И.С. Лазухин[1], М.И. Петровский[2], И.В. Машечкин[3]

ИССЛЕДОВАНИЕ И РАЗРАБОТКА РЕКУРРЕНТНЫХ НЕЙРОСЕТЕЙ ДЛЯ ЗАДАЧ АНАЛИЗА ТЕХНОЛОГИЧЕСКИХ ПРОЦЕССОВ

Введение

Современные технологические процессы отличаются высоким уровнем автоматизации, а также обширным наблюдением и контролем над общим состоянием системы. Предоставляемых данных должно быть достаточно для построения точных математических моделей множества процессов системы. Такие модели позволят лучше использовать и контролировать производственный процесс и комплекс в целом.

В данной работе рассмотрен крупный нефтеперерабатывающий комплекс, работа которого регулируется десятками датчиков, предоставляющими подробную информацию о производственном процессе в каждый момент времени. Каждый контроллер системы можно представить, как временной ряд в дискретном времени, что сводит задачу моделирования производственной системы к исследованию многомерных временных рядов.

Несмотря на новизну комплекса управление, регулируется устаревшими коммерческими утилитами, основанными на простых линейных моделях и физических понятиях о сущности процесса. В недавнем прошлом уже было проведено исследование и попытка внедрения классических методов, касательно данной задачи. Однако, возникает интерес исследовать применимость более специфических, относительно современных методов исследования, нацеленных на временные ряды.

Технологический процесс

Рассмотрим производственный процесс поближе. Опустив структурное устройство конкретного завода, зафиксировав некоторый момент во времени, будем рассматривать его как автомат, имеющий некоторый набор вещественных переменных, ассоциируемых с соответствующими датчиками-контроллерами. С производственной точки зрения нефтеперерабатывающая установка имеет:

[1] Кафедра ИИТ ВМК МГУ имени М.В. Ломоносова, ivanlazuhin@mail.ru.
[2] Кафедра ИИТ ВМК МГУ имени М.В. Ломоносова, michael@cs.msu.ru.
[3] Кафедра ИИТ ВМК МГУ имени М.В. Ломоносова, mash@cs.msu.su.

- Некий набор *физических входов сырья* – характеризуются объемом подачи, обычно в тоннах в единицу времени, подачу на каждом из них можно как-то варьировать.

Пример: подача необработанного сырья (тонны) в производственном блоке 123.

- Набор *внутренних факторов* – способ воздействия и контроля процесса производства. Степень открытия клапанов, значения температуры, уровни давления – все, чем можно охарактеризовать производственный процесс изнутри.
- Набор *внешних факторов* – явления, не связанные напрямую с производственным процессом, но влияющие на него.

Пример: Давление топливного газа на установку.

- Набор *физических выходов продукта* – так же характеризуются объемом производства. Получить больше продукта при меньших затратах – хороший предлог для задачи оптимизации.
- Набор *виртуальных датчиков качества*. Эти показатели не существуют физически как датчики для снятия показаний, а рассчитываются в лаборатории. Очевидно, такие показатели не могут быть представлены в каждый заданный момент времени, при этом являясь критически важной составляющей процесса, потому их моделирование и предсказание является важным фактором. Виртуальные показатели так же характеризует т.н. величина задержки – ведь каждое лабораторное исследование требует времени на проведение и сопутствующие затраты, в том числе взятие образцов, их транспортировка и другие. Стоит заметить, что для них уже существуют готовые интерполяции на основе линейных физических моделей. Однако, качество такой интерполяции оценить весьма сложно – в силу невозможности проведения лабораторных исследований с частотой, близкой к автоматическим датчикам показания отсутствует какой-либо эталон интерполяции, отсюда же и происходит их виртуальность.

Пример: октановое число продукта на выходе номер 1.

Технологические задачи

Переменные контроллеров системы в свою очередь явно разделены на три группы по логическому предназначению:

- *Управляющие (MV)* – подмножество значений датчиков, что задаются оператором.

Пример: степень открытия специфического клапана.

- *Контролируемые (CV)* – характеризуют внутреннее состояние системы, то есть те факторы, на которые мы оказываем прямое и косвенное воздействие (в том числе, реальные отклики

управляющих переменных). К таким переменным так же относятся лабораторные исследования.

Пример: температура на выходе специфической колонны.

- *Наблюдаемые (DV)* – явления, на которые мы не можем повлиять, то есть соответствуют всем внешние факторы производства.

Пример: погодные характеристики.

Наибольший интерес представляют наборы контролируемых переменных, так как именно они описывают отклик нашей системы в зависимости от управляющих воздействий и других факторов. Такие переменные характеризуют выходы установки, то есть саму цель производства, а также важные технологические ограничения, нарушение которых может привести к серьезным последствиям. Однако, возможна зависимость внутренних переменных от внешних факторов, потому далее будем считать, что в исследовании нуждаются все переменные кроме управляющих.

Задача управления: поиск подходящих значений управляющих переменных, таких, что контролируемые переменные не превышают своих технологических порогов, при этом учитывая показания наблюдаемых переменных.

Задача оптимального управления накладывает еще одно условие – оптимальность заданного подмножества контролируемых переменных, будь то показатель качества или объем выхода.

Обе задачи сводятся к построению модели всего производственного процесса во времени.

Состояние предметной области

Несмотря на востребованность изучения индустриальных процессов с точки зрения машинного обучения – почти каждый индустриальный комплекс сегодня обладает высоким уровнем автоматизации, которая может быть подвержена анализу, достаточно тяжело найти открытые исследования и статьи, посвященные данной проблеме. Скорее всего, это обусловлено коммерческой стороной процесса и конкуренцией – отсутствие реальных открытых наборов данных и обилие коммерческих решений и исследований. Большинство публичных IT-исследований производственных процессов посвящены вопросам их информационной безопасности, однако нашлись и те, где встречается близкая нам подзадача моделирования и прогноза.

Ближайшая к нашей области исследования научная статья [1] датируется 2016 годом, в которой были успешно адаптированы рекуррентные LSTM [6] сети для задач детектирования аномалий (а именно кибератак) в работе нефтеперерабатывающего комплекса. Однако данная

задача исследовалась и апробировалась на искусственном сгенерированном наборе данных.

Другой пример индустриальной задачи [2] фокусируется на применении нейросетей в прогнозе выходного напряжения трехфазного преобразователя в целях сокращения задач оптимизации. Данное исследование опять же опирается на симуляцию как источник данных. В качестве основной модели здесь исследовались традиционные нейросети.

Еще одно исследование индустриальных кибератак [3] предлагает использовать свёрточные нейросети. В этой задаче используется набор данных, собранный на основе показателей искусственного комплекса по очистке воды, специально построенного для исследовательских задач. Однако, в подзадаче моделирования комплекса, которая ближе к нашему исследованию лучшие результаты в данной работе показала исследуемая LSTM архитектура.

Задача моделирования производственного процесса

Рассмотрим фиксированную дискретную сетку во времени:

$$T = t_1, t_2, \dots, t_M, \text{где } t_{(i+1)} - t_i = \tau = const \quad (1)$$

Полагая каждый датчик установки i за непрерывную функцию f_i от времени, а задачу интерполяции виртуальных датчиков решенной, получаем следующую проекцию:

$$F(t) = f_1(t), f_2(t), \dots, f_N(t);\ A := \{f_j(t_i)\}_{MxN} \quad (2)$$

Таким образом, мы получили матрицу показаний A, что дискретно моделирует производственный процесс на промежутке времени $[t_1, t_M]$ с частотой τ. Именно в таком виде мы будем далее представлять производственный набор данных для подачи на вход исследуемым методам. **Матрицу A будем именовать многомерным временным рядом или же набором данных (датасетом).**

Будем строить математические модели во времени для исследуемого процесса. Рассмотрим временной ряд A как множество записей A_t, где $A_t = a_{t1}, \dots, a_{tN}$. Тогда задачу прогноза состояния $t + k$, имея историю показаний размером r (соответственно, лаг модели) до момента t включительно можно поставить тремя способами:

- Построить простую одношаговую модель и последовательно применить её, тогда получим следующую последовательность предсказаний:

$$\tilde{A}_{t+1} = F(A_t, \dots, A_{t-r}) \quad (3)$$

$$\tilde{A}_{t+2} = F\left(\tilde{A}_{t+1}, A_t, \dots, A_{t-r+1}\right), \dots,\ A_{t+k} \approx \tilde{A}_{t+k}, \text{где } t \geq r \quad (4)$$

- Построить многомерную модель (обычно, рекуррентную нейросеть), обрабатывающую и возвращающую последовательности событий:

$$A_{t+k}, \dots, A_{t+k-r} \approx F(A_t, \dots, A_{t-r}), \text{где } t \geq r \quad (5)$$

- Индивидуальные модели для необходимого шага вперед:

$$A_{t+k} \approx F_k(A_t, \dots, A_{t-r}), \text{где } t \geq r \quad (6)$$

Таким образом, возникает задача подбора оптимального вида искомой функции авторегрессии F для моделирования технологических процессов. Будем искать такую целевую функцию среди регрессий на основе популярных методов машинного обучения.

Базовая модель производственного процесса

Рассмотрим базовое, предварительное решение задачи моделирования производственного процесса, построенное на основе классических методов. Оно было предложено в рамках ограниченных объемов данных и нацелено на универсальность моделирования и оптимизации процесса, используя варьируемый набор индивидуальных моделей.

Отбор данных

В рамках ограниченности данных исследуемого набора – порядок размерности по оси времени набора данных составлял несколько тысяч показаний, необходимо было вынести максимальную пользу из имеющихся показаний. По данным экспертов предметной области, установка может обладать множеством промежуточных состояний – период загрузки, период простоя, период ремонта и другие. Поэтому, первый этап построения решения – это поиск *стабильных периодов работы установки*, интервалов, пригодных для построения моделей.

Проблема усугубляется тем, что такое понятие стабильности не имеет четкого определения (более того, стабильные источники метаинформации по состояниям системы отсутствуют и потому не рассматриваются), а значит оценка результата получается весьма субъективной. Более того, в разные периоды стабильности, выбранные глобально, данные могут вести себя статистически совершенно по-разному в локальных областях. Отсюда следует несколько задач:

- Данные необходимо разбивать по признаку стабильной работы;
- Данные являются шумными, сырые показания каждого датчика требует предобработки (далее, см. раздел Предобработка).

Предлагается решение на основе простой кластеризации матрицы A показаний системы по оси времени. Для фиксированных числа кластеров n и алгоритма кластеризации, непрерывные периоды значений установки, удовлетворяющие минимальной заданной границе по длине,

автоматически объявляются стабильными. Предлагается использовать алгоритм кластеризации K-means как наиболее хорошо себя зарекомендовавший (см. Рисунок 1).

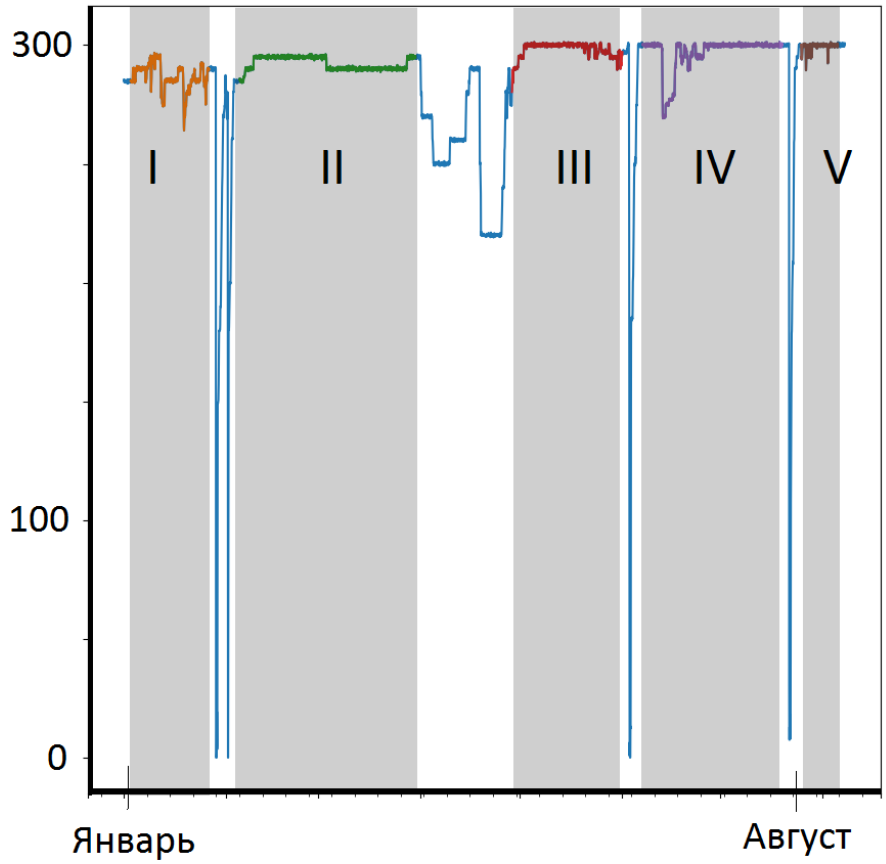

Рисунок 1 - Кластеризация данных по периодам (датчик-загрузка)

Возможным решением второй проблемы является использование простой предобработки вкупе с алгоритмами, не требующих специфических статистических параметров от входных данных.

В качестве результата данного этапа получается маска значений-индексов, соответствующая совокупности индексов меток отобранных классов. Остается только спроецировать исходные данные на полученный индекс.

Отбор признаков

Задача отбора признаков предполагает преобразование входов выбранной модели регрессора с целью ее упрощения. Как мы уже знаем, входы прогнозной модели представляют собой матрицу лагов A_r^t, где t отвечает за момент времени, а r соответствует рассматриваемому числу записей показаний, т.е. лагу (параметру) исследуемой модели.

В нашей задаче отбор входов можно проводить двумя способами. Первый способ — это сокращать размерность модели напрямую, поэлементно просеивая значения матрицы входа – тогда целевая функция должна или учитывать пропуски, или же принимать неструктурированные данные в качестве входа. Второй способ предлагает фильтровать данные по переменным, то есть по столбцам и является более универсальным, однако при этом более сложным в построении.

Целевые переменные так же участвуют двумя способами – или отбор проходит индивидуально для каждого выхода, или мы оцениваем значимость признаков многомерно. Экспериментально был предложен подход индивидуального отбора входов модели с фильтрацией поэлементно. Метод основан на построении ансамбля градиентного бустинга LightGBM [5] и последующей оценки значимости признаков, для каждого выхода независимо. Существует два критерия важности признака:

- Суммарное число вхождений признака в лес;
- Суммарный вклад разветвлений в которых участвует признак.

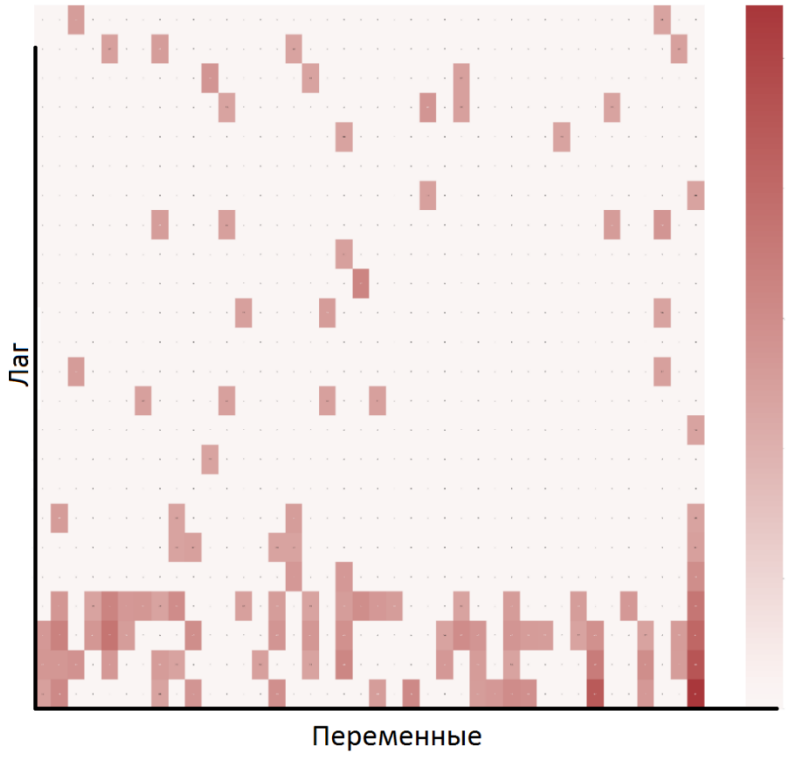

Рисунок 2 - Важность признаков по LightGBM

Такой подход позволяет сократить пространство признаков в десятки раз (см. Рисунок 3). Однако, используя его, мы теряем возможность использования целевых функций, опирающихся на непрерывность исходных данных по оси времени, в том числе сверточных и рекуррентных

сетей. Так же, признаки отбираются индивидуально, что накладывает ограничения на построение многомерной модели.

Из плюсов данного подхода стоит отметить его интерпретируемость (см. Рисунок 2) – каждому входу приписывается вес, который можно визуализировать в качестве его важности.

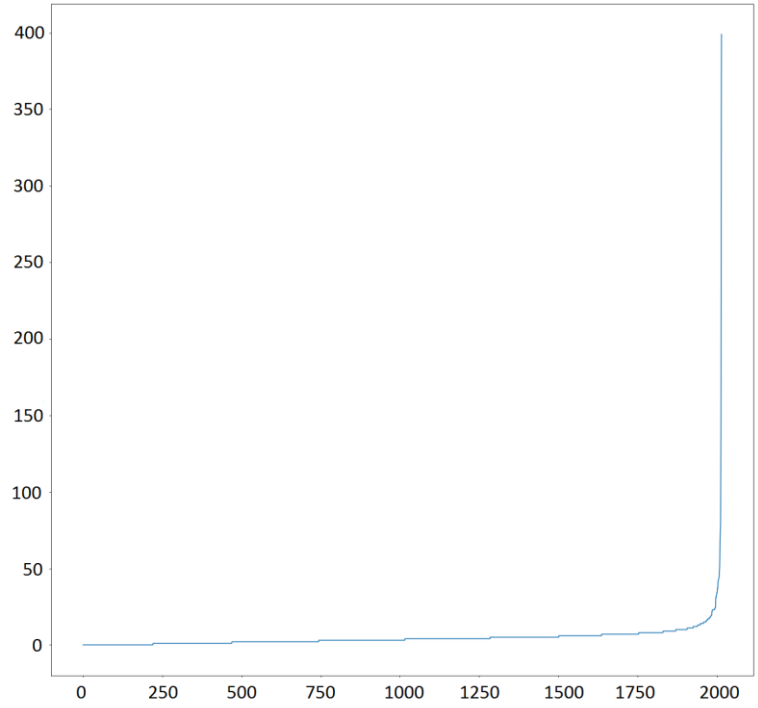

Рисунок 3 - Отсортированная важность признаков

Целевая функция

В качестве целевой функции было предложено использовать простую однослойную нейросеть – персептрон. Такая модель обладает дифференцируемостью, что позволяет применять множество методов оптимизации, а также её сложность (число нейронов на скрытом слое) легко можно варьировать. Возможна реализация дообучения моделей для поддержания их актуальности. Полученная целевая функция математически представляет собой следующую операцию:

$$Y = < V, f(< W, X >) > \quad (7)$$

V и W - параметры модели, а f - фиксированная функция.

Полученная модель позволяет унифицировать индивидуальные прогнозные модели, чтобы моделировать работу установки и оптимизировать необходимые выходы любым подмножеством датчиков. Однако используемые методы совсем не учитывают специфики задачи, а именно представления данных как многомерного временного ряда, которое теряется еще на отборе признаков. Более того, выбранной функции активации не важен порядок операндов и вообще какая-либо структура, так как перед обработкой данные всегда приводятся к «плоскому» одномерному виду.

Краткая характеристика исследуемых данных

Для исследования был предоставлен объемный набор технологических данных производственного комплекса, содержащий порядка двух миллионов записей – за промежуток с 2016 по 2019 гг., представленных с частотой $\tau = 1$ минута. Условно разделим датчики установки на несколько групп по ролям (исключительно в целях оценки качества):

- Загрузка – обычно в тоннах, потребление ресурса на специфическом агрегате, условно обозначим переменные этого типа как *Consumption*;

Пример: Расход гидрогенизата в секцию 123.

- Отношения (*Percent*) – процентные отношения, ассоциируемые с состояниями агрегатов установки, в том числе значений показаний входа и выхода;

Примеры: Вывод регулятора давления в блоке 16; Выход регулятора уровня в блоке 14.

- Давление (*Pressure*) – отвечают за давление во всевозможных камерах и колоннах.

Пример: Давление топливного газа в агрегате 202.

- Температура (*Temperature*) – температурные показатели всевозможных факторов.

Пример: Температура в зоне питания агрегата 300.

- Показатели качества (*Quality*) – характеризуют продукцию, обычно являются виртуальными датчиками.

Примеры: Октановое число катализата; Содержание ароматики в катализате.

Далее полностью заменим понятие датчика именем переменной, с ним ассоциирующейся. Рассмотрим далее меры предобработки и отбора переменных, заслуживающих исследования и включения в прогнозные и оптимизационные модели. Всего для исследования получается около полусотни переменных.

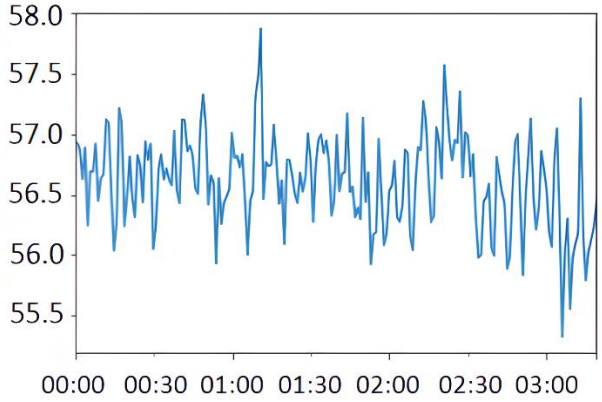

Рисунок 4 - Пример последовательности показаний переменной

Усреднение данных

Напомним, что набор данных, имеющийся в нашем распоряжении, обладает достаточно высокой частотой снятия показаний – каждую минуту. Для задач контроля и прогнозирования такая точность не понадобится – изменять параметры управления рекомендуется гораздо реже. В добавок к этому, показания на таком небольшом интервале имеют тенденцию сильно колебаться, вследствие чего являются шумными. Потому, предлагается усреднить данные индивидуально по переменным до более крупного масштаба времени – 5 минут.

Дифференцирование

Распределение большинства переменных набора далеко от нормального – гистограммы несимметричны и многомодальны. Поэтому, выглядит логичным преобразовать набор данных, при этому потеряв как можно меньше информации.

Нам известно, что производственная установка, в конечном счете, стремится к стабильной работе и загрузке – это означает отсутствие глобальных трендов, потому рассмотрим дискретные производные полученных усредненных переменных. Как видно из примеров, это преобразование весьма удобно центрирует распределение и убирает многомодальность (см. Рисунок 9-10). Дискретное дифференцирование:

$$dX_t = X_t - X_{t-1}, t > 1 \quad (8)$$

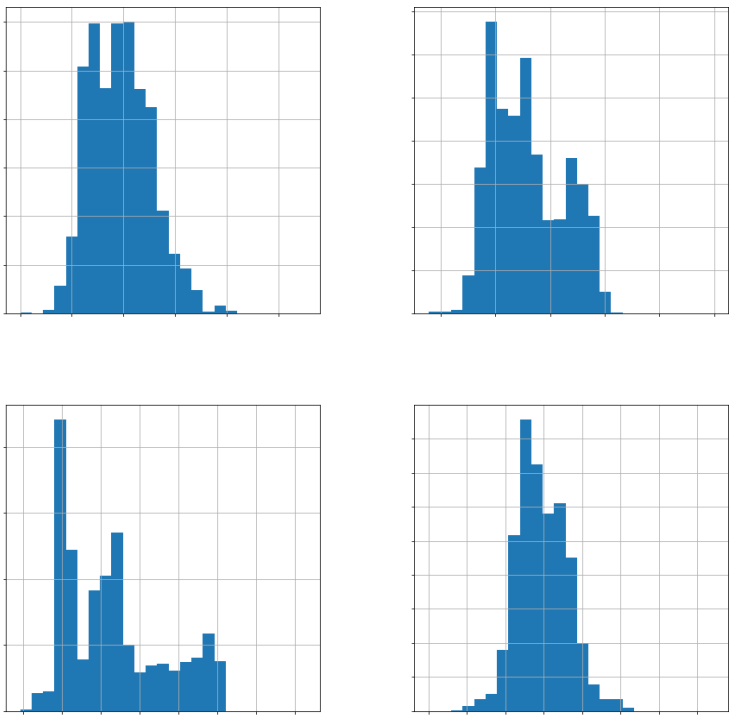

Рисунок 5 - Гистограммы некоторых переменных до дифференцирования

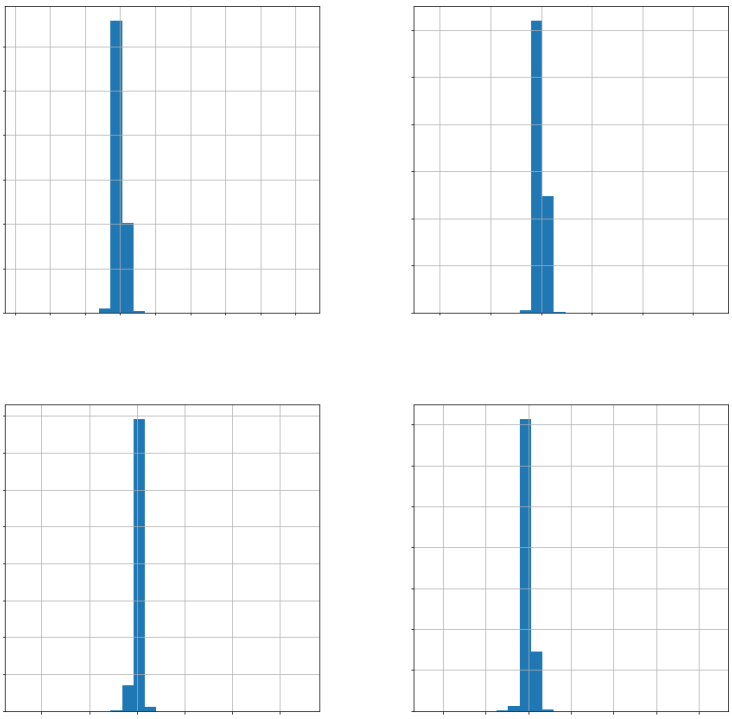

Рисунок 6 - Гистограммы тех же переменных после дифференцирования

Масштабирование

Большинство переменных установки имеют совершенно разные шкалы и единицы измерения, потому далее мы применяем стандартное масштабирование чтобы одновременно привести временные ряды на сравнимые размерности и центрировать у нуля те переменные, у которых все-таки мог присутствовать глобальный тренд. Стандартное масштабирование:

$$X_t = \frac{(X_t - \mathrm{mean}(X))}{\mathrm{std}(X)} \quad (9)$$

Отбор переменных

Для опционального отбора переменных моделей будем использовать описанную выше модель градиентного бустинга (в частности, популярную реализацию LightGBM). Это, в свою очередь, накладывает ограничение на выход модели в связи с одномерностью выхода ансамбля деревьев: получаем фиксированное соответствие одна переменная – одна модель - один шаг во времени, что соответствует первому варианту регрессии для прогноза (см. Формулы (3) (4)).

Метрики оценки качества прогноза

Одной из проблем задачи прогнозирования временных рядов в машинном обучении является тривиализация обученной модели – схождение обучаемых весов к модели сдвига $X_{t+1} \approx X_t$(см. Рисунок 7).

Общепринятые метрики MSE, MAE не отличаются сильной информативностью для человека, не знакомого с набором данных, так как не учитывают масштабирование, к тому же способствуют тривиальной модели. Определим остатки модели e_t для фиксированной переменной $x_t \epsilon X_t$ как $x_t - \tilde{x}_t$. Тогда получим:

$$MAE = mean(|e_t|) \quad (10)$$

$$MSE = mean(|e_t|^2) \quad (11)$$

Метрика MAPE отличается масштабированием, однако на нее сильно влияет амплитуда исследуемого сигнала. Если отклонение исследуемой величины во времени будет сильно меньше чем абсолютное значение сигнала, то и ошибка будет казаться маленькой. Переход к исследованию производных не всегда решает эту проблему, к тому же, данная ошибка не может корректно обрабатывать величины, близкие к нулю, что вызывает новые сложности.

$$MAPE = mean\left(\left|\frac{x_t - \tilde{x}_t}{x_t + \varepsilon}\right|\right), \quad (12)$$

где ε - достаточно малая фиксированная величина.

Поэтому вводится метрика MASE (**mean absolute scaled error**), ориентированная на проблемы прогнозирования временных рядов. Она является независимой от масштаба – так как оценивает отношение метрик, а значит и позволяет ранжировать разные переменные по точности. Эта метрика специально направлена на борьбу со моделью «сдвига». Пусть $\{y_t\}, y_t \in Y_t$ соответствует тренировочной выборке построенной модели. Тогда:

$$MASE = \frac{MAE}{\frac{1}{T-1}\sum_{t=2}^{T}|y_{t+1} - y_t|} \quad (13)$$

Однако, эта метрика не приносит никакой новизны для обучения модели – знаменатель является константной при фиксированной тренировочной выборке, и она становится эквивалентной обычной MAE. Главное ее преимущество — это наглядная интерпретируемость оценок построенных моделей, потому выберем ее в качестве основной.

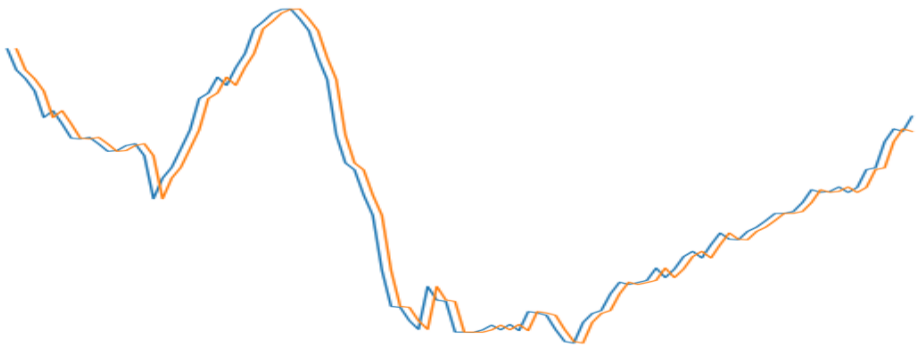

Рисунок 7 - Модель «сдвига» (прогноз смещен вправо)

Исследуемые архитектуры. Подбор параметров

Рассмотрим построение выбранных архитектур нейронной сети. Для начала зафиксируем целевую частоту, тем самым обозначая дискретизацию моделей. Для нашего основного набора данных будем проводить эксперименты на частоте 5 минут. Получаем усреднение в качестве первого этапа подготовки данных. Следующим шагом будет дифференцирование (8) и последующее масштабирование (9) переменных временного ряда.

Далее рассмотрим подбор параметров, специфичных для каждой из исследуемых моделей. Для исследования моделей в равных условиях заранее зафиксируем заданные тренировочную, валидационную (для ранней остановки построения модели в целях борьбы с переобучением) и тестовую выборку (для оценки качества).

Полносвязная нейросеть (базовая модель)

Базовая модель требует отбора признаков, так как размерности входов, а значит и число связей на внутреннем слое растут очень быстро с увеличением лага. Это же подтверждается экспериментально – лучше всего себя показывают модели со сравнительно небольшим числом входов.

Необходимо зафиксировать шаг модели. Особенности ансамблей деревьев как регрессоров (используемых для отбора признаков, см. базовая модель) накладывают ограничение на размер выхода модели, а именно одномерный выход целевой функции. Отсюда следует, что мы должны ограничиться одношаговыми моделями, индивидуальными для каждой переменной. Остановимся на простом общем случае – прогнозируем на шаг вперед.

Фиксируем исследуемую переменную, параметры модели далее подбираем индивидуально и независимо. Здесь стоит напомнить, что модели придется подбирать не только для целевых переменных с расчетом на оптимизацию, но и всех зависимых переменных (наблюдаемых и контролируемых, см. технологические задачи), попавших во множество признаков целевых переменных.

Выбираем лаг модели. Так как отбор переменных происходит путем просеивания матрицы лагов, он только косвенно влияет на сложность модели. Обозначим этот параметр модели как r. В подборе лага так же могут помочь технологические знания о работе комплекса и масштабах прогноза – вряд ли управляющее воздействие неделю назад окажет сильное влияние на значение выхода a в момент $t + 1$ через 5 минут. Заметим, что усреднение старых значений в качестве трансформации признакового пространства так же не зарекомендовало себя.

Отбор переменных - необходимо построить модель градиентного бустинга. Экспериментальные испытания в данной работе использовали реализацию LightGBM [5], как показавшую себя наиболее эффективной

для нашего набора данных. Для этого мы фиксируем достаточно свободные параметры разбиения деревьев и ограничений на число листьев, так как не боимся переобучения. Обучаем модель, пока улучшается качество на внутренней, заранее определенной валидационной выборке.

Зафиксировав способ отбора признаков, выбираем необходимую входную размерность модели – обозначим ее за d. Для наших экспериментов использовался способ отбора *split* с последующим отсечением признаков до того момента, пока их количество не будет превышать d.

Фиксируем один слой модели и функцию активации $tanh$, в качестве экспериментально подобранных. Более сложные архитектуры не зарекомендовали себя – сложные модели оказывают слабую конкуренцию простой однослойной при большей затрачиваемой вычислительной мощности на обучение.

Осталось выбрать оптимальное число скрытых нейронов – параметр модели n. Если мы в итоге захотим объединить построенные переменные в многомерную модель, то ее сложность будет соответственно $\sum n$.

Обучаем модели по метрике MAE, используя по возможности кросс-валидацию. Оптимальная тройка параметров $< r, d, n >$ предварительно подбирается по сетке для каждой из переменных.

Классическая LSTM сеть

Так же задаем единичный шаг модели. Ничего не мешает построить сложную модель, преобразующую и возвращающую сразу прогнозную последовательность, однако, оставаясь в рамках сравнения архитектур, мы будем строить общую многомерную одношаговую модель, общую для всех переменных. Таким способом мы одновременно сильно сократим сложность подбора параметров и будем учитывать кросс-зависимости при прогнозе значений разных переменных.

Фиксируем лаг модели – в рамках рекуррентных сетей он влияет только на обучение, не затрагивая сложность модели. В принципе, возможно обучение (соответственно, тестирование) на тренировочных (соответственно, тестовых) данных с разными лагами, но, в целях вычислительной эффективности, будем использовать достаточно большой неизменный лаг, который позволит предоставить необходимый объем информации о технологическом процессе для построения точной модели. Здесь мы пользуемся свойством LSTM-сети самой отбирать значимую и не значимую информацию во времени. В нашем эксперименте фиксируем лаг 64, что для установленной частоты 5 минут соответствует примерно 5 часам истории показаний.

Явный отбор переменных не используем. Возложим его на веса, задающие линейную комбинацию рядов-входов для каждого рекуррентного нейрона на первом скрытом слое. Помним, что каждый

рекуррентный нейрон получает взвешенную линейную комбинацию рядов-входов, что позволяет сделать веса при незначимых переменных достаточно маленькими.

Сложные многослойные архитектуры в этой архитектуре так же себя не зарекомендовали. Используем один рекуррентный скрытый слой в связке или с полносвязным или с масштабирующим сверточным слоем на выходе (для поддержания размерности ряда).

Используем стандартные функции активации/рекуррентной активации – гиперболический тангенс и сигмоида. Рекуррентная активация отвечает за степень учета информации при общении с внутренним состоянием, потому обычно моделируется сигмоидой (см. Рисунок 8), значения которой всегда лежат в $[0,1]$. Основная же функция активации $tanh$ тяжело заменима ввиду ряда проблем с эффективностью реализаций, ее не использующих.

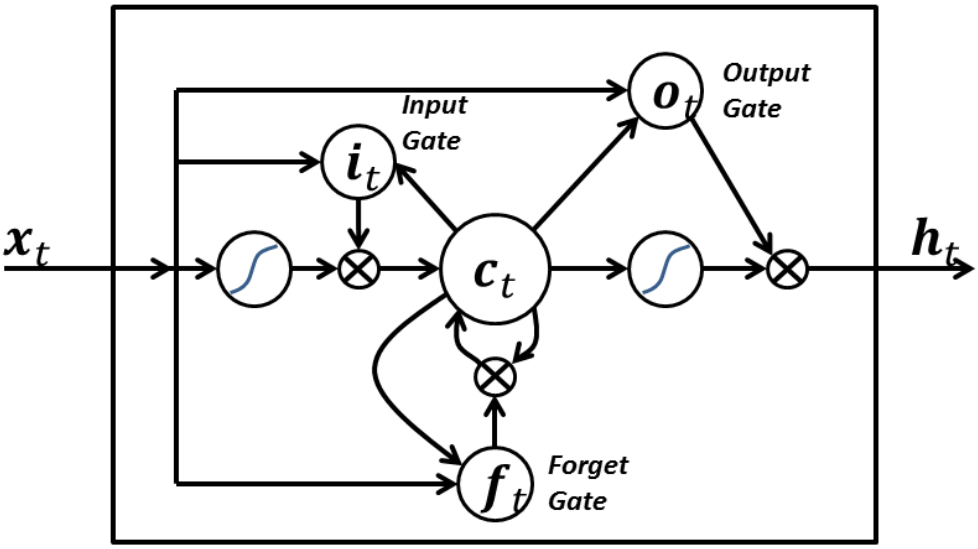

Рисунок 8 - Схема нейрона LSTM

Основным же параметром становится сложность модели, роль которой играет число скрытых нейронов n. Типичные случаи представляют собой $\{1, \sqrt{\mathrm{N}}, \mathrm{N}/2, \mathrm{N}\}$, где N – размерность входов. В проведенных экспериментах себя зарекомендовало число нейронов N, что интерпретируется тем, что за каждый ряд отвечает свой нейрон, который получает свою взвешенную комбинацию входов. Это же освобождает нас от нужды поддерживать размерность выходов дополнительными перекрестными слоями.

Отдельная задача — это интерпретация вывода сети. Рекуррентная сеть может генерировать как последовательность выходов, так и последовательность внутренних состояний t (см. Рисунок 8). Фиксируем последнее обработанное состояние в качестве выхода модели, что показало себя наиболее эффективно в задаче прогноза.

Классическая CNN-сеть

Здесь мы предложим многослойную многомерную одношаговую модель для частоты 5 минут. В случае сверточной модели выбор архитектуры оказывается нетривиальной задачей.

Одномерная сверточная модель может так же принимать любой лаг r в процессе обучения; однако, ввиду зависимости размерности на выходе от входной для сверточных сетей, получаем зависимость для лага $r = 2^k$, где k соответствует числу слоев, в случае простой схемы «пирамида» для прогнозирования на один шаг. В нашем эксперименте остановимся на лаге 64, для чистоты сравнения моделей.

Рассмотрим одномерные сверточные блоки $1D\ convolution \rightarrow ReLU \rightarrow 1D\ pooling$. Такая схема реализует преобразование размерности $< lag, n > \rightarrow < lag/2, n >$ при фиксированной конечной размерности фильтров n.

Каждый такой сверточный блок характеризуется числом обучаемых фильтров и их размером. Фиксируем стандартный размер фильтров в 3 шага. Число фильтров будет отвечать за общую сложность построенной архитектуры. В поставленном эксперименте себя зарекомендовала размерность $n = 64$ фильтра. Тогда число слоев рассчитывается как 2^r, то есть 8 слоев для нашего эксперимента.

На последнем слое сети необходимо преобразование размерности. Для этого будем использовать масштабирующий сверточный (или же полносвязный) слой. Для изучаемого набора данных себя зарекомендовал полносвязный слой с линейной активацией.

Разреженная CNN-сеть

Опциональный шаг (см. Рисунок 9) между входами сверточного фильтра называют **dilation**. Этот прием позволяет избавиться от излишней локализации обрабатываемых данных в случае классической свертки, которая является частным случаем $dilation = 1$. Популярная архитектура WaveNet, основанная на таких свертках успешно обрабатывает аудиоданные. Потому рассмотрим так же расширение [7] этой модели для работы с временными рядами.

Строим индивидуальные одношаговые модели прогноза, в соответствии с предложенной архитектурой. Для вычислительной эффективности при обучении будем рассматривать модель как многомерную, представляющую собой конкатенацию N независимых моделей с общим входом, где N – число переменных. Рассматриваемая архитектура так же в перспективе не ограничена числом прогнозных шагов.

Здесь возникают аналогичные понятия относительно структуры сети по сравнению с классической сверточной моделью. $r = 2^k$, где k -число

слоев, а r - лаг модели . Зафиксируем лаг и число слоев, аналогичные классической модели.

Модель строится на основе постоянно увеличивающихся окон в обучаемом фильтре при повышении слоя модели, а именно: $dilation = 2^n$ на слое n.

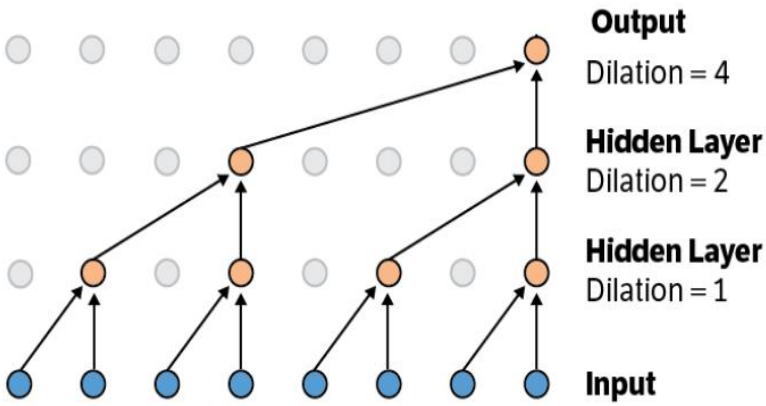

Рисунок 9 - Dilated CNN (из оригинальной статьи [7])

Основной параметр модели – сложность, задается в виде числа фильтров на каждом слое. Здесь, как и в оригинальной работе будем считать его равным на каждом слое модели.

GAN-сеть

Так же, экспериментально рассмотрим генеративный подход применительно к данной задаче. Данный комплекс методов не привносит новой идеи регрессора, способного формировать прогнозный ряд, но предоставляет новый подход к обучению модели генерировать правдоподобные данные. Процесс обучения подразумевает построение двух нейросетей:

- Генератор: $G(x)$ – генерирует выходные вектора, обычно на основе случайных последовательностей;
- Дискриминатор: $D(x)$ – его задача отличить данные, сгенерированные моделью от настоящих.

Задача обучения сети GAN – научить модель «обманывать» дискриминатор, то есть генерировать данные, не отличимые от подлинных. Обучение порождающей сети состоит в максимизации функционала $D(G(z))$. Обучение классифицирующей – это максимизация $D(x)(1 — D(G(z)))$.

Для задачи прогноза в ролях классификатора и генератора поставим LSTM архитектуры. Это позволяет воспользоваться свойством рекуррентной сети обрабатывать последовательности любой длины.

Будем обучать генератор отображать случайное пространство размерности n (соответственно лаг и размерность пространства белого шума) на пространство (r, N) (соответственно лаг и число переменных):

$$(a_n, \dots, a_n)_r \to A_{r \times n} \quad (14)$$

При прогнозе мы будем решать задачу восстанавления «кода» a_n входной последовательности, далее, применив генератор, мы сможем получить искомый прогноз:

$$(a_n, \dots, a_n)_{r+k} \to A_{r+k \times n} \quad (15)$$

В качестве классификатора зададим простую однослойную рекуррентную сеть, с полносвязным выходным слоем одномерной сигмоидной активации. Параметр модели – размер скрытого слоя.

В качестве генератора предлагается один скрытый рекуррентный слой и слой линейного масштабирование на выходе (сдвиг и множитель для каждого выхода). Параметр модели – размерность входного пространства «шума».

Экспериментальное исследование

Постановка эксперимента: исследование и сравнение прогнозных моделей на наибольшем стабильном промежутке набора данных. В поиске такого промежутка нам поможет кластеризация на основе K-means (см. базовая модель). Пригодным для эксперимента выберем промежуток данных за шесть месяцев (см. Рисунок 10, штриховка), попадающий в кластер, отвечающий стабильным промежуткам. Таким образом, получаем около 200 тыс. показаний частоты 1 минута.

Задача моделей: многомерный прогноз на один шаг для всех переменных, усредненных до частоты дискретизации 5 минут. Будем рассматривать пять архитектур: базовую (MLP), рекуррентную (LSTM), простую сверточную (CNN), WaveNet-подобную (dilated CNN) и GAN на базе LSTM.

Построение метрик качества и выбор лучшей модели проведем на основе кросс-валидации для временных рядов. Для этого используем 4 фолда в целом (см. Рисунок 11), группируем фолды по два последовательных, соответственно назначаем им роли – обучение и валидация (тестирование, валидационная выборка будет выбираться из обучающей). Результирующая оценка качества – это среднее значение метрик по всем фолдам.

Рассматриваемые метрики качества прогноза: MAE, MAPE, MASE. Приведем результаты эксперимента, разбив переменные на группы по единицам измерения. Результаты представлены в исходной размерности тестовых данных (см. Таблицы 1-5). Как видно, на текущем этапе почти везде побеждает LSTM, так что именно эта архитектура заслуживает дальнейшего развития в рамках этой задачи.

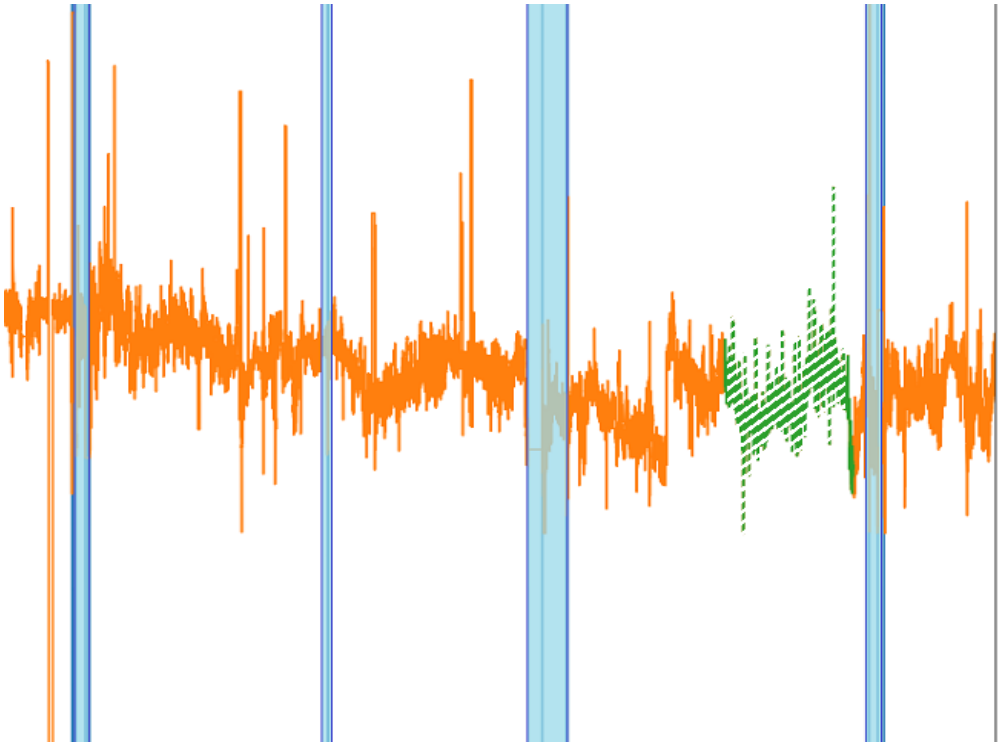

Рисунок 10 – Экспериментальный набор данных. Выбор периода.

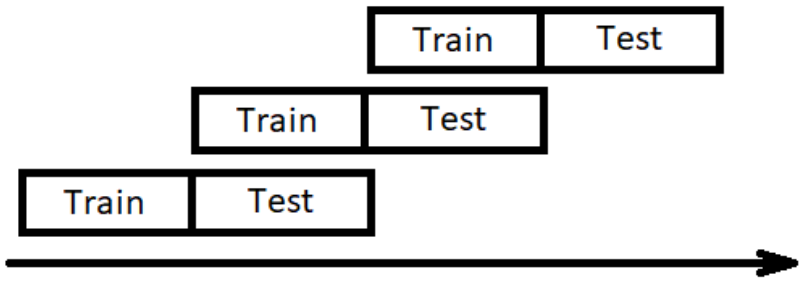

Рисунок 11 – Кросс-валидация для временных рядов

Таблица 1 – MASE для переменных загрузки

Группа	cnn	cnn_dilated	gan	lstm	mlp	BEST
consumption_01	0.703173	0.567972	0.642778	**0.540197**	0.665821	lstm
consumption_02	0.714433	0.663543	0.733393	**0.550417**	0.699411	lstm
consumption_03	0.722089	0.738251	0.732189	0.753565	**0.702587**	mlp
consumption_04	0.844797	0.820001	0.852101	**0.780569**	0.801934	lstm
consumption_05	0.835523	0.832826	0.809576	**0.795012**	0.821582	lstm

Таблица 2 – MASE для переменных, отвечающих за давление

Группа	cnn	cnn_dilated	gan	lstm	mlp	BEST
pressure_01_LAB	0.634155	**0.330023**	0.715535	0.345607	0.61587	cnn_dilated
pressure_02	0.679959	0.38729	0.72924	**0.353157**	0.661035	lstm
pressure_03	0.796106	0.644526	0.835951	**0.541588**	0.779184	lstm
pressure_04	0.794391	0.650361	0.822071	**0.542184**	0.776182	lstm
pressure_05	0.699386	0.659154	0.729057	**0.581209**	0.683815	lstm
pressure_06	0.68274	0.658988	0.653611	**0.612036**	0.633138	lstm
pressure_07	0.790195	0.707941	0.741544	**0.656808**	0.78175	lstm
pressure_08	0.732766	0.717187	0.73183	**0.67208**	0.720569	lstm
pressure_09	0.729238	0.746533	0.765775	0.762919	**0.707452**	mlp
pressure_10	0.832385	0.808749	0.7994	**0.75956**	0.823074	lstm
pressure_11	0.836028	0.83105	0.875383	**0.796587**	0.819878	lstm

Таблица 3 – MASE для переменных, отвечающих за качество

Группа	cnn	cnn_dilated	gan	lstm	mlp	BEST
quality_01_LAB	0.691976	0.596602	0.757797	**0.477806**	0.654611	lstm

Таблица 4 – MASE для переменных температуры

Группа	cnn	cnn_dilated	gan	lstm	mlp	BEST
temperature_01_LAB	0.650445	0.551393	0.667105	**0.442277**	0.625931	lstm
temperature_02	0.78732	0.533988	0.719712	**0.481046**	0.779924	lstm
temperature_03	0.668768	0.575998	0.765417	**0.523696**	0.625167	lstm
temperature_04	0.705313	0.605855	0.784144	**0.528535**	0.664736	lstm
temperature_05	0.72906	0.591166	0.762232	**0.528579**	0.714442	lstm
temperature_06	0.684983	0.590734	0.761733	**0.541449**	0.646045	lstm
temperature_07	0.742938	0.654762	0.772029	**0.556119**	0.720162	lstm
temperature_08	0.779631	0.642921	0.754465	**0.562608**	0.772379	lstm
temperature_09	0.703226	0.626629	0.750289	**0.569749**	0.689358	lstm
temperature_10	0.847339	0.6294	0.800256	**0.574069**	0.833943	lstm
temperature_11	0.804945	0.664067	0.815476	**0.5878**	0.796553	lstm
temperature_12	0.693711	0.689453	0.719686	**0.589393**	0.67779	lstm
temperature_13	0.685857	0.672019	0.747017	**0.599209**	0.651277	lstm
temperature_14	0.873767	0.69333	0.812022	**0.616915**	0.86011	lstm
temperature_15	0.714917	0.730469	0.728547	**0.641869**	0.695801	lstm
temperature_16	0.738084	0.73006	0.714678	**0.650148**	0.727064	lstm
temperature_17	0.808236	0.768728	0.832644	**0.652061**	0.784464	lstm
temperature_18	0.840013	0.744721	0.819308	**0.699669**	0.830218	lstm
temperature_19	0.841843	0.841897	0.833541	**0.7687**	0.831273	lstm

Таблица 5 – MASE для переменных, отвечающих за отношения

Группа	cnn	cnn_dilated	gan	lstm	mlp	BEST
percent_01_LAB	0.634881	0.389263	0.714734	**0.343591**	0.613724	lstm
percent_02	0.579776	0.42793	0.645853	**0.358343**	0.55887	lstm
percent_03	0.665998	0.419339	0.732452	**0.409785**	0.643856	lstm
percent_04_LAB	0.67086	0.598652	0.742991	**0.489516**	0.630152	lstm
percent_05	0.681194	0.62545	0.689888	**0.585115**	0.668194	lstm
percent_06	0.773002	0.727906	0.764936	**0.66717**	0.760483	lstm
percent_07	0.830341	0.733988	0.789603	**0.68518**	0.825582	lstm
percent_08	0.802698	0.778014	0.834582	**0.699672**	0.783298	lstm
percent_09	0.774476	0.741867	0.761753	**0.721028**	0.766523	lstm
percent_10	0.863754	0.790602	0.814982	**0.734367**	0.853273	lstm
percent_11	0.851575	0.808591	0.804122	**0.768275**	0.84162	lstm
percent_12	0.922645	0.878545	0.895744	**0.838765**	0.908859	lstm

Задача оптимизации

Предположим, мы имеем некоторое множество $Target$ «интересных» переменных системы (т.е. подмножество контролируемых переменных, см. Технологические задачи) и соответствующий им набор уже построенных одинарных/многомерных моделей y_i. Для простоты ограничимся одношаговыми моделями. Рассмотрим общую многомерную модель процесса – для этого сконкатенируем выходы имеющихся моделей. Для каждого фиксированного момента времени t выбранной дискретной частотной сетки, такая модель принимает на вход некоторый набор варьируемых признаков x и выдает значение прогноза, т.е. набор значений датчиков в момент $t+1$. Такой набор признаков уже был описан под именем переменных управления, именно их значения в момент t образуют значимые входы модели в момент t – все остальные входы модели уже являются историей и варьированию не подлежат – заморозим их и будем работать с моделями $y_i := y_i^t(x)$.

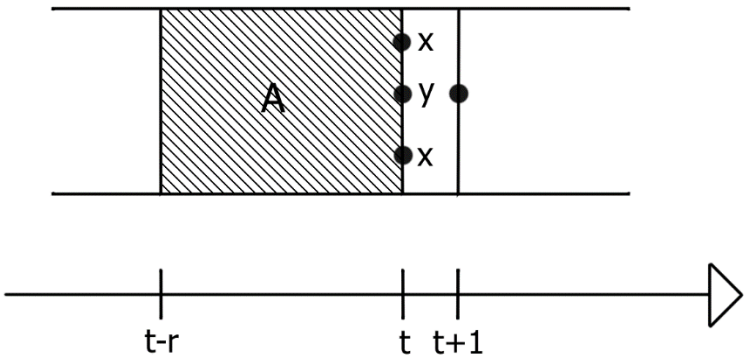

Рисунок 12 – Процесс оптимизации

Пусть каждая переменная имеет собственный набор ограничений, выходить за которые она не должна. Пусть существует подмножество целевых переменных, каждой из которых сопоставлен ее вес w_i (напр. показатель качества соответствующего продукта). Тогда необходимо в каждый момент времени t уметь найти допустимое оптимальное значение переменных управления, которое бы максимизировало взвешенную сумму выходов при условии на ограничения для внутренних переменных. Целевую функцию будем моделировать линейной комбинацией выходов многомерной модели, а ограничения – проекциями на соответствующие переменные. Так же, учтем технологические ограничения как функции $lower$ и $upper$:

$$\begin{cases} argmax_x \sum_{y_i \in Target} y_i(x) w_i : lower(x) < x < upper(x) \\ \forall y_i = y_i(x) \in CV : lower(y_i) < y_i(x) < upper(y_i) \end{cases} \quad (16)$$

Предположим непрерывность и дифференцируемость рассматриваемой многомерной модели. Из рассмотренных в экспериментальном исследовании, к таким моделям относятся CNN модели, LSTM сети и классические нейронные сети MLP. Каждую модель можно развернуть в виде направленного графа с непрерывно-дифференцируемыми функциями в узлах (рекуррентные модели так же можно промежуточно представить в виде сети прямого распространения).

Для решения поставленной задачи оптимизации в случае этих целевых функций было решено использовать метод внутренней точки. Они (следовательно, и их граничные условия) являются достаточно гладкими для применения этого метода. В качестве базы для испытаний данного метода был использован функционал библиотек *tensorflow* [10] (для работы с графами моделей) и *scikit-learn* [11] (реализация методов оптимизации).

Так как построенная задача оптимизации — задача оптимизации выхода некоторой гладкой функции с дополнительными ограничениями, которая в общем случае может возникнуть в любой другой области, рассмотрение оптимизационного подхода в данном исследовании по большей части ограниченно практическим подходом.

Очевидно, алгоритм находит какое-либо оптимальное значение для заданной гладкой функции, однако на данный момент мы не можем оценить его значимость. Более того, успешность поставленного подхода можно будет установить только после дальнейшей экспертной оценки специалистов области, на основе анализа взаимодействия входов и выходов моделируемой системы. Реальность применения данного решения можно проверить только при наличии под рукой производственного процесса.

В случае генеративного алгоритма (GAN) задача оптимизации развивается несколько сложнее – осуществление прогноза путем поиска

обратного отображения набора признаков модели в скрытое пространство не обладает какими-либо свойствами непрерывности, более того эта целевая функция является неявной. Пусть R^k соответствует k-мерному вектору случайных чисел, соответствующему некоторому множеству состояний установки, тогда:

- Генератор:

$$LSTM_G(R^k\ times\ r) \rightarrow A_{t-r}, \dots, A_t, t > r, \quad (17)$$

- Целевая функция:

$$A_{t-r}, \dots, A_t \rightarrow R^K \quad (18)$$

$$LSTM_G(R^k\ times\ (r+1)) \rightarrow A_{t-r}, \dots, A_t, A_{t+1} \quad (19)$$

Потому предлагается другой экспериментальный (эвристический) подход. Мы зададим (соответственно знаку весов w_i) максимально допустимые значения переменных из $Target$, а значения управления x оставим прежними. Тогда, получив образ нашей последовательности от генератора путем стандартной процедуры, мы увидим «скорректированные» значения целевых и переменных управления, то есть правдивую согласно GAN последовательность состояний установки.

Заключение

В данной работе была рассмотрена задача исследования рекуррентных сетей для задач прогнозирования технологических временных рядов, с акцентом на последующую возможную оптимизацию процесса на основе полученных прогнозных моделей. Было замечено, что данная проблема, не смотря на свою важность не является широко популярной у исследователей.

Было рассмотрено и предложено несколько подходов к прогнозу и оптимизации временного ряда производственной системы, в том числе: «базовая» модель, на основе простых нейросетей в комбинации с классической кластеризацией и современными реализациями ансамблей деревьев; вариации архитектур сверточных нейросетей обработки сигналов; вариации LSTM архитектур, в том числе генеративный подход.

По результатам экспериментального исследования было установлено, что лучше всего себя показывают классические рекуррентные подходы архитектуры LSTM. Было предложено строить многомерные модели во времени, предложен набор мер по построению моделей и предобработки технологических данных.

Для каждой предложенной архитектуры был разработан и предложен теоретический подход к оптимизации.

Литература

1. Multivariate Industrial Time Series with Cyber-Attack Simulation: Fault Detection Using an LSTM-based Predictive Data Model. URL: https://arxiv.org/abs/1612.06676
2. A Neural-Network-Based Model Predictive Control of Three-Phase Inverter With an Output LC Filter. URL: https://arxiv.org/abs/1902.09964
3. Detecting Cyberattacks in Industrial Control Systems Using Convolutional Neural Networks. URL: https://arxiv.org/abs/1806.08110
4. *Keras*. URL: https://keras.io/
5. *Ke G., Meng Q., Finley T., Wang T., Chen W., Ma W., Ye Q. Liu T-Y.* LightGBM: A highly efficient gradient boosting decision tree //Neural Information Processing Systems Conference. URL: https://lightgbm.readthedocs.io
6. *Hochreiter S., Schmidhuber J.* Long short-term memory //Neural Computation 1997. N 9. P. 1735–1780
7. *Borovykh A., Sander B., Cornelis. W.O.* Conditional time series forecasting with convolutional neural networks. URL: https://arxiv.org/abs/1703.04691.
8. *Goodfellow I., Pouget-Abadie J., Mirza M., Xu B., Warde-Farley D., Ozair S., Courville A., Bengio Y.* Generative Adversarial Networks (PDF). //Proceedings of the International Conference on Neural Information Processing Systems (NIPS 2014). pp. 2672–2680
9. Unsupervised Anomaly Detection with Generative Adversarial Networks to Guide Marker Discovery. URL: https://arxiv.org/abs/1703.05921
10. Библиотека *tensorflow*. URL: https://www.tensorflow.org/
11. Библиотека *scikit-learn*. URL: https://scikit-learn.org/