекрестные исследования, например, в пространственном анализе могут анализироваться географические объекты, без учета временной компоненты [14] .

Существует классификация временных рядов [11] :

– одномерные и многомерные;
– стационарные и нестационарные;
– линейные и нелинейные.

Пусть есть процесс x_t, где $t \geq 0$ или $-\infty < t < \infty$.

Одномерные представляют собой временной ряд, которые генерируется на основе одного атрибута, а многомерный имеет больше одного, другими словами в одномерном случае x_t представляет собой вектор, а в многомерном x_t определяется как матрица.

Данные временного ряда можно представить в виде $n \times d$ матрицы данных [8]:

$$\mathbf{D} = \begin{pmatrix} x_{1,1} & x_{1,2} & \cdots & x_{1,d} \\ x_{2,1} & x_{2,2} & \cdots & x_{2,d} \\ \vdots & \vdots & \ddots & \vdots \\ x_{n,1} & x_{n,2} & \cdots & x_{n,d} \end{pmatrix} \tag{1}$$

Вектор-строки вида:

$$\mathbf{x}_i = (x_{i,1}, ..., x_{i,d}) \tag{2}$$

в зависимсоти от предметной области имеют различные названия, например: сущности, объекты, примеры и т.д.

Вектор-столбцы вида:

$$\mathbf{X}_j = (x_{1,j}, ..., x_{n,j}) \tag{3}$$

также могут иметь различные названия: атрибуты, свойства, переменные и т.д.

Стационарность определяется через требование равносильности совместных распределений вектора $(x_{t_1}, ..., x_{t_k})$ и вектора $(x_{t_1+t}, ..., x_{t_k+t})$. Дру-

гими словами, в стационарном временном ряде есть повторяющиеся паттерны, в отличие от нестационарного [6] .

Линейные модели определяются через свойства линейности параметров временного ряда, например среднего и дисперсии. Нелинейные модели напротив могут иметь нелинейную структуру, что требует использования нелинейных методов, например, методов нелинейной регрессии [5] .

Самыми малоизученными являются нелинейные нестационарные модели [15] . Примеры реализации системы обнаружения и предсказания ошибок, основанные на данном типе моделей не были найдены и обсуждаться в данной работе не будут, так как данная область выходит за рамки темы дипломной работы.

Обозначенную выше подзадачу можно разбить также на два элемента:

1. выявление ошибок, обнаружение аномалий;
2. п редсказание поведения устройства.

Данные задачи можно решать как отдельно, где для каждой задачи будет использоваться свой метод, так и совместно через один метод. Каждый вариант будет рассмотрен в существующих примерах.

1.1.6 Интеллектуальный анализ данных временных рядов

Также можно сказать, что поставленная задача обнаружения и предсказания ошибок и сбоев является задачей интеллектуального анализа данных, а именно интеллектуального анализа данных временных рядов. Такое абстрагирование обосновано, так как существуют аналогичные задачи, которые решаются именно с использованием данной парадигмы.

Интеллектуальный анализ данных (data mining) – процесс выявления паттернов из больших объемов данных [19] .

Интеллектуальный анализ данных представляет собой междисциплинарную область, в которой используются методы, идеи и принципы из следу-

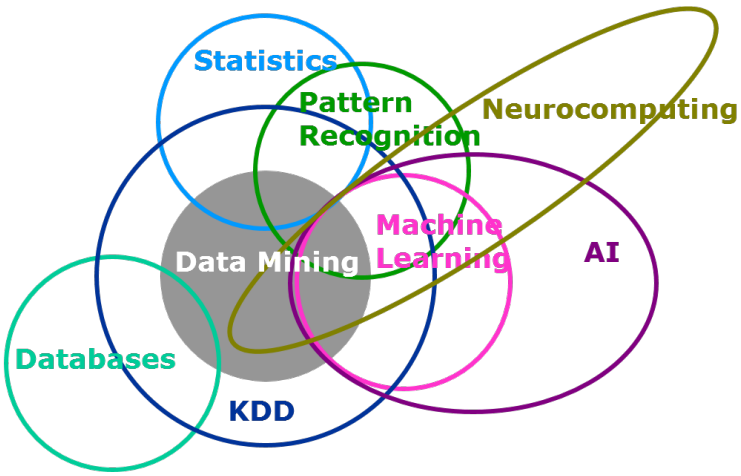


Рисунок 9 – Место интеллектуального анализа данных среди других областей

ющих областей (рисунок 9): статистика, машинное обучение, распознавание образов, вычислительная нейробиология, базы данных. Стоит отметить, что данная область является частью более широкой области, которая называется выявление (обнаружение) знаний из баз данных (knowledge database discovery).

Кроме этого, интеллектуальный анализ данных можно определить как часть распознавания образов/паттернов (pattern recognition), которая связана с обработкой данных из баз данных и выявлением паттернов, связанных с определенной предметной областью [16] . Сама область распознавания образов покрывает и другие области, например, обработка сигналов и машинное зрение [17] .

Опишем приведенные области, обозначим их основные идеи и принципы.

База данных – организованная коллекция данных, которая собирается и хранится при помощи компьютерных систем. Для взаимодействия пользователя с базой данных используется система управления базами данных (СУБД).

Теория баз данных поставляет формализованные методы и принципы проектирования и разработки баз данных и СУБД. Обычно теоретически

выделяют два главных типа построения баз данных: реляционные (SQL) и нереляционные базы данных (NoSQL). Также иногда выделяют смешанный тип, появившейся сравнительно недавно: новые реляционные базы данных, который совмещает лучшие идеи и принципы из реляционных и нереляционных теорий и подходов (NewSQL) [3] .

В контексте интеллектуального анализа данных базы данных используются как источник данных. Сами данные перед анализом обрабатываются, что также является частью процесса обнаружения знаний. От типа и особенностей сбора и хранения данных в базе зависят последующие шаги анализа данных.

Обнаружение знаний из баз данных (knowledge database discovery, KDD) – более широкая область работы с данными из баз данных, частью которой является интеллектуальный анализ данных. KDD определяют как процесс выявлений знаний (паттернов), который разбит на основные шаги [44] :

1. селекция – выборка данных из баз данных по определенным критериям;
2. обработка (pre-processing) – приведение выбранных данных в подходящий вид (очистка и удаление пропущенных значений, фиксирование атрибутов) для последующего анализа;
3. трансформация (transformation) – трансформация данных в подходящую структуру данных, например, в матрицу данных (1);
4. интеллектуальный анализ данных (data mining) – обнаружение паттернов в данных;
5. интерпретация и оценка (interpretation and evaluation).

Следующим шагом также может быть использование, развертывание и эксплуатация обнаруженных знаний и паттернов для целей предметной области.

Статистика, распознавание образов и машинное обучение являются поставщиками методов для каждого шага процесса KDD. Каждая из этих областей имеет свои особенности, однако все они имеют также много общего.

1.2 Анализ существующих решений задачи обнаружения и прогнозирования неисправностей в устройствах

Итак, рассмотрим существующие решения для обнаружения аномалий, например, ошибок и сбоев в данных, а также подходы к прогнозированию и оповещению об этом пользователей.

Во-первых, можно сказать, что модели прогнозирования временных рядов, такие как авторегрессивная модель (AR), модель скользящего среднего (MA), экспоненциальная модель, модель авторегрессии скользящего среднего (ARMA), интегрированная авторегрессионная модель скользящего среднего (ARIMA), зависят от параметров, полученных из исторических временных рядов. Эти параметры являются неотрицательными целыми числами, которые относятся к порядку авторегрессионной части, степени участвующей первой разности и порядку части скользящей средней. Хотя эти процессы могут обрабатывать нестационарные данные, они ограничены в запоминании любого состояния за любой промежуток времени [13].

В [26] авторы рассмотрели потоковые данные как частично наблюдаемый Марковский процесс [27]. Такое решение было принято с целью создания композиционной системы, которая состояла из различных моделей: экспоненциальные модели, сезонные модели, модель лог-гауссовского процесса Кокса. Данная статья содержит хорошие идеи использования фильтров на композиции частично наблюдаемых Марковских процессов, а также теоретического обоснования такого использования. Однако, хорошая разработанная теория и слабое тестирование на реальных данных не показало высокой эффективности такого подхода. В данной статье авторы научились только

предсказывать аномалии, но не смогли их определить, а также использовать данный подход в реальном времени.

Федеративное обучение – один из это методов машинного обучения, который состоит в обучении алгоритма на нескольких децентрализованных периферийных устройствах или серверах, содержащих локальные выборки данных, без обмена их выборками данных. Использование федеративного обучения для фильтрации данных сети медицинских устройств было рассмотрено в [28]. Благодаря использованию методов федеративного обучения, авторам удалось повысить энергоэффективность обработки и анализа данных, то есть они решили одну из самых распространенных проблем в сфере интернета вещей. Также в статье описаны этапы локального анализа данных на самих устройствах и глобального анализа данных на сервере. Для первого этапа авторы использовали адаптивные фильтры [29], для второго матричный анализ возмущений [30]. Недостатком такого подхода является низкая гибкость, так как может потребоваться использовать корреляционный анализ устройств, а в распределенном анализе корреляция может быть утрачена, тем самым будут утрачены полезные признаки. Кроме этого, существуют усовершенствованные адаптивные фильтры, например ядерный рекурсивный фильтр [31]. Еще можно сказать, что матричный анализ возмущений может не учитывать асинхронную структуру предобработанных данных.

В целях мониторинга профиля и обнаружения неисправностей в [32] авторы применили модель нелинейной параметрической регрессии для разработки системы, которая была бы устойчивой и нечувствительной к изменениям температуры в производственной практике. В [33] был разработан метод мониторинга, который может автоматически адаптировать параметры контрольной карты (метод из теории управления). В [34] авторы добавили все профили каналов и применили анализ главных компонентов (PCA) агрегированным профилям тоннажа для извлечения характеристик. В [35] использовались как статистические, так и вейвлет-характеристики, извлечен-

ные из сигналов датчиков, для разработки адаптивного метода обучения для оценки износа инструмента в процессе высокоскоростного фрезерования. Каждое из этих исследований было сосредоточено на анализе индивидуального профиля данных, то есть анализ был проведен с одномерными данными. Однако, выходы датчиков устройства обычно являются многоканальными, что требует многомерного анализа временных рядов.

У многомерных данных могут быть следующие особенности [10]:

- С увеличением числа датчиков обработка каждого временного ряда от каждого датчика становится как более интенсивной, что сказывается на повышении вычислительной нагрузки;
- Данные имеют больше выбросов и шумов, по сравнению с одномерными данными;
- Временные ряды от разных датчиков имеют корреляции на разных уровнях, что приводит к большому количеству избыточной информации, однако необходимо учитывать возможные корреляции;
- Состояния работы до простоя обычно продолжаются в течение некоторого времени и включают состояния отказа оборудования. Кроме того, после восстановления неисправного датчика также появляется период задержки в рабочем состоянии. Следовательно, модель прогнозирования должна иметь инвариантность памяти относительно таких состояний.

Эти особенности данных приводят к следующим вопросам исследования:

- Как уменьшить избыточную информацию без ухудшения точности прогнозирования?
- Как определить выбросы и шумы?
- Как моделировать долгую память состояний?
- Как точно предсказать текущее состояние на основе исторических данных?

Опишем существующие результаты, которые отвечают на эти вопросы.

Авторы в статье [36] применили многофакторную PCA для уменьшения размерности данных с целью повышения эффективности системы анализа профиля и отслеживания данных многоканального профиля с помощью контрольных диаграмм. Автор статьи [37] извлек набор нескольких функций мониторинга из данных профиля большого размера, чтобы обнаружить износ инструмента. В другой статье [38] автор применил некоррелированный многолинейный анализ главных компонентов (UMPCA) [39] для мониторинга профиля и диагностики неисправностей, который учитывал взаимосвязь каналов различных профилей. По сравнению с UMPCA, основанной на PCA, линейный дискриминантный анализ (LDA) также является классическим методом выделения признаков в данных. Однако обычный LDA не может работать с многомерными данными напрямую, потому что это метод предназначен для одномерных задач.

В статье [40] был предложен некоррелированный мультилинейный дискриминантный анализ для распознавания лиц и обработки изображений. Такой анализ напрямую работает с многомерными данными и извлекает некоррелированные дискриминантные признаки посредством проекции тензор вектор. По сравнению с алгоритмом UMPCA, который не контролируется, такой метод представляет собой контролируемый мультилинейный экстрактор признаков, который будет учитывать информацию о классе при извлечении объектов и может быть более подходящим для распознавания лиц. Несмотря на то, что есть некоторые предварительные исследования по применению UMLDA для распознавания лиц и обработки изображений, в литературе опубликовано мало исследований по использованию технологии UMLDA для анализа многомерных данных для автоматического управления процессом в промышленных системах. В [41] автора предложили решение проблемы обнаружения неисправностей в реальном времени на основе объединения многомерных данных временных рядов в модели UMLDA,

однако авторы не смогли до конца достичь высокой эффективности, которые предоставляют другие методы.

1.3 Постановка решаемых задач

Как уже было обозначено выше, основной целью дипломной работы является создание системы для обнаружения и прогнозирования неисправностей на станках лазерной резки Навигатор КС-12В компании ВНИТЭП, установленных на предприятии Сеспель.

Основной гипотезой является достижение цели через методы интеллектуального анализа данных многомерных временных рядов.

Окончательно определим задачи, которые необходимо решить:

1. подготовка и обработка накопленных данных со станка Навигатор;
2. первичный описательный анализ данных;
3. выбор и обоснование выбора модели для предсказания многомерных временных рядов данных станка;
4. выбор оптимального окна обучения;
5. определение адаптивного окна предсказания;
6. выбор и обоснование выбора модели кластеризации в данных;
7. анализ полученных кластеров;
8. выбор обучающейся на данных модели для обнаружения неисправностей;
9. реализация моделей и тестирование моделей;
10. интеграция моделей в систему компании Omnicube.

1.4 Выводы

В данном разделе была обозначена проблема выявления ошибок и сбоев на станках лазерной резки модели Навигатор КС-12В, а также в устройствах вообще. Были описаны особенности данной проблемы, а также ее теоретические аспекты. Кроме этого, был проведен анализ существующих решений, базирующихся на различных методах, подходах и идеях. Для каждого случая была дана критика, а именно обозначены достоинства и недостатки каждого решения.

В конце раздела были обозначены задачи, решения которых может помочь достичь решения проблемы выявления ошибок и прогнозирования.

2 ОПИСАНИЕ МЕТОДОВ И ИНСТРУМЕНТОВ ДЛЯ РЕАЛИЗАЦИИ

2.1 Система для промышленного интернета вещей

Как было описано в предыдущим разделе 1, промышленный анализ данных предоставляет идеи для создания интеллектуальной системы для автоматизации и повышения эффективности работы промышленных устройств. Одной из множества компаний, которая ведет деятельность в данной области, является компания Omnicube, которая, как уже говорилось, имеет свою универсальную систему для множества задач не только промышленного интернета вещей, но интернета вещей в общем, то есть задачи, решаемые данной системы, не ограничиваются промышленной отраслью. Реализация решения для обнаружения и прогнозирования неисправностей решается в контексте данной системы. Опишем данную систему.

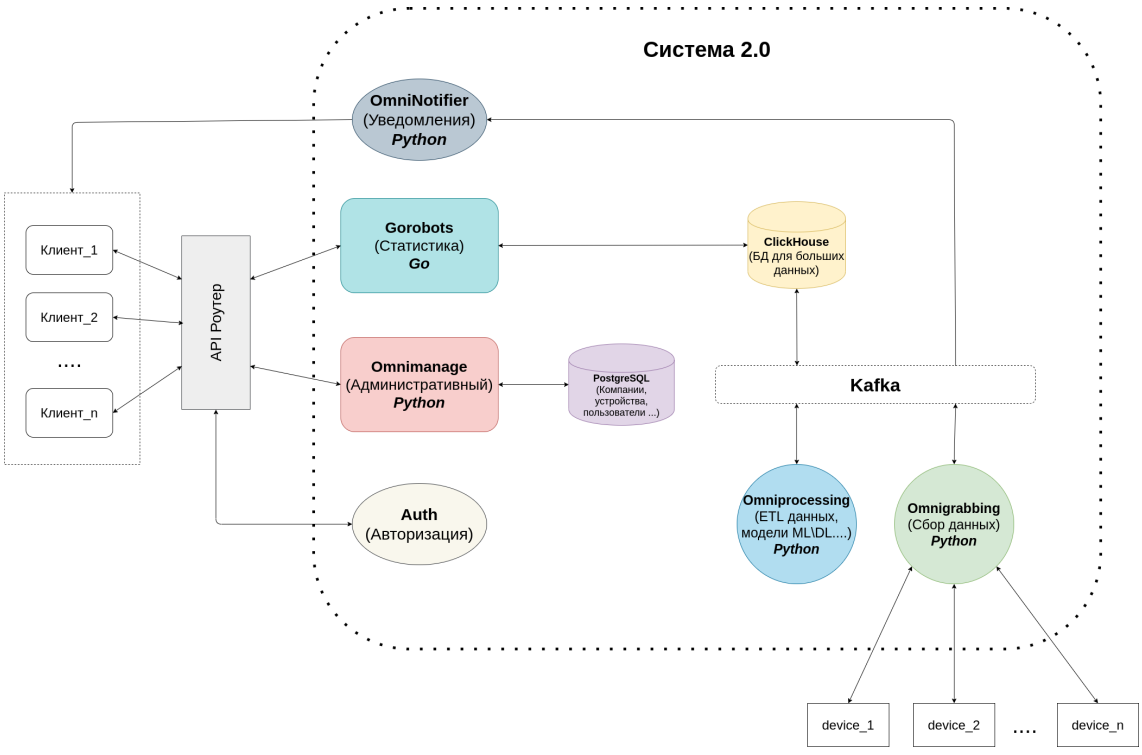


Рисунок 10 – Разработанная система для интернета вещей

Система имеет микросервисную архитектуру (рисунок 10). Микросервисная архитектура представляет собой рассмотрение разрабатываемого приложения как совокупности связанных сервисов, каждый их которых отвечает за какую-то часть общей системы, а также решает свою определенную задачу. Такой подход имеет свои преимущества и недостатки Основным преимуществом является модульность, которая позволяет разрабатывать и поддерживать компоненты независимо друг от друга. Недостатками являются: высокая сложность разработки, сложный процесс развертывания системы на собственных кластерах, и, особенно, на кластерах клиентов.

Основными взаимодействующими с системой структурами являются клиенты и устройства клиентов. Каждое клиентское устройство соединяется с системой при помощи инженеров компании Omnicube с учетом специфики самих устройств. Например, для станка лазерной резки Навигатор использовались датчики, которые устанавливались на критически важные узлы и с которых собиралась необходимая информация. Помимо этого, сами устройства клиентов иногда предоставляют функционал, который позволяет собирать данные без установки датчиков. Далее, данные устройств агрегируются и обрабатываются также с учетом специфики устройств. Обработанные данные выдаются клиентам в виде пользовательского интерфейса, примером которого является интерфейс для предприятия Сеспель с данными станка лазерной резки Навигатор.

Главными узлами системы являются два микросервиса: omnimanage и gorobots. Микросервис omnimanage отвечает за обработку первичных данных клиента: сведения о клиенте, сведения о пользователях компании клиента, сведения об устройствах клиента (но не сами данные устройств), сведения о ролях пользователей для авторизации и т.д. Gorobots отвечает за обработку собранной с устройств статистики: вычисляет основные статистические показатели (среднее, дисперсию, КПД). Omnimanage написан на языке Python

с использованием базы данных PostgreSQL. Gorobots написан на языке Go, данные в этом микросервисе берутся из базы данных ClickHouse.

Микросервис omnigrabbing отвечает за сбор и первоначальную обработку данных клиентских устройств. в данном микросервисе происходит конвертация в необходимых формат для отправления данных в Kafka, и последующую загрузку в ClickHouse.

Kafka – платформа для обработки потоковых данных в реальном времени. Проект направлен на создание унифицированной высокопроизводительной платформы с низкой задержкой для обработки потоков данных в реальном времени. Kafka использует двоичный протокол на основе TCP, а также системы для обработки асинхронных данных. Такой брокер был выбран по причине большого количества поступающих данных. Kafka позволяет гибко решать задачи обработки большого количества асинхронных данных [46].

ClickHouse - это быстрая система управления базами данных OLAP с открытым исходным кодом. Данная система ориентирована на столбчатую структуру, а также позволяет генерировать аналитические отчеты с использованием SQL-запросов в режиме реального времени. База данных была выбрана по причине высокой нагрузки, заключающейся в необходимости хранить большое количество данных. Данная СУБД позволяет обрабатывать несколько тысяч запросов в секунду, что является ключевой особенностью, которая выделяет данную систему среди других конкурентов [47].

В системе компании Omnicube также присутствуют вспомогательные микросервисы:

– API роутер – система для перенаправления запросов клиентов;
– Auth – сервис для авторизации пользователей;
– OmniNotifier – система для оповещения клиентов.

Микросервис omniprocessing (рисунок 11) является местом, где располагаются различные модели для мониторинга и обработки данных, напри-

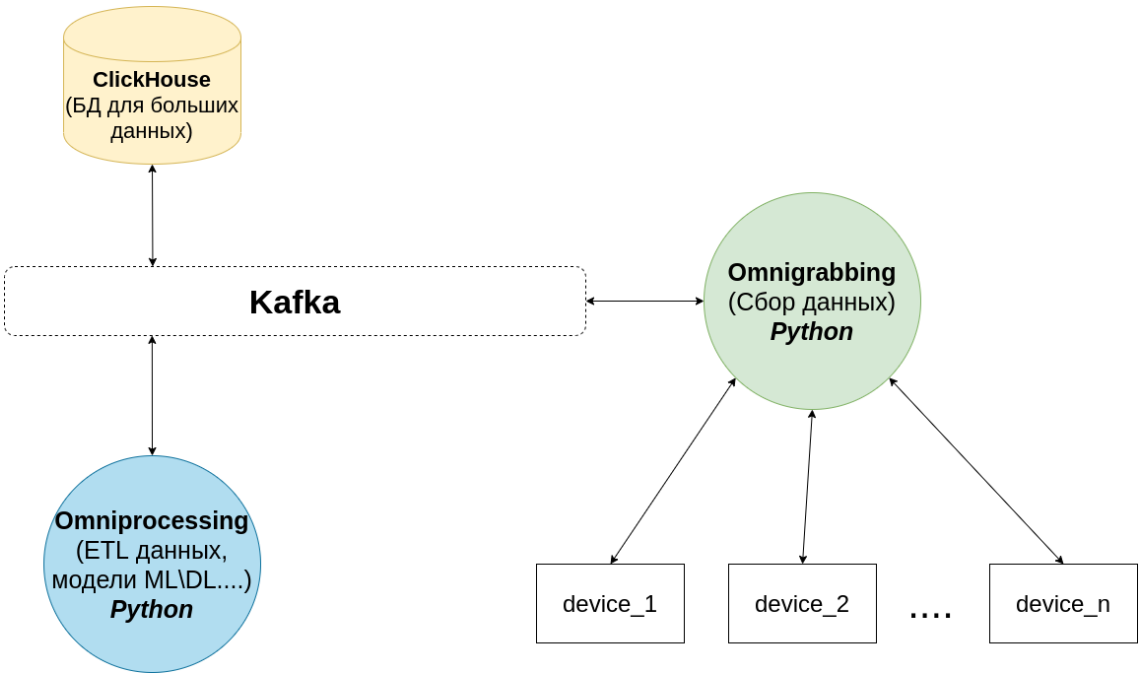


Рисунок 11 – Часть системы, отвечающая за анализ и обработку данных

мер, модели машинного обучения. Основные цели данного микросервиса – извлечение, трансформация и загрузка.

В omniprocessing используется Python фреймфорк Faust, который позволяет отправлять данные в очередь брокеру сообщений Kafka. Из Kafka данные попадают в базу данных ClickHouse. Благодаря данному фреймворку можно быстро строить и встраивать пайплайны машинного обучения [48]. Разрабатываемое решение для станков Навигатор располагается именно в микросервисе omniprocessing.

2.2 LSTM нейронная сеть для прогнозирования временных рядов

В последнее время использование нейронных сетей помогло улучшить результаты решения большого количества задач в различных сферах человеческой деятельности. Это обусловлено повышенными вычислительными мощностями современных компьютеров. Именно повышение мощности позволило использовать теоретические изыскания многих поколений ученых, занимающихся нейронными сетями, максимальным образом.

Область прогнозирования временных рядов не стала исключением для улучшения результатов на основе использования нейронных сетей. Наибольшего прорыва достигли LSTM или long-short term memory сети (долгосрочная краткосрочная память), которые представляют собой разновидность архитектуры рекуррентной нейронной сети.

Рекуррентные нейронные сети, в отличие от однонаправленных многослойных перцептронов, способны сохранять в памяти различные признаки, которые необходимы для поставленной задачи. В случае временных рядов это могут быть долговременные зависимости, которые могут влиять на прогноз.

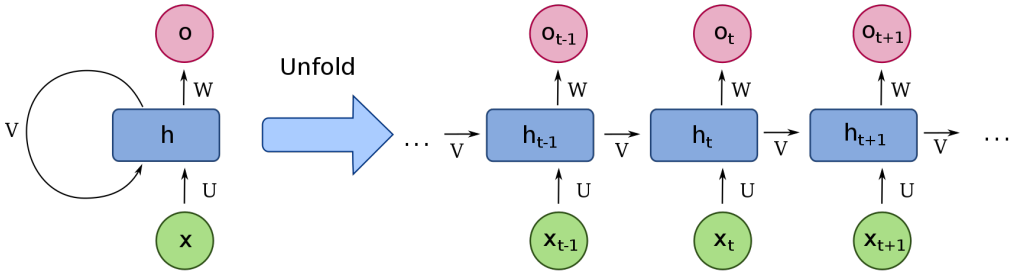


Рисунок 12 – Архитектура рекуррентной нейронной сети

На рисунке 12 отображена архитектура рекуррентной нейронной сети, которая представляет собой направленную последовательность узлов нейронной сети, каждый из которых зависит и влияет на другие узлы на основе перераспределения весов.

Пусть имеется матрица $X = \{\mathbf{x}_1, ..., \mathbf{x}_n\}$ входных данных в нейронную сеть h, которая меняет свое состояние от h_1 до h_n, с учетом последовательной обработки данных X с различными временными шагами t, а также на основе вычислений предыдущих шагов:

$$h_t = f(U\mathbf{x}_t + V\mathbf{x}_{t-1}) \tag{4}$$

Функция 4 представляет собой функцию активации, например, сигмоидальную функцию σ или гиперболический тангенс tanh, либо совокупность функций активации, которая зависит от конкретной архитектуры сети. На каждом шаге t нейронная сеть выдает результат o_t. Матрицы U и V содержат входные и рекуррентные веса соответственно.

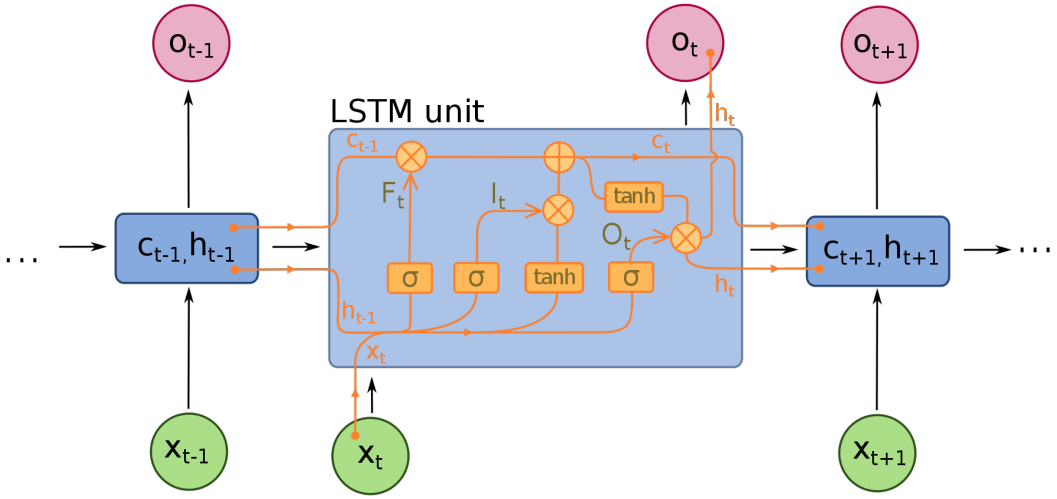


Рисунок 13 – Архитектура LSTM

Рекуррентные нейронные сети имеют различные архитектуры, каждая из которых имеет свои особенности, достоинства и недостатки. Самой эффективной для решения широкого спектра задач является LSTM архитектура (рисунок 13).

$$
\begin{aligned}
F_t &= \sigma(W_F x_t + U_F h_{t-1} + b_F) \\
I_t &= \sigma(W_I x_t + U_I h_{t-1} + b_I) \\
\widetilde{c}_t &= \tanh(W_c x_t + U_c h_{t-1} + b_c) \\
O_t &= \sigma(W_O x_t + U_O h_{t-1} + b_O) \\
c_t &= F_t \circ c_{t-1} + I_t \circ \widetilde{c}_t \quad h_t = o_t \circ \tanh(c_t)
\end{aligned}
\tag{5}
$$

Система уравнений 5 описывает последовательность вычислений нового скрытого состояния нейронной сети h_t.

– F_t – вектор активации забывающего узла;
– I_t – вектор активации входного обновляемого узла;
– $\widetilde{c}_t$ – вектор активации входящей клетки;
– O_t – вектор активации выходного узла;
– c_t – состояние клетки в момент t;
– W и U – матрицы весов;
– b – вектор смещения.

2.3 Оптимальное окно обучения

Обычно, точность модели прогнозирования увеличивается пропорционально количеству обучающих данных, однако такой подход не всегда является подходящим. При изменении поведения или статистики поступающих данных, обученная модель может не отследить эти изменения, что приведет к накоплению ошибок. Такую проблему называют дрейфом концептов. Для решения данной проблемы используется движущееся окно для повторного обучения, которое формируется по последним данным на основе заданного размера окна.

Выбор оптимального размера окна обучения моделей машинного обучения является открытым вопросом. Большой размер окна может иметь более точные результаты, но это увеличивает сложность модели, что делает ее непригодной для приложений реального времени, тогда как небольшой размер окна может привести к увеличению погрешности и, следовательно, к снижению надежности системы, переобучению или недообучнению.

Авторы статьи [42] предложили метод, на основе которого определяется оптимальный размер обучающего окна. Основная идея заключается в использовании спектрального анализа наименьших квадратов, так же известного как метод Ломба-Скаргл. Данный метод работает лучше на данных с пропусками, например, такого похожего метода как быстрое преобразование

Фурье. Кроме этого, быстрое преобразование Фурье требует равномерной распределенности данных, которая не всегда возможна при анализе данных с датчиков устройств.

$$P_X(f) = \frac{1}{2\sigma^2}\left(\frac{\left[\sum_{n=1}^{N}(x(t_n)-\overline{x})\cos(2\pi f(t_n-\tau))\right]^2}{\sum_{n=1}^{N}\cos^2(2\pi f(t_n-\tau))} + \frac{\left[\sum_{n=1}^{N}(x(t_n)-\overline{x})\sin(2\pi f(t_n-\tau))\right]^2}{\sum_{n=1}^{N}\sin^2(2\pi f(t_n-\tau))}\right) \tag{6}$$

Функция 6 является описанием метода Ломба-Скаргла. Оптимальный размер обучающего окна определяется на основе нахождения наибольшей периодической компоненты, встречающейся в данных. Другими словами, определяется размер сезонности данных. Однако, перед использованием данного метода данные необходимо протестировать на сезонность для того, чтобы убедиться корректности использования данного метода.

$$\tan(4\pi f\tau) = \frac{\sum_{n=1}^{N}\sin(4\pi f t_n)}{\sum_{n=1}^{N}\cos(4\pi f t_n)} \tag{7}$$

Параметр τ определяется в уравнении 7.

2.4 Алгоритм k-Shape

Для обнаружения аномалий или неисправностей в данных со станка необходимо использовать основные принципы кластерного анализа, который представляет собой разбиение данных на кластеры с целью выявления закономерностей.

Обычно задача кластеризации формулируется следующим образом. Обозначим за $X = \{\mathbf{x}_1, ..., \mathbf{x}_n\}$ набор n данных, где $\mathbf{x}_i \in \mathbb{R}^m$. При кластеризации данные X необходимо разбить на k непересекающихся кластеров $P = \{p_1, ..., p_k\}$ таким образом, что сумма квадратов дистанций между экземпляром данных x_i и центроидов $\mathbf{c}_j$, который представляет собой точку концентрации кластера p_j, минимальна:

$$P^* = \arg\min \sum_{j}^{k} \sum_{\mathbf{x}_i \in p_j} dist(\mathbf{x}_i, \mathbf{c}_j) \tag{8}$$

В Евклидовом пространстве подобная задача оптимизации является NP-полной. Метод k-средних позволяет найти локальный оптимум посредством случайного распределения k центроидов среди данных.

В кластерном анализе существуют различные алгоритмы кластеризации, каждый из которых имеет свою специфику и может зависеть от структуры анализируемых данных.

В последние несколько десятилетий, кластеризация последовательностей временных рядов получила значительное внимание, не только как мощный самостоятельный исследовательский метод, но и также как один из множества шагов в исследовании и обработки данных.

Большинство методов анализа временных рядов, включая кластеризацию, зависят от выбора меры расстояния. Ключевой вопрос при сравнении двух последовательностей временных рядов заключается в том, как обрабатывать различные искажения, которые характерны для временных последовательностей.

Из-за различных трудностей и потребностей в инвариантности относительно предметных областей, большое внимание уделялось созданию новых мер расстояния, а не созданию новых алгоритмов кластеризации. Обычно считается, что выбор меры расстояния более важен, чем сам алгоритм кластеризации. Как следствие, кластеризация по временным рядам основывается главным образом на классических методах кластеризации: либо путем замены расстояния по умолчанию на более подходящее для временных рядов, либо путем преобразования временных рядов в «плоские» данные для того, чтобы существующие алгоритмы кластеризации могли быть использованы непосредственно.

Однако выбор метода кластеризации может повлиять на точность, так как каждый метод выражает однородность и разделение кластеров поразному, и эффективность, поскольку вычислительные затраты отличаются от одного метода к другому. Например, спектральная кластеризация или некоторые варианты иерархической кластеризации являются более подходящими для идентификации кластеров на основе плотности распределения (на основе областей с более высокой плотностью), чем методы разделения, такие как метод k-средних. С другой стороны, метод k-средних более эффективен, чем иерархические или спектральные методы в общем случае.

Современные подходы к кластеризации на основе формы данных, в которых используются методы разбиения с мерами расстояния, не зависящими от масштаба и сдвига, имеют два основных недостатка: (i) эти подходы не могут масштабироваться до больших объемы данных, поскольку они зависят от вычислительно дорогостоящих методов или мер расстояния; (ii) эти подходы были разработаны для конкретных предметных областей или их эффективность была показана только для ограниченного числа наборов данных. Более того, наиболее успешные методы кластеризации на основе форм обрабатывают фазовую инвариантность посредством локального нелинейного выравнивания координат последовательности, даже если глобальное выравнивание часто является адекватным.

Описанные подходы никогда не подвергались всесторонней оценке друг против друга, против других методов разделения или против различных подходов, таких как иерархические или спектральные методы. Данный сравнительный анализ присутствует только частично в различных статьях.

В статье [43] авторы предложили новый алгоритм k-Shape для кластеризации временных рядов на основе форм данных, который эффективен и не зависит от конкретной предметной области. k-Shape основан на масштабируемой процедуре итеративного уточнения, аналогичной той, которая используется алгоритмом k-средних, но с существенными отличиями. В

частности, k-Shape использует и другую меру расстояния, и другой метод вычисления центроидов, чем метод k-средних. Алгоритм k-Shape пытается сохранить формы последовательностей временных рядов при их сравнении. Для этого k-Shape требуется мера расстояния, которая инвариантна к масштабированию и сдвигу. В отличие от других подходов кластеризации, для k-Shape адаптируется статистическая мера кросс-корреляции.

Кросс-корреляция - это мера сходства сигналов с задержкой по времени, которая широко используется для обработки сигналов и изображений. Данная статистическая мера позволяет определить сходство двух последовательностей $\mathbf{x} = (x_1, ..., x_m)$ и $\mathbf{y} = (y_1, ..., x_m)$ даже если они не выровнены относительно друг друга (и даже если они имеют разную длину). Инвариантность относительно сдвигов достигается благодаря фиксированию одной последовательности, например, **y**, и последовательному скольжению **x** через **y**, при котором вычисляется скалярное произведение этих последовательностей.

В статье [43] авторы алгоритма k-Shape описали разбили этот алгоритм на три части: k-shape, extract-shape и sdb. Опишем каждую часть.

На сначала вход алгоритму подается набор z-нормализованных временных рядов, а также k кластеров, количество которых определяется заранее, что является отдельной задачей.

$$z_i = \frac{x_i - \mu}{\sigma} \tag{9}$$

Уравнение 9 описывает z-нормализацию, представляющую собой отношение отклонения случайного элемента x_i от математическое ожидания μ к стандартному отклонению σ.

```
def k_shape(x, k):
    iter = 0, idx = []
    centroids = [0,...,0]
    for i in range(100):
        old_idx = idx
        for j in range(k):
            centroids[j] = extract_shape(
                idx, x, j, centroids[j])
        distances = (1 - sbd(centroids[j], x))
        if old_idx == idx:
            break
```

Рисунок 14 – Алгоритм k-Shape

На рисунке 14 отображен псевдокод алгоритма k-Shape. На первом шаге инициализируются необходимые переменные: счетчик итераций (iter), список для заполнения значений на основе кластеров (idx), список заполненных нулями центроидов (centroids), длина которого зависит от количества кластеров k. Далее происходят итерации, которые определяют центроиды и сами кластеры соответственно. Авторы посчитали, что 100 итераций будет достаточно для нахождения локального оптимума. Функция extract_shape предназначена для вычисления нового центроида. Функция sbd предназначена для вычисления так называемого shape-distance нормализации коэффициентов кросс-корреляции.

```
def extract_shape(idx, x, j, cur_center):
    a = []
    for i in range(len(idx)):
        x[i] = sbd(cur_center, x[i])
    S = x[i]*x[i]^T
    Q = I - 1/m * O
    M = Q^T * S * Q
    C' = Eig(M, 1)
    return C`
```

Рисунок 15 – Алгоритм вычисления нового центроида на основе расстояния SBD

На рисунке 15 отображен псевдокод вычисления нового центроида при помощи максимизации квадратов SBD расстояний (уравнения 10 и 11). Матрицы I и O представляют собой единичную матрицу и матрицу, заполненную единицами. В уравнении 10 функция $CC(\mathbf{x}, \mathbf{y})$ означает кросс-корреляцию, R_0 – функция автоматической кросс-корреляции.

$$SBD(\mathbf{x}, \mathbf{y}) = 1 - \max\left(\frac{CC(\mathbf{x}, \mathbf{y})}{\sqrt{R_0(\mathbf{x}, \mathbf{x}) \cdot R_0(\mathbf{y}, \mathbf{y})}}\right) \tag{10}$$

$$\mathbf{y}_k^* = \arg\max \sum_{\mathbf{x}_i \in P_k} \Big(SBD(\mathbf{x}_i, \mathbf{y}_k)\Big)^2 \tag{11}$$

На основе свойств кросс-корреляции авторы алгоритма переписали исходное уравнение 11 в виде максимизации отношения Рэлея:

$$\mathbf{y}_k^* = \arg\max \frac{\mathbf{y}_t^T \cdot M \cdot \mathbf{y}_t}{\mathbf{y}_t^T \cdot \mathbf{y}_t} \tag{12}$$

В уравнении 12 матрица M является эрмитовой и имеет разложение Шура в виде $M = Q^T \cdot S \cdot Q$. Вычисление матриц Q и S отображено в листинге

2. Максимальное значение $\mathbf{y}_k^*$ авторы нашли в качестве собственного вектора, который соответствует максимальному собственному значению симметричной матрицы M.

```
def sbd(x, y):
    length = 2^2(2*length(x)-1)
    CC = IFFT{FFT(x, length) * FFT(y, length)}
    NCC c = CC/(||x|| ||y|| )
    [value, index] = max(NCC)
    dist = 1 - value
    shift = index - length(x)
    if shift   0 then
        y_0 = [zeros(1 , shift), y(1 : end - shift)]
    else
        y_0 = [y(1 - shift : end), zeros(1 , -shift)]
```

Рисунок 16 – Эффективное вычисление SBD расстояния

На рисунке 16 отображен алгоритм вычисления SBD расстояния с использованием быстрого и обратного быстрого преобразования Фурье (FFT и IFFT). Также, в данном листинге описаны последующие манипуляции с полученной после работы с быстрым преобразованием Фурье кросс-корреляционной мерой.

2.5 Градиентный бустинг

Для выявления неисправностей необходима модель, обученная на основе заданных критериев неисправностей и выявленных кластеров временных рядов. Так как критерии и кластеры представляют собой выявленные признаки на этапе анализа каждого временного ряда по отдельности, необходимо использовать модель, способную распознавать признаки у всех временных рядов одновременного, то есть, контексте данной работы, определять

неисправности на основе данных из различных датчиков в виде многомерных временных рядов.

Для такой задачи лучше всего подходит подход на основе бустинга. Бустинг – это специальный алгоритм, который основан на композиции других моделей машинного обучения. Существуют различные модификации бустинга, каждый из которых имеет свои особенности, достоинства и недостатки. Однако, в последнее время, наилучшие результаты показывает алгоритм градиентного бустинга.

Градиентный бустинг – разновидость бустинга, в основе которого лежит оптимизация дифференцируемой функции потерь с использованием алгоритма градиентного спуска. Опишем более формально данный алгоритм.

Пусть имеется набор данных $\{(\mathbf{x}_i, y_i)\}_{i=1}^{n}$. Необходимо найти аппроксимацию $\hat{F}$ функции $F(x)$, которая минимизирует математическое ожидание функции потерь $L(y, F(x))$ (уравнение 13).

$$\hat{F} = \arg\min \mathbb{E}[L(y, F(x))] \tag{13}$$

Градиентный бустинг принимает y и ищет аппроксимацию $\hat{F}$ на основе взвешенных функций $h_i(x)$ из некоторого класса H, называемых слабыми моделями обучения (уравнение 14).

$$\hat{F} = \sum_{i=1}^{M} \omega_i h_i(x) + const \tag{14}$$

На первом шаге вычисляется функция F_0, которая представляет себе первичную минимизацию функции потерь.

$$F_0(x) = \arg\min \sum_{i=1}^{n} L(y_i, \omega) \tag{15}$$

В итоге задается рекуррентное соотношение метода градиентного спуска для функций потерь (уравнение 16) и весов (уравнение 17).

$$F_m(x) = F_{m-1}(x) - \omega_m \sum \nabla_{F_{m-1}} L(y_i, F_{m-1}(x_i)) \tag{16}$$

$$\omega_m = \arg\min \sum L(y_i, F_{m-1}(x_i) - \omega \nabla_{F_{m-1}} L(y_i, F_{m-1}(x_i)) \tag{17}$$

За счет оптимизации параметров множества слабых моделей, градиентный бустинг способен выдавать результаты высокой точности.

2.6 Инструменты для разработки моделей

Для людей, занимающимися анализом данных и машинным обучением, существует множество инструментов для решения различных их задач. Инструменты могут представлять собой как полноценную платформу (Matlab), так и набор дополнительных программных пакетов (Python, R).

Разработка системы обнаружения и прогнозирования неисправностей на станках лазерной резки будет вестись с использованием языка программирования Python. Данный язык выбран по двум причинам. Во-первых, микросервис, в который будет внедрятся разрабатываемое решение, как уже было описано, разработан на языке Python. Во-вторых, для языка Python созданы гибкие и качественные инструменты для манипуляции и анализа данных, а также для инструменты для использования последних достижений машинного обучения.

Основными инструментами для манипуляции с данными являются библиотеки Numpy и Pandas. Именно они используются для разработки.

Библиотека Numpy представляет собой набор функций для операций над многомерными массивами. Numpy хоть и используется для языка Python, данная библиотека почти полностью написана на языке C, поэтому она имеет высокие показатели производительности, в отличие от стандартных инструментов, которые предоставляет язык Python.

Pandas – библиотека, позволяющая быстро и гибко создавать комплексные датасеты, а также предоставляющая эффективные методы манипуляции с данными различного типа: числовые, категориальные, временные, бинарные и т.д.

Для нужд описательной статистики, а также для визуализации данных используется библиотека Matplotlib. Данный пакет содержит набор функций, который позволяет строить двумерные и трехмерные графики различной сложности. Широкий функционал Matplotlib позволяет наглядно анализировать данные, а также отображать обработанные результаты.

Для нужд статистики и анализа временных рядов существует множество сторонних библиотек для языка Python. Наиболее развитыми являются: SciPy, Statsmodels, Tslearn. SciPy – библиотека для сложных научных вычислений. Statsmodels содержит набор большинства используемых статистических моделей. Tslearn – библиотека для анализа временных рядов.

Многие задачи машинного обучения можно решить с использованием библиотеки машинного обучения Scikit-learn. Данная библиотека предоставляет функционал для решения разнообразных задач машинного обучения: регрессии, классификации, кластеризации, сокращения размерности и т.д. Кроме этого, данная библиотека имеет возможности подготовки данных для тренировки и тестирования моделей машинного обучения. Данная библиотека будет использоваться именно для таких задач.

Библиотека Scikit-learn покрывает большое количество задач машинного обучения, однако она не имеет подходящих инструментов для такой области как глубокое обучение. Также очень часто из-за отсутствия оптимизации некоторых инструментов, библиотека Scikit-learn может не подойти для использования в работающих системах, поэтому ее в основном используют для прототипирования. Поэтому были созданы библиотеки, в которых решены данные проблемы. Нейронные сети, в том числе и используемые в работе LSTM, можно использовать с помощью библиотек TensorFlow и Keras.

TensorFlow – библиотека для задач машинного обучения и глубокого обучения в частности, написанная на языке Python и C++, с дополнительными возможностями использования технологии CUDA – технологии для параллельных вычислений на графических картах компании NVidia. TensorFlow позволяет наглядно и быстро натренировать сложные модели машинного обучения, а также быстро интегрировать и развернуть модели в рабочих средах. Также данная библиотека может использоваться не только в языке Python, но и в языках JavaScipt, C++ и Swift.

Keras представляет собой фреймворк для задач глубокого обучения, включающий в себя инструменты для работы со сложными моделями глубокого обучения – нейронным сетями различной архитектуры. Keras создан на основе TensorFlow, что является преимуществом, так как фреймворк и библиотека способны работать согласованно. Фреймворк Keras будет использоваться для тренировки нейронной сети LSTM.

2.7 Выводы

В данном разделе был проведен анализ системы компании Omnicube, в которую будет интегрировано решение для прогнозирования и обнаружения ошибок станков Навигатор.

Для задач прогнозирования была выбрана рекуррентная нейронная сеть LSTM. В разделе были описаны преимущества перед другими архитектурами нейронных сетей для задач прогнозирования временных рядов. Было дано подробное описание работы сети LSTM. Также был описан принцип, по которому определяется оптимальное окно обучения.

Кроме этого, был проведен анализ алгоритмов кластеризации, а также выбран и описан алгоритм k-Shape для кластеризации многомерных временных рядов. Именно данный алгоритм поможет качественно определять

особенности данных временных рядов станка Навигатор, и на основе этих особенностей определять неисправности.

В конце раздела было дано краткое описание используемых для разработки инструментов, а именно анализ библиотек и фреймворков для задач анализа данных и задач машинного обучения, предназначенных для языка Python.

3 РАЗРАБОТКА И ВНЕДРЕНИЕ В ЭКСПЛУАТАЦИЮ

3.1 Предобработка и первичный анализ данных

Используемые данные были собраны из базы данных ClickHouse. Период собранных данных: от 2 февраля по 11 июня 2020 года. Данные представляют собой набор около миллиона строк в JSON формате, которые имеют следующие признаки:

- состояние (state) – категориальный признак, отображающий состояние станка;
- оператор станка (operator) – ФИО оператора станка, выполняющего работы в момент t;
- название программы (program) – название запущенной программы;
- режим работы лазера (mode), например, автоматический или ручной;
- ампераж (amperage);
- вольтаж (voltage);
- счетчик металлических листов (sheet counter);
- температура лазера;
- мощность лазера;
- установленная мощность лазера;
- время в Unix формате.

Основными анализируемыми параметрами являются: время, мощность и температура лазера, установленная мощность лазера. На момент сбора данных не было возможности собирать данные по вольтажу и амперажу, поэтому значения в этих параметров нулевые. Также не было возможности взятия статистики по ошибкам программ, так как все ошибки записывались в связи с запущенными программами. Данные проблемы будут решены в последующих разработках.

3.1.1 Проверка на нормальность

Первым этапом анализа была проверка распределений, генерируемых временные ряды, на близость к нормальному. Для этого использовался K-квадрат тест д'Агостина, который позволяет определить меру сходимости исходого распределения к нормальному. Гипотеза H_0 заключалась в том, что заданное распределение не является нормальным. Для этого использовалась оценка p-значения.

```
In [5]: from scipy import stats

def ntest(data):
    """Проверка распределения на нормальность."""
    # stat = k^2 + s^2 -- z-scored эксцесс и ассиметрия
    stat, p = stats.normaltest(data)
    svalues = f'\nstat={stat}, p-value={p}\n'
    alpha = 0.05
    if p > alpha:
        return 'Данные похожи на нормальное распределение:'+svalues
    else:
        return 'Данные не порождены нормальным распределением:'+svalues

print("laser_temp_series")
print(ntest(df.laser_temp_series))
print("laser_power_series")
print(ntest(df.laser_power_series))
```

```
laser_temp_series
Данные не порождены нормальным распределением:
stat=460049.0790975577, p-value=0.0

laser_power_series
Данные не порождены нормальным распределением:
stat=247273.96052036807, p-value=0.0
```

Рисунок 17 – Тестирование распределений температуры и мощности лазера

Из рисунка 17 видно, что распределения температуры и мощности не имеют нормальную структуру.

Для более точного представления структуры данных были вычислены коэффициенты эксцесса и асимметрии.

```
laser_temp_series
Эксцесс (острота пика распределения): 4.319019244019009
Ассиметрия (искаженность данных): -2.4806179934401724
```

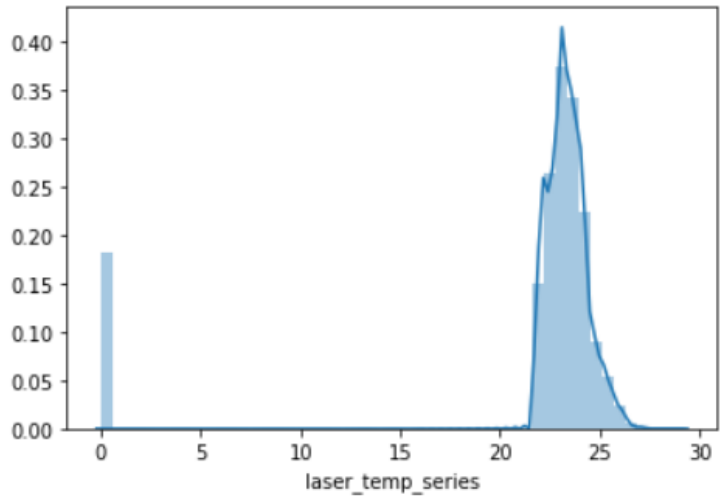


Рисунок 18 – Проверка структуры распределения температуры лазера

```
laser_power_series
Эксцесс (острота пика распределения): 1.5045682121592465
Ассиметрия (искаженность данных): 1.5639472853029839
```

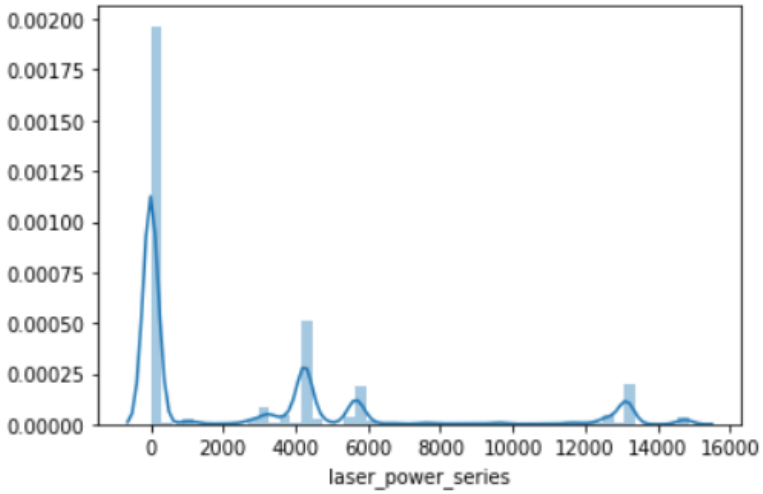


Рисунок 19 – Проверка структуры распределения мощности лазера

На рисунках 18 и 19 отображены данные по структуре распределений температуры и мощности лазера. Видно, что структура у обоих распределений комплексная, состоящая из смеси распределений с нетривиальными свойствами.

3.1.2 Анализ трендов

На рисунке 20 видно, что данные в основном имеют устойчивый тренд по месяцам. Дисперсия данных температуры лазера невысокая.

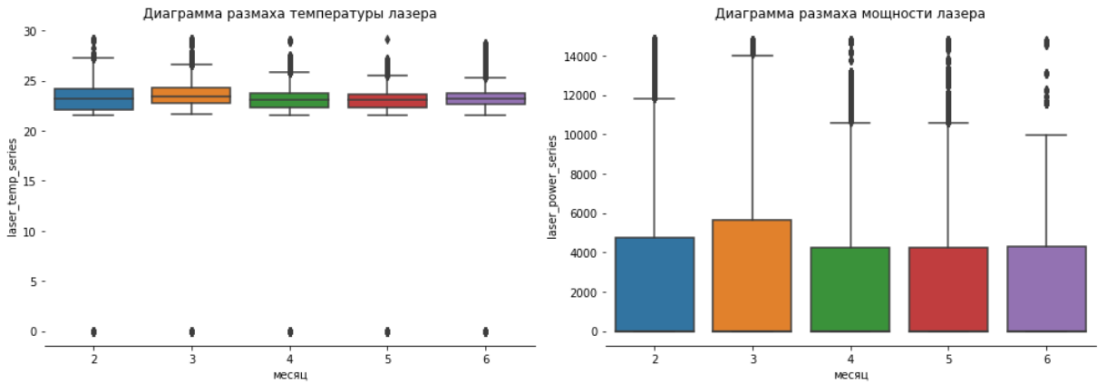


Рисунок 20 – Диаграммы размаха

На рисунках 21 и 22 отображены тренды по дням, неделям и месяцам. Заметно, что тренд мощности лазера уменьшается. Это говорит о том, что лазерная голова станка начинает изнашиваться, поэтому данный тренд необходимо отслеживать.

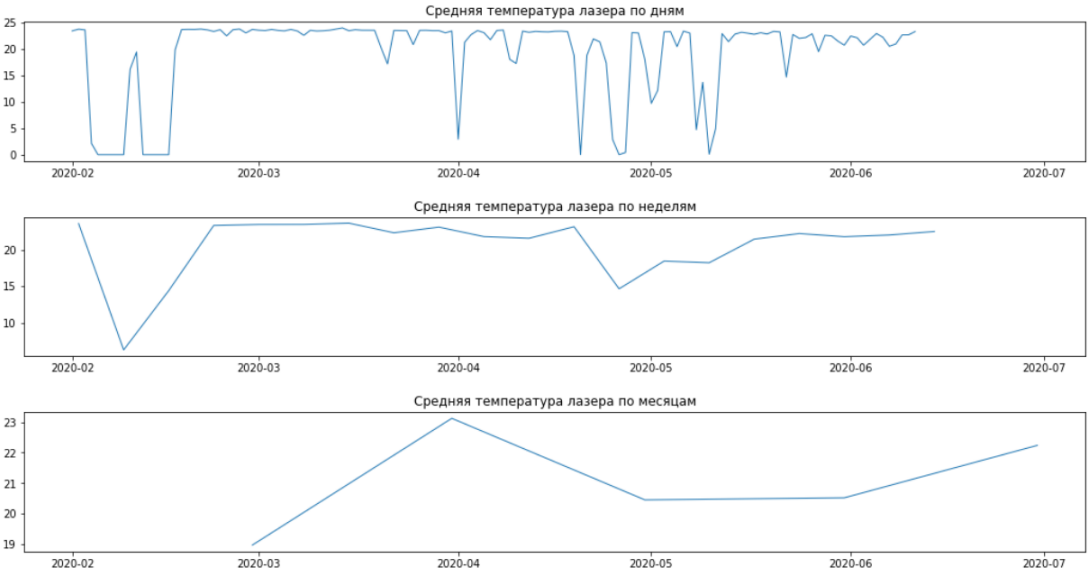


Рисунок 21 – Тренды температуры лазера

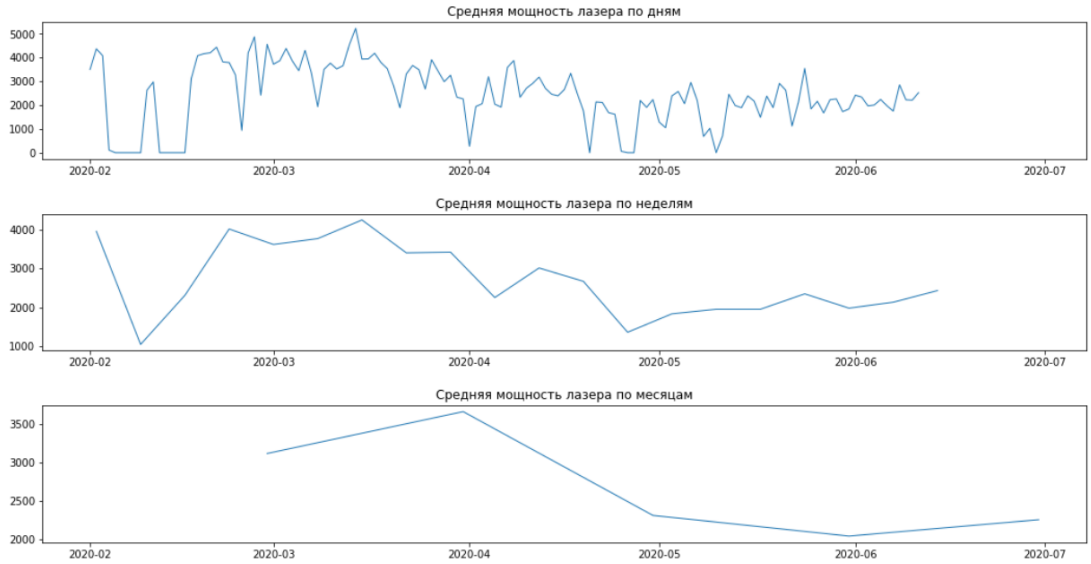


Рисунок 22 – Тренды мощности лазера

На риунке 23 отображены временные ряды за короткие период двух параметров: фактическая и установленная мощность. Видно, что установленная мощность постоянна, однако ее могут изменять. Инженеры компании ВНИТЭП также рекомендуют следить за регулярным превышением фактической мощности, что может привести к более быстрому износу, однако, как уже было описано, тренд мощности пока только падает. Для системы предупреждения важно обозначить требования, на основе которых позволялось использовать мощность лазера выше установленной в течение какого-то интервала времени.

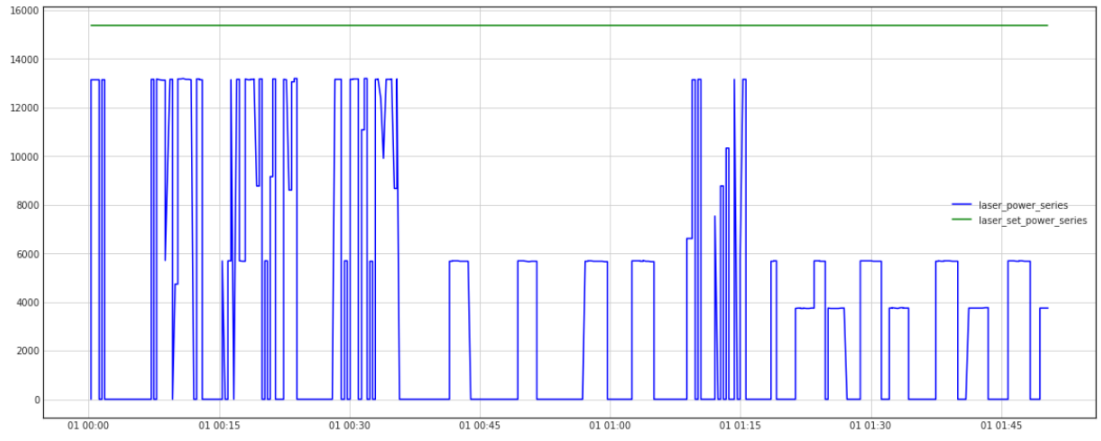


Рисунок 23 – Фактическая и установленная мощность

Рисунок 24 представляет собой отображение очевидной связи между мощностью и температурой лазера. Данные на рисунке нормализованы для наглядного сравнения.

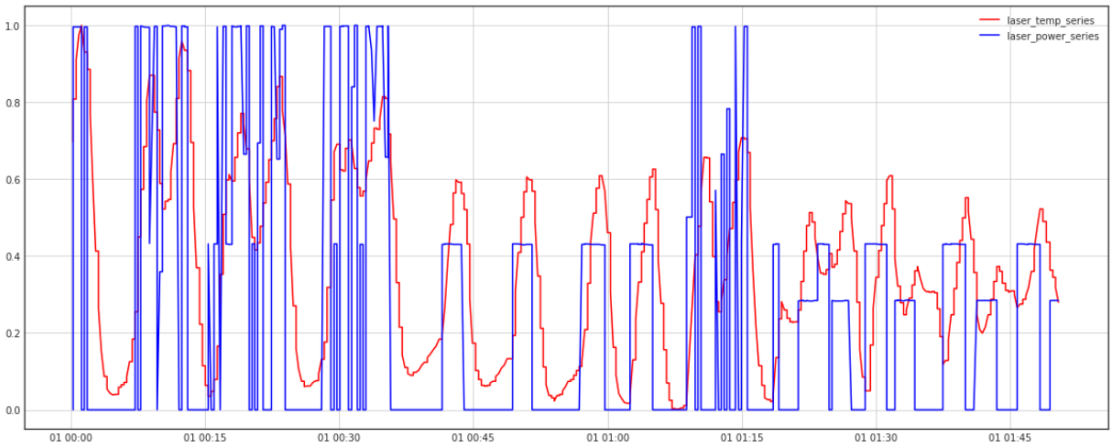


Рисунок 24 – Отношение температуры и мощности лазера

3.1.3 Тестирование на стационарность

Для того, чтобы всецело использовать дальнейшие более сложные методы, необходимо узнать являются ли исследуемые временные ряды стационарными. Именно стационарность позволяет более эффективно работать с временными рядами. Для этого можно использовать тест Дики-Фуллера. Данный тест позволяет определить меру того, насколько данный временной ряд является стационарным.

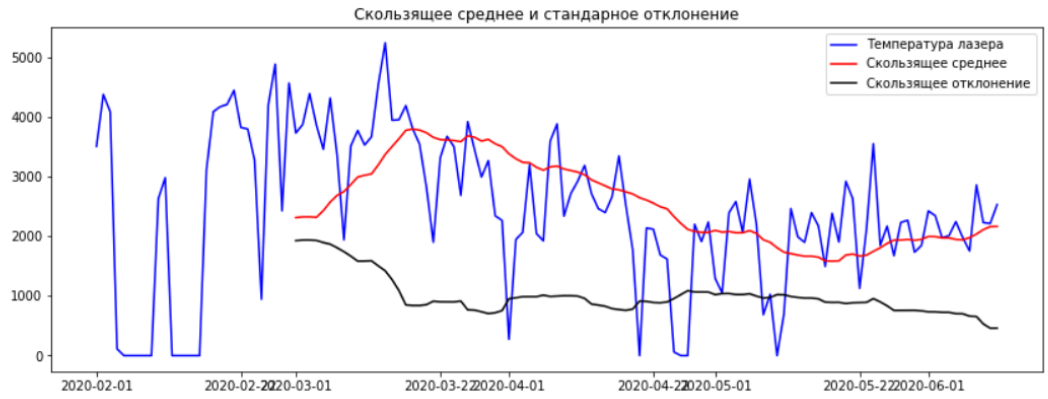
```
In [14]: """Тест Дики-Фуллера на стационарность"""

from statsmodels.tsa.stattools import adfuller

df2=df.resample('D').mean()

def test_stationarity(timeseries):
    rolmean = timeseries.rolling(window=30).mean()
    rolstd = timeseries.rolling(window=30).std()

    plt.figure(figsize=(14,5))
    sns.despine(left=True)
    orig = plt.plot(timeseries, color='blue',label='Температура лазера')
    mean = plt.plot(rolmean, color='red', label='Скользящее среднее')
    std = plt.plot(rolstd, color='black', label = 'Скользящее отклонение')

    plt.legend(loc='best'); plt.title('Скользящее среднее и стандарное отклонение')
    plt.show()

    print ('Результаты теста Дики-Фуллера')
    dftest = adfuller(timeseries, autolag='AIC')
    dfoutput = pd.Series(dftest[0:4],
                         index=['Test Statistic','p-value','#Lags Used','Number of Observations Used'])
    for key,value in dftest[4].items():
        dfoutput['Critical Value (%s)'%key] = value
    print(dfoutput)
test_stationarity(df2.laser_temp_series.dropna())
test_stationarity(df2.laser_power_series.dropna())
```

Рисунок 25 – Код для теста Дики-Фуллера

На рисунке 25 отображен код, в котором анализируются два временных ряда температуры и мощности лазера на стационарность. Оба ряда усреднены по дням.


```
Результаты теста Дики-Фуллера
Test Statistic                  -5.048146
p-value                          0.000018
#Lags Used                       0.000000
Number of Observations Used    131.000000
Critical Value (1%)             -3.481282
Critical Value (5%)             -2.883868
Critical Value (10%)            -2.578677
dtype: float64
```

Рисунок 26 – Результаты теста для температуры лазера




```
Результаты теста Дики-Фуллера
Test Statistic                  -3.688439
p-value                          0.004279
#Lags Used                       2.000000
Number of Observations Used    129.000000
Critical Value (1%)             -3.482088
Critical Value (5%)             -2.884219
Critical Value (10%)            -2.578864
dtype: float64
```

Рисунок 27 – Результаты теста для мощности лазера

На рисунках 26 и 27 отображен результат выполнения теста с дополнительными параметрами. Основным условием для принятия гипотезы стационарности было p-значения меньше $0,05$, что означает то, что у ряда нет единичных корней (характеристическое уравнение авторегрессионной модели временного ряда не имеет корней равных по модулю единице). Как видно из рисунков, ряды проявляют свойства стационарности.

3.2 Архитектура разработанного решения

На рисунке 28 отображена архитектура в виде однонаправленного графа, узлы которых являются модулями разработанного решения, а вершины – поток данных. Поток разбивается на два параллельных процесса вычислений: предсказание и генерацию признаков.

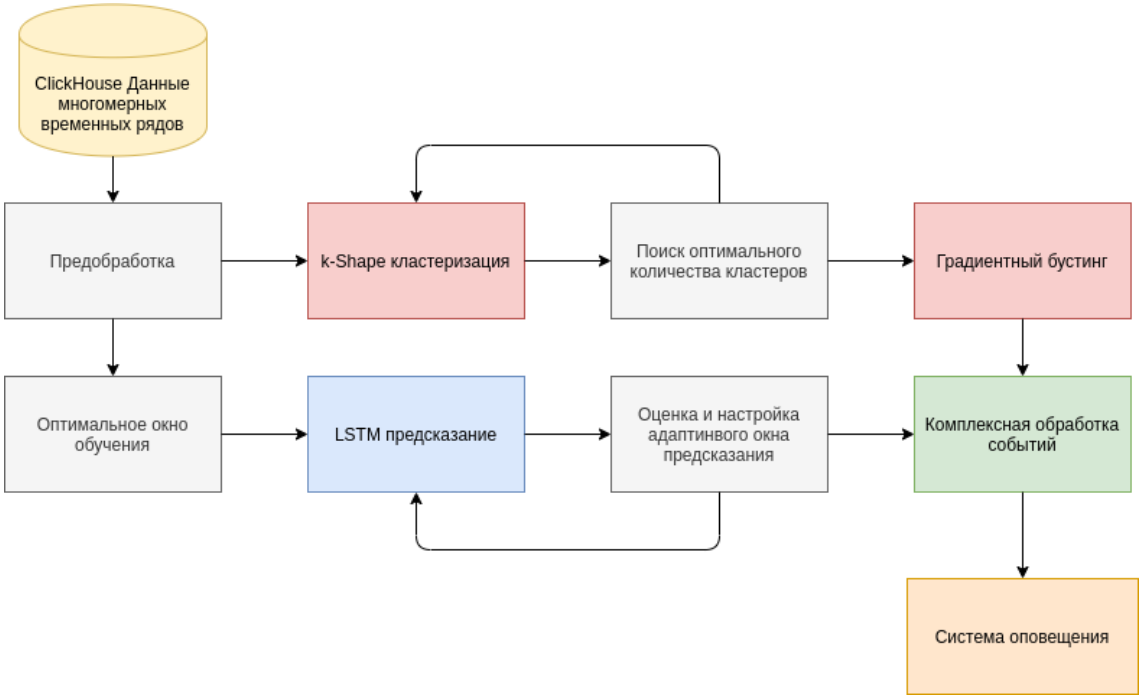


Рисунок 28 – Диаграмма архитектуры

За предсказание отвечает нейронная сеть LSTM архитектуры, с дополнительными особенностями в виде определения оптимального окна обучения для повышения точности и адаптивного окна предсказания, также для повышения точности.

Алгоритмы k-Shape и градиентный бустинг отвечают за выявление дополнительных признаков, характеризующих набор данных. Данный подход необходим для выявления дополнительных паттернов, некоторые из которых могут быть признаками неисправностей, либо близких к неисправностям состояний. Возможно, что данный блок может не помочь в поиске неисправностей, однако такой блок поспособствует лучшему понимаю данных.

Предсказанные LSTM данные и модель градиентного бустинга объединяют в модули комплексной обработки событий. Моделью градиентного бустинга и некоторыми наложенными условиями определяется являются ли предсказанные отрезки времени потенциальными неисправностями или сбоями, либо нормальным состоянием. В случае выявления неисправностей,

в зависимости от рода неисправностей, происходит обработка результатов и конечные данные отправляются в систему оповещения о неисправностях.

Поток данных может обрабатываться рекуррентно в блоках предсказания и кластеризации с целью повышения точности.

3.3 Обучение LSTM

Обучение моделей на временных рядах может происходить различными способами. Основным способ является обучение на основе скользящего окна. Предположим, что есть временной ряд $TS = \{x_1, ..., x_n\}$. Тогда, скользящим окном называется фрагмент ряда TS размера m с шагом s. Тем самым, исходный ряд разбивается на сегменты одинакового размера: $TS = \{S_1, ..., S_{\frac{n}{s}}\}$. В данной работе оптимальный размер окна вычисляется на основе периодограмма построенной с помощью метода Ломба-Скаргл, который был описан в предыдущем разделе. Также стоит отметить, что общее окно обучения, то есть отрезок всего временного ряда выбран максимальным, так как нейронные сети лучше обучаются на большем количестве данных.

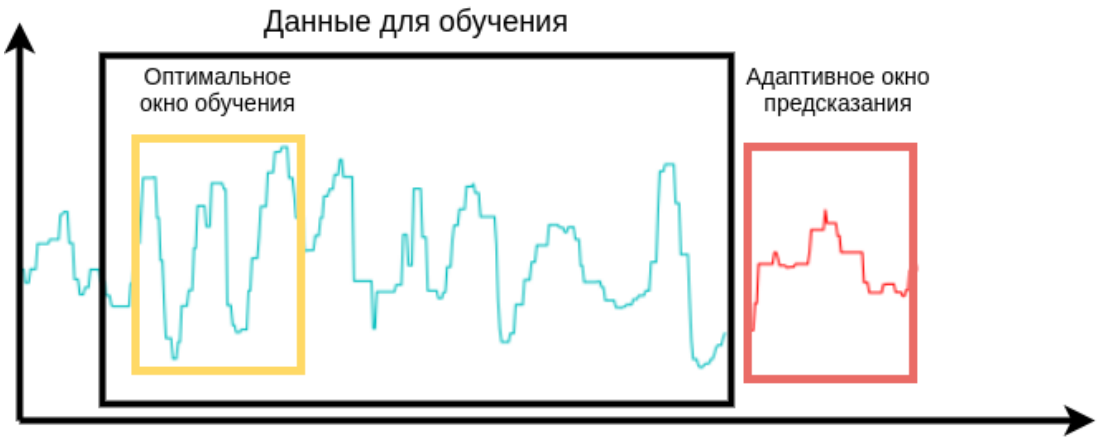


Рисунок 29 – Описание различных окон временного ряда

На рисунке 29 описано примерное представление различных окон, которые могут настраиваться на основе тех или иных критериев оптимальности. На диаграмме также присутствует так называемое адаптивное окно

предсказания, которое определяется на основе наиболее меньшего значения функции потерь.

function PREDICTIONWINDOW(y_{act}, y_{pred})
  $MAPE = mean(abs((y_{act} - y_{pred})/y_{act}) * 100)$
  if $MAPE > 20\%$ **then**
    $PredictionWindow = PredictionWindow - 1$
  else if $MAPE < 5\%$ **then**
    $PredictionWindow = PredictionWindow + 1$
  else
    $PredictionWindow = PredictionWindow$
  end if
  return $PredictionWindow$
end function

Рисунок 30 – Алгоритм адаптивного окна предсказания

$$MAPE = \frac{1}{n}\sum_{t=1}^{n}\left|\frac{A_t - F_t}{A_t}\right| \tag{18}$$

На рисунке 30 описан псевдокод программы для вычисления адаптивного окна предсказания. MAPE (mean absolute percentage error) – средняя абсолютная процентная ошибка, являющейся мерой точности предсказания и выражается по формуле 18, A_t – реальное значение, а F_t – предсказанное.

```
import scipy.signal as signal

df_days = df.resample("H").mean()
f = np.linspace(0.01, 1, 1000)

pgram_laser_temp = signal.lombscargle(
    df.tail(1000).index, df.tail(1000).laser_temp_series, f, normalize=True)
pgram_laser_power = signal.lombscargle(
    df.tail(1000).index, df.tail(1000).laser_power_series, f, normalize=True)
```

Рисунок 31 – Код для использования метода Ломба-Скаргл

В программном коде на рисунке 31 происходит вычисление периодограмм для усредненных по часам временных рядов температуры и мощности лазера. Для этого использовалась библиотека Scipy. Данные автоматически

нормализуются и на основе заданной частоты определяются преобладающие частоты временных рядов.

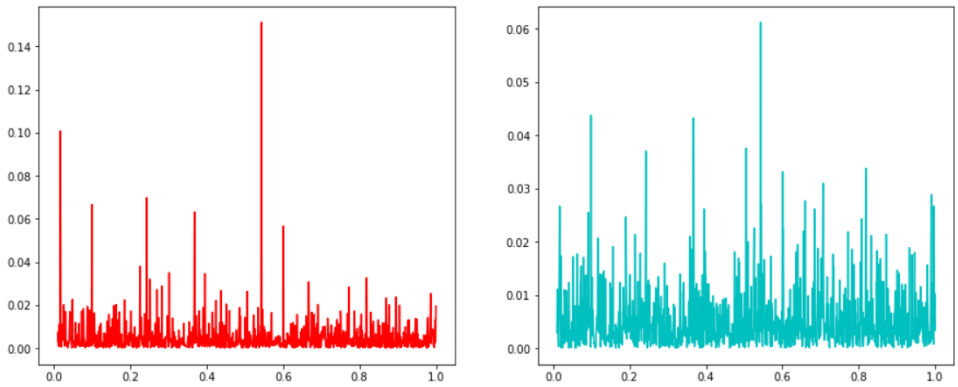


Рисунок 32 – Результат использования метода Ломба-Скаргл

На рисунке 32 видно, что сильно выделенных частот в данных почти нет, однако в обоих случаях преобладают частоты около значения 0.5, поэтому можно сказать, что оптимальным историческим окном обучения является разбиение данных на последнюю половину.

```
optimal_df = int(len(df.index)*0.5) # Оптимальное историческое окно
dataset = df.tail(optimal_df).laser_temp_series.values # numpy.ndarray
dataset = dataset.astype('float32')
dataset = np.reshape(dataset, (-1, 1))
scaler = MinMaxScaler(feature_range=(0, 1)) # Featue scaling
dataset = scaler.fit_transform(dataset)
train_size = int(len(dataset) * 0.80)
test_size = len(dataset) - train_size
train, test = dataset[0:train_size,:], dataset[train_size:len(dataset),:]

def create_dataset(dataset, look_back=1):
    X, Y = [], []
    for i in range(len(dataset)-look_back-1):
        a = dataset[i:(i+look_back), 0]
        X.append(a)
        Y.append(dataset[i + look_back, 0])
    return np.array(X), np.array(Y)

look_back = optimal_df*0.1 # Размер тренировочого окна
X_train, Y_train = create_dataset(train, look_back)
X_test, Y_test = create_dataset(test, look_back)

# reshape input to be [samples, time steps, features]
X_train = np.reshape(X_train, (X_train.shape[0], 1, X_train.shape[1]))
X_test = np.reshape(X_test, (X_test.shape[0], 1, X_test.shape[1]))
```

Рисунок 33 – Подготовка выборок температуры лазера для обучения LSTM

На рисунке 33 отображен код для подготовки данных для тренировки. В данном случае описан код с разбиением выборок на тренировочную и тестовую в соответствии с выделенными оптимальными окнами в виде $\{(\mathbf{x}_i, y_i)\}_{i=1}^{n}$. В последующем данный код будет встраиваться и динамически изменять параметры оптимальных окон обучения и предсказания на основе описанных методов.

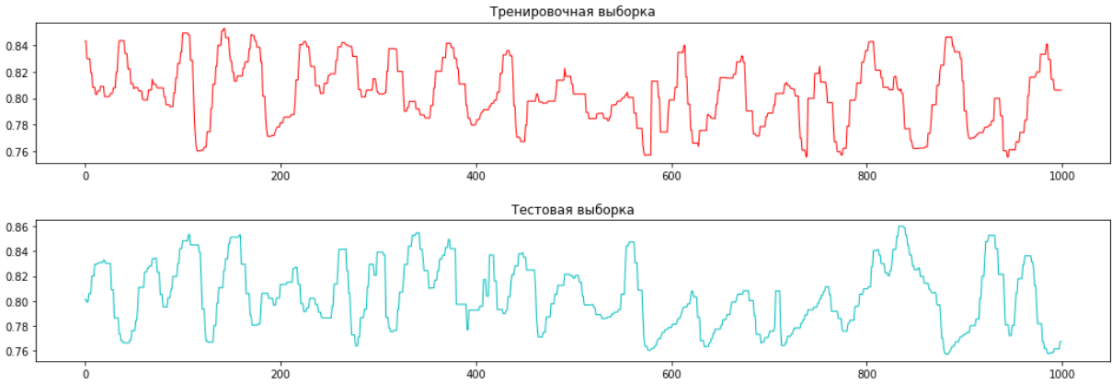


Рисунок 34 – Подготовленные выборки температуры лазера

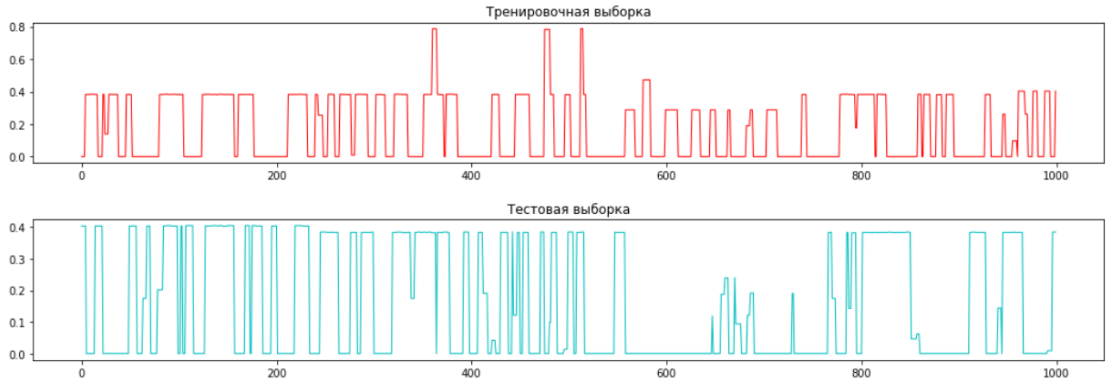


Рисунок 35 – Подготовленные выборки мощности лазера

На графиках рисунков 34 и 35 отображены первые 1000 значений тренировочных и тестовых выборок.

```
model = Sequential()
model.add(LSTM(100, input_shape=(X_train.shape[1], X_train.shape[2])))
model.add(Dropout(0.2))
model.add(Dense(1))
model.compile(loss='mean_squared_error', optimizer='adam')

history = model.fit(
    X_train, Y_train,
    epochs=10, batch_size=70,
    validation_data=(X_test, Y_test),
    callbacks=[EarlyStopping(monitor='val_loss', patience=10)],
    verbose=1, shuffle=False)

model.summary()
```

Рисунок 36 – Код для тренировки нейронной сети LSTM

Для тренировки нейронной сети LSTM был выбран ряд параметров (рисунок 36). Для самой сети было определено 100 нейронов. Такой показатель не обязательно является оптимальным, но может быть достаточно подходящим для большинства случаев.

Кроме этого, для тренировки используется метод регуляризации Dropout, который представляет собой случайное исключение определенного процента нейронов в процессе обучения нейронной сети. Данный метод нацелен на предотвращение переобучения сети, и, эмпирически, было выведено то, что такой вид регуляризации наиболее эффективен для большинства случаев. Для разрабатываемой системы был выбран Dropout в размере 20 процентов, что также как и количество нейронном не обязательно является оптимальным, однако является вполне подходящим вариантом.

Была выбрана функция потерь в виде среднеквадратичной ошибки, а также метод для стохастической оптимизации этой функции Adam. Метод Adam считается наилучшим для оптимизации функций потерь на 2020 год. Для поиска оптимального окна будет использоваться другая функция потерь, как уже было описано, а именно средняя абсолютная процентная ошибка.

Для тренировки был выбран размер отправляемых данных, который равен 70, а также количество тренировочных эпох равным 10, что является достаточным.

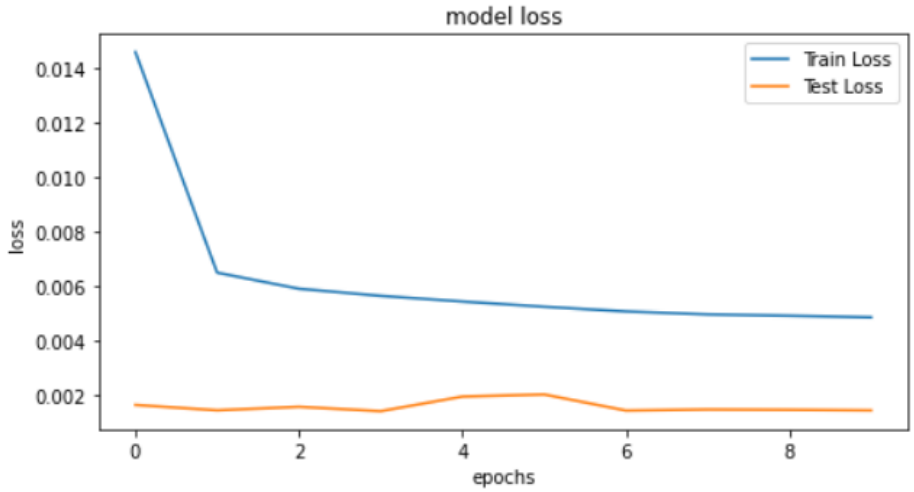


Рисунок 37 – Динамика функций потерь в процессе тренировки LSTM для температуры лазера

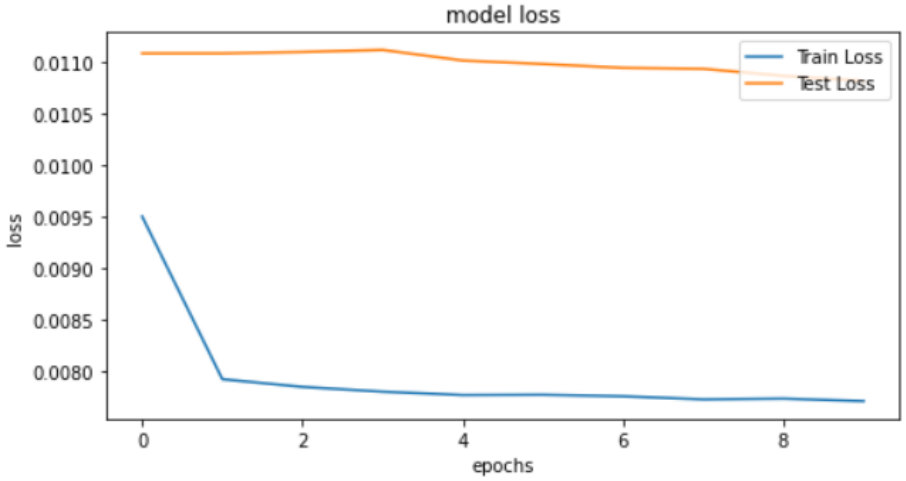


Рисунок 38 – Динамика функций потерь в процессе тренировки LSTM для мощности лазера

Графики на рисунках 37 и 38 отображают динамику изменения значений функций потерь (loss) на тренировочной и тестовых выборках в процессе тренировки по эпохам (epochs). Для данных температуры лазера минимизация прошла успешнее, чем для данных мощности лазера.

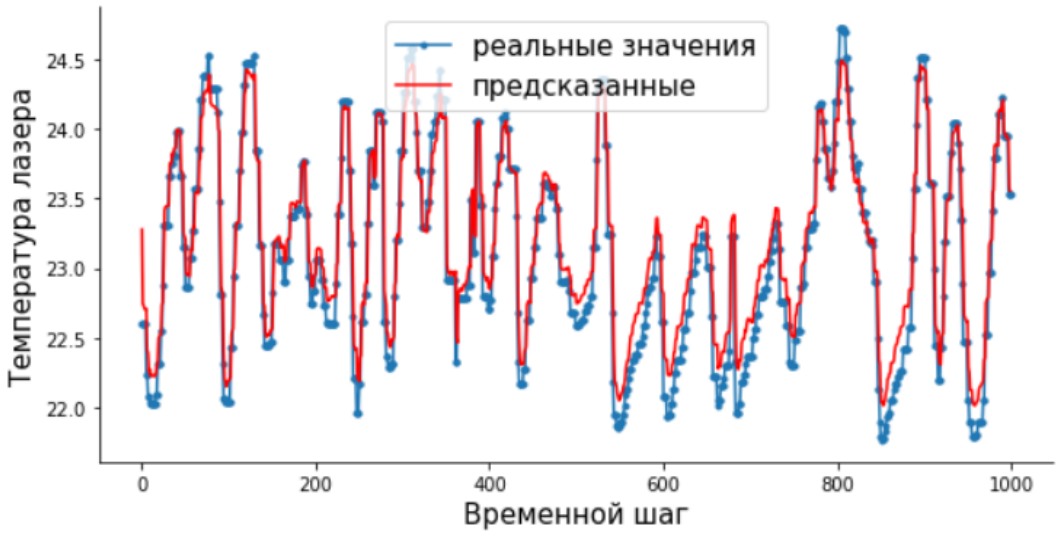


Рисунок 39 – Реальные и предсказанные значения для температуры лазера

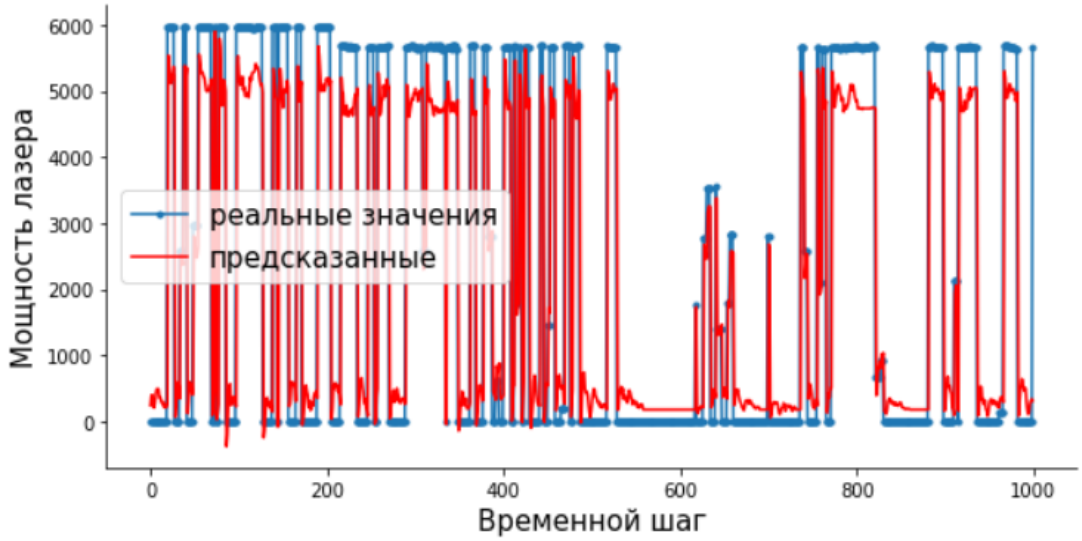


Рисунок 40 – Реальные и предсказанные значения для мощности лазера

Итоговые результаты предсказания значений отображены на графиков рисунков 39 и 40 для первых 1000 значений. Предсказанные значения для температур оказались лучше, чем для мощности. Такой результат получился по причине нетривиальной структуры временных рядов мощности лазера. Однако, такая разница не является критичной, так как температура зависит от мощности, что означает возможность использования только температурных значений.

3.4 Кластеризация на основе k-Shape

Кластеризация является экспериментальным этапом в процессе обработки данных станка лазерной резки и может быть иметь неопределенные результаты. Однако, в данной работе описаны первоначальные результаты использования алгоритма k-Shape, которые будут в дальнейшем улучшаться за счет использования оптимизационных методов.

Многие алгоритмы кластеризации требуют предварительную установку количества кластеров. Такая задача является исследовательской и полноценно пока не решена. Обычно, выбор количества кластеров зависит от специфики задачи. В задаче распознавания неисправностей сложно определить подходящее количество кластеров без использования автоматических оптимизационных методов. Однако, было определено 3 кластера: для нормального состояния, для среднего и для состояния неисправности.

На рисунке 41 отображен код для подготовки данных температуры лазера, использования алгоритма k-Shape на основе Tslearn и вывода результатов при помощи библиотеки Matplotlib. Было выбрано окно длинной 10 с шагом в половину тренировочного исторического окна. Кроме этого, данные были нормализованы на основе z-нормализации при помощи функции $TimeSeriesScalerMeanVariance()$.

На рисунке 42 отображен результат кластеризации временного ряда температуры лазера. По данным результатам видно, что выявлены три паттерна-состояния станка: снижение, повышение и стабильное состояние. На основе выявленных паттернов можно создать модель классификации для предсказанных данных, что и будет использовано для градиентного бустинга в дальнейшем.

```
batch_series = []
train_size = 10
for i in range(int(len(df_tail.index)/2)):
    batch_series.append(temp_tail[i:i+train_size])

X_train = to_time_series_dataset(np.array(batch_series))
X_train = TimeSeriesScalerMeanVariance().fit_transform(X_train)

sz = X_train.shape[1] - 1

num_clus = 3

# k-Shape кластеризация
ks = KShape(n_clusters=num_clus, verbose=True)
y_pred = ks.fit_predict(X_train)

# Отображение кластеров
plt.figure(figsize=(6,6))
for yi in range(num_clus):
    plt.subplot(num_clus, 1, 1 + yi)
    for xx in X_train[y_pred == yi]:
        plt.plot(xx.ravel(), "k-", alpha=.2)
    plt.plot(ks.cluster_centers_[yi].ravel(), "r-")
    plt.xlim(0, sz)
    plt.ylim(-4, 4)
    plt.title("Кластер %d" % (yi + 1))

plt.tight_layout()
plt.show()
```

Рисунок 41 – Код для использования алгоритма k-Shape

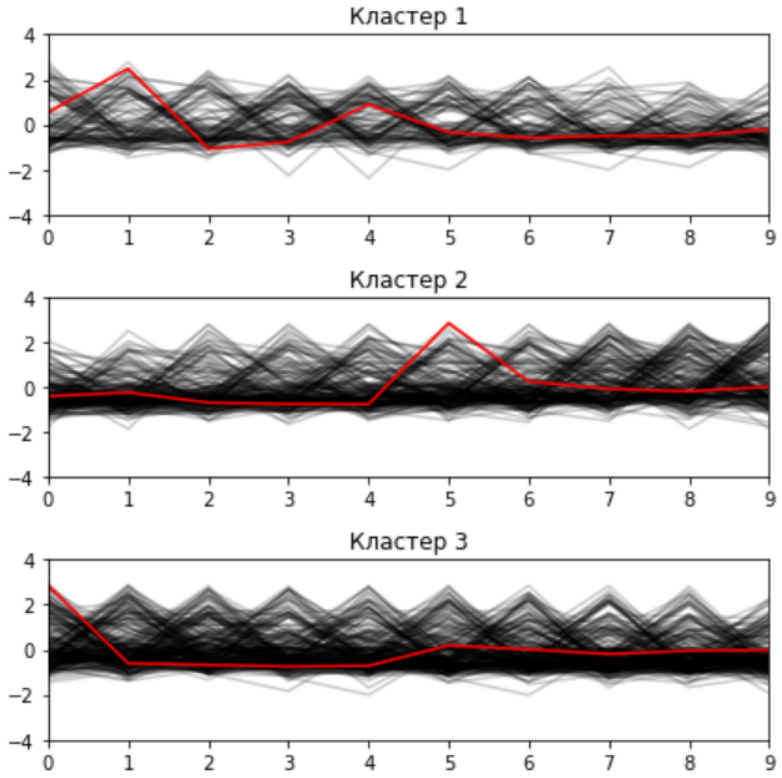


Рисунок 42 – Результаты кластеризации

3.5 Анализ предсказанных значений

Предсказанные на основе LSTM временные ряды могут использоваться для последующей проверки на выявление неисправностей. Для определения неисправностей используется анализ трендов и среднего предсказанных значений.

Критичным для значения температуры является стойкая высокая температура выше 120 градусов, и стойкая низкая температура ниже 30 градусов по Цельсию, на протяжении длительного времени (около суток). Для мощности лазера необходимо отслеживать тренд ее снижения на протяжении суток, а также превышение используемой мощности выше заданной в течение минуты. Данные ограничения для исследуемых параметров станка Навигатор определены на основе инженерных требований разработчиков станка компании ВНИТЭП.

Описанные ограничения будут реализованы в мироксервисе Omniprocessing на основе простого подсчета среднего с использованием микросервиса статистики Gorobots, а также на основе простых условных конструкций, написанных на языке Python.

3.6 Введение в эксплуатацию

Вся описанная архитектура разработанного модуля для предсказания и определения неисправностей будет интегрирована, как уже было описано, в микросервис Omniprocessing. Однако, необходимо решить ряд проблем, связанных с кластеризацией и использованием алгоритма градиентного бустинга, а именно определение оптимальных параметров для алгоритмов, например, определение оптимального количества кластеров. Сам алгоритм градиентного бустинга необходимо реализовывать на основе оптимизированной кластеризации, поэтому в данной работе он не был описан. Кроме этого,

сама интеграция является нетривиальной задачи и требует решения проблем связанных с сопряжением всех подмодулей с учетом среды, в которой эти подмодули интегрируются, а именно в среде фреймворка Faust. Также отдельной задачей является использование сервиса Gorobots для подсчета средних трендов и предсказанных значений.

После решения поставленных задач, разработанный модуль будет разворачиваться вместе со всей платформой компании Omnicube с использованием Kubernetes. Отчет о неисправностях будет отправлен в сервис оповещения OmniNotifier.

3.7 Выводы

На основе предобработки данных со станка лазерной резки Навигатор КС-12В был проведен первичный анализ особенностей и свойств выделенных параметров станка: температуры и мощности лазера. Для этого был проведен ряд тестов: на нормальность и на стационарность рядов.

Выявлено оптимальное окно для тренировки модели предсказания. Описано использование модели предсказания – нейронной сети LSTM. Дана характеристика гиперпараметров модели.

Описано использование алгоритма k-Shape для кластеризации, а также результаты этого использования. Выявлены паттерны состояний станка, который могут в дальнейшем будут использоваться алгоритмом градиентного бустинга. Обозначены проблемы оптимизации данного алгоритма.

В конце раздела дано описание принципов интеграции с системой компании Omnicube.

ЗАКЛЮЧЕНИЕ

Результатом данной дипломной работы является решение проблемы автоматического прогнозирования и выявления неисправностей на станках лазерной резки Навигатор КС-12В компании ВНИТЭП. Для получения результата был решен ряд задач.

Была обозначена проблема выявления ошибок и сбоев на станках лазерной резки модели Навигатор КС-12В, а также в устройствах вообще. Описаны особенности данной проблемы, а также ее теоретические аспекты. Проведен анализ существующих решений, базирующихся на различных методах, подходах и идеях. Для каждого случая была дана критика, а именно обозначены достоинства и недостатки каждого решения. Также были обозначены задачи, решения которых может помочь достичь решения проблемы выявления ошибок и прогнозирования.

Проведен анализ системы компании Omnicube, в которую будет интегрировано решение для прогнозирования и обнаружения ошибок станков Навигатор.

На основе предобработки данных со станка лазерной резки Навигатор КС-12В был проведен первичный анализ особенностей и свойств выделенных параметров станка: температуры и мощности лазера. Для этого был проведен ряд тестов: на нормальность и на стационарность рядов.

Для задач прогнозирования была выбрана рекуррентная нейронная сеть LSTM. Описаны преимущества перед другими архитектурами нейронных сетей для задач прогнозирования временных рядов. Было дано подробное описание работы сети LSTM. Также был описан принцип, по которому определяется оптимальное окно обучения. Выявлено оптимальное окно для тренировки модели предсказания. Описано использование модели предска-

зания – нейронной сети LSTM. Дана характеристика гиперпараметров модели.

Кроме этого, был проведен анализ алгоритмов кластеризации, а также выбран и описан алгоритм k-Shape для кластеризации многомерных временных рядов. Описано использование алгоритма k-Shape для кластеризации, а также результаты этого использования. Выявлены паттерны состояний станка, который могут в дальнейшем будут использоваться алгоритмом градиентного бустинга. Обозначены проблемы оптимизации данного алгоритма.

Дано краткое описание используемых для разработки инструментов, а именно анализ библиотек и фреймворков для задач анализа данных и задач машинного обучения, предназначенных для языка Python.

В конце дано описание принципов интеграции с системой компании Omnicube.

В заключении можно сказать, что все поставленные задачи выполнены, а цель дипломной работы достигнута.

СПИСОК ЛИТЕРАТУРЫ

1. Лоскутов А.Ю. Анализ временных рядов. Курс лекций. – Физический факультет МГУ, 2018. – 113 с.
2. Шитиков В.К. Классификация, регрессия и другие алгоритмы Data Mining с использованием R. – Институт экологии Волжского бассейна РАН, 2017. – 351 с.
3. Мартин Фаулер. NoSQL: новая методология разработки нереляционных баз данных. – Вильямс, 2016. – 192 с. – ISBN 978-5-8459-1829-1.
4. Marc S. Paolella. Linear Models and Time-Series Analysis. – Wiley, 2019. – 897 c. – ISBN: 978-1-119-43185-5.
5. Douglas C. Montgomery. Introduction to time series analysis and forecasting. – Wiley, 2015. – 671 c.
6. Terence C. Mills. Applied Time Series Analysis. A practical guide to modeling and forecasting. – Academic press, 2019. – 356 c. – ISBN: 978-0-12-813117-6.
7. George Box. Time series analysis. Forecasting and control. – Wiley, 2016. – 709 c. – ISBN 978-1-118-67502-1.
8. Mohammed J. Zaki. Data mining and analysis. // Fundamental Concepts and Algorithms. – Cambridge university press, 2018. – 562 c. – ISBN 978-0521766333.
9. Zhiqiang Ge. Multivariate Statistical Process Control. – Springer, 2019. – 203 c. – ISBN 978-1-4471-4513-4.
10. William W.S. Wei. Multivariate Time Series Analysis and Applications. – Wiley, 2019. – 528 c.

11. Zhihua Zhang. Multivariate Time Series Analysis in Climate and Environmental Research. – Springer, 2018. – 293 c. – ISBN 978-3-319-67339-4.

12. Ruey S. Tsay. Multivariate Time Series Analysis. – Wiley, 2016. – 522 c.

13. Kamil Feridun Turkman. Non-Linear Time Series. Extreme Events and Integer Value Problems. – Springer, 2018. – 255 c. – ISBN 978-3-319-07027-8.

14. Robert H. Shumway. Time Series Analysis and Its Applications. With R Examples. – Springer, 2019. – 202 c. – ISBN 978-1-4419-7864-6.

15. Tata Subba Rao. Time Series Analysis: Methods and Applications. – Elsevier, 2017. – 777 c. ISBN: 978-0-444-53858-1.

16. Boris Mirkin. Core Concepts in Data Analysis: Summarization, Correlation and Visualization. – Springer, 2018. – 402 c.

17. Mehmed Kantardzic. Data mining. Concepts, Models, Methods, and Algorithms. – Wiley, 2020. – 661 c.

18. Charu C. Aggarwal. Frequent Pattern Mining. – Springer, 2019. – 85 c.

19. Jiawei Han. Data Mining. Concepts and Techniques. – Morgan Kaufmann Publishers, 2012. – 740 c.

20. Jure Leskovec. Mining of Massive Datasets. – Stanford, 2019. – 513 c.

21. Predictive Statistics. Analysis and Inference beyond Models. – Cambridge University Press, 2018. – 657 c. – ISBN 978-1-107-02828-9.

22. Max Kuhn. Applied Predictive Modeling. – Springer, 2018. – 615 c. – ISBN 978-1-4614-6848-6.

23. Richard Sutton. Reinforcement learning. – The MIT Press, 2018. – 548 c.

24. Mehryar Mohri. Foundations of Machine Learning. – The MIT Press, 2018. – 505 c.

25. Boyes, Hugh. The industrial internet of things (IIoT): An analysis framework. – Computers in Industry, 2018. – 1-12 c.

26. Jonathan Law. Composable models for online Bayesian analysis of streaming data. – Springer Statistics and Computing 28, 2017. – 1119–1137 c.

27. E. L. Ionides, C. Bretó, and A. A. King. Inference for nonlinear dynamical systems. Proceedings of the National Academy of Sciences of the United States of America, 103(49):18438–43, 2016.

28. Sunny Sanyal, Dapeng Wu, and Boubakr Nour. A Federated Filtering Framework for Internet of Medical Things. IEEE International conference on communications (IEEE ICC 2019).

29. Simon Haykin. Adaptive Filter Theory. – Prentice Hall, 2016. – ISBN 0-13-048434-2.

30. G. W. Stewart. Matrix perturbation theory. – Wiley, 2017. – 234 c.

31. Steven Van Vaerenbergh. Kernel Recursive Least-Squares Tracker for Time-Varying Regression. – IEE Transactions on neural networks and learning systems, vol. 23, No 8, 2018.

32. Paynabar K, Jin J, Agapiou J, Deeds P (2012) Robust Leak Tests for Transmission Systems Using Nonlinear Mixed-Effect Models. Journal of Quality Technology 44 (3):265-278

33. Grasso M, Colosimo BM, Tsung F (2017) A phase I multi-modelling approach for profile monitoring of signal data. International Journal of Production Research 55 (15):4354-4377. doi:10.1080/00207543.2016.1251626

34. Lei Y, Zhang Z, Jin J (2016) Automatic Tonnage Monitoring for Missing Part Detection in Multi-Operation Forging Processes. Journal of Manufacturing Science and Engineering-Transactions of the Asme 132 (5). doi:10.1115/1.4002531

35. Yang W-A, Zhou Q, Tsui K-L (2016) Differential evolution-based feature selection and parameter optimisation for extreme learning machine in tool wear estimation. International Journal of Production Research 54 (15):4703-4721. doi:10.1080/00207543.2015.1111534

36. Grasso M, Colosimo BM, Pacella M (2018) Profile monitoring via sensor fusion: the use of PCA methods for multi-channel data. International Journal of Production Research 52(20):6110-6135. doi:10.1080/00207543.2014.916431

37. Zerehsaz Y, Shao C, Jin J (2016) Tool wear monitoring in ultrasonic welding using high-order decomposition. Journal of Intelligent Manufacturing 30 (2):657-669. doi:10.1007/s10845-016-1272-4

38. Paynabar K, Jin J, Pacella M (2017) Monitoring and diagnosis of multichannel nonlinear profile variations using uncorrelated multilinear principal component analysis. IIE Transactions 45 (11):1235-1247. doi:10.1080/0740817x.2013.770187

39. Lu H, Plataniotis KN, Venetsanopoulos AN (2019) Uncorrelated Multilinear Principal Component Analysis for Unsupervised Multilinear Subspace Learning. IEEE Transactions on Neural Networks 20 (11):1820-1836. doi:10.1109/tnn.2009.2031144

40. Lu H, Plataniotis KN, Venetsanopoulos AN (2019) Uncorrelated Multilinear Discriminant Analysis With Regularization and Aggregation for Tensor

Object Recognition. IEEE Transactions on Neural Networks 20 (1):103-123. doi:10.1109/tnn.2008.2004625

41. Feng Ye, Zhijie Xia, Min Dai, Zhisheng Zhang Real-Time Fault Detection and Process Control Based on Multi-channel Sensor Data Fusion. Journal of Intelligent Manufacturing, 2020.

42. Adnan Akbar, Abdullah Khan. Predictive Analytics for Complex IoT Data Streams. IEEE INTERNET OF THINGS JOURNAL, VOL. 10, NO. 10, 20 2017.

43. John Paparrizos, k-Shape: Efficient and Accurate Clustering of Time Series. Energy and Buildings Volume 146, 1 July 2017, Pages 27-37 https://doi.org/10.1016/j.enbuild.2017.03.071

44. Data Mining Curriculum. – ACM SIGKDD Curriculum Committee, 2016. – 10 с.

45. Сайт компании Omnicube [Электронный ресурс]. – Режим доступа: https://www.omnicube.ru/, свободный. – (01.06.20)

46. Официальная документация Kafka [Электронный ресурс]. – Режим доступа: https://kafka.apache.org/documentation/, свободный. – (10.05.20)

47. Официальная документация ClickHouse [Электронный ресурс]. – Режим доступа: https://clickhouse.tech/, свободный. – (10.05.20)

48. Faust фреймворк [Электронный ресурс]. – Режим доступа: https://faust.readthedocs.io/en/latest/, свободный. – (10.05.20)