

Министерство науки и высшего образования Российской Федерации
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ
УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ ЯДЕРНЫЙ УНИВЕРСИТЕТ «МИФИ»
(НИЯУ МИФИ)
ИНСТИТУТ ИНТЕЛЛЕКТУАЛЬНЫХ КИБЕРНЕТИЧЕСКИХ СИСТЕМ
КАФЕДРА КИБЕРНЕТИКИ

На правах рукописи

УДК 004.89

Тарасов Д.Ю.

ПРОГРАММНАЯ РЕАЛИЗАЦИЯ АЛГОРИТМИЧЕСКОГО ОБЕСПЕЧЕНИЯ
ДЛЯ РЕШЕНИЯ ЗАДАЧИ НЕЙРО-НЕЧЕТКОГО МОДЕЛИРОВАНИЯ
СОЦИАЛЬНО-ЭКОНОМИЧЕСКИХ СИСТЕМ ПО ДАННЫМ ОТКРЫТЫХ
ИСТОЧНИКОВ.

Выпускная квалификационная работа бакалавра

Направление подготовки 01.03.02 Прикладная математика и информатика

Выпускная квалификационная
работа защищена

«__»_____20 г.

Оценка _____

Секретарь ГЭК _____

г. Москва

2020

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ
УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
Национальный исследовательский ядерный университет
«МИФИ»




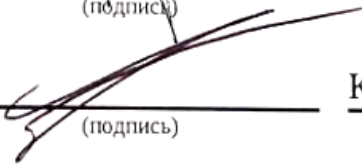
Институт
интеллектуальных кибернетических систем
Кафедра №22 «Кибернетика»

Направление подготовки 01.03.02 Прикладная математика и информатика

Пояснительная записка

к ВКР на тему:

Программная реализация алгоритмического обеспечения для решения
задачи нейро-нечеткого моделирования социально-экономических
систем по данным открытых источников.

Группа	<u>Б16-501</u>	
Студент	<u> (подпись)</u>	<u>Тарасов Д. Ю.</u> (ФИО)
Руководитель	<u> (подпись)</u>	<u>Киреев В. С.</u> (ФИО)
Научный консультант	<u> (подпись)</u>	<u> (ФИО)</u>

Москва 2020

**Национальный исследовательский ядерный университет
«МИФИ»**

Институт	Интеллектуальных кибернетических систем	Кафедра Группа	№22 «Кибернетика» Б16-501
Специальность (направление)	Прикладная математика и информатика (01.03.02)	«Утверждаю» Зав. кафедрой (подпись) «__» _____ 20__г.	Загребаяев А.М.

ЗАДАНИЕ НА ДИПЛОМНУЮ РАБОТУ

(выпускную квалификационную работу ВКР)

1. Фамилия, имя, отчество студента
_____ Тарасов Дмитрий Юрьевич _____
(ФИО) (подпись)
2. Тема работы (ВКР) «Программная реализация алгоритмического обеспечения для решения задачи нейро-нечеткого моделирования социально-экономических систем по данным открытых источников.» _____»
3. Срок сдачи студентом готовой работы: 16 июня 2020 г.
4. Место выполнения НИЯУ МИФИ
5. Руководитель работы
_____ Киреев В.С. _____
(ФИО) (уч. степень, должность) (подпись)
6. Соруководитель работы от НИЯУ МИФИ
_____ (ФИО) _____ (уч. степень, должность) _____ (подпись)
7. Консультант работы
_____ (ФИО) _____ (уч. степень, должность) _____ (подпись)

1. Аналитическая часть

Изучение и сравнительный анализ методов моделирования социально-экономических систем для задачи прогнозирования.

Изучение и анализ статистических и нейросетевых методов прогнозирования временных рядов.

2. Теоретическая часть

Разработка модифицированной нейро-нечеткой модели когнитивных карт с использованием рекуррентных нейросетей.

Оценка возможных недостатков модели.

Выбор метрик для анализа качества модели.

3. Технологическая часть

Описываются основные требования, выдвигаемые к приложению.

Описывается процесс разработки: выбор методологии разработки.

Проектирование и описание архитектуры разрабатываемой нейро-нечеткой модели.

Описывается выбор инструментальных средств для разработки.

4. Практическая часть

Планируется получить реализацию нейро-нечеткой модели социально-экономической системы и сравнить разработанную модель с существующими методами прогнозирования.

Ожидаемым результатом является алгоритмическое обеспечение нейро-нечеткого моделирования социально-экономических систем со следующими отличительными характеристиками: интерпретируемость, возможность легкого дообучения, возможность модификации модели без необходимости полного переобучения модели.

Дата выдачи задания «12» июня 2020 г.

Реферат

Пояснительная записка содержит 46 страниц (из них 5 страниц приложений), 2 таблицы, 35 рисунков, 12 формул. Количество использованных источников – 35,

Ключевые слова: когнитивные карты, нейро-нечеткие системы, нейронные сети, LSTM, рекуррентные нейросети, математическое моделирование.

Целью данной работы является создание нечеткой когнитивной карты для моделирования социально-экономических систем с использованием нейросетевых методов.

В первой главе проводится обзор существующих методов предсказания временных рядов и рассматривается модель когнитивных карт.

Во второй главе описываются использованные алгоритмы для оптимизации когнитивных карт с помощью нейронных сетей. Проводится выбор метрик для анализа качества работы разрабатываемой модели.

В третьей главе проектируется архитектура модели. Описывается процесс работы с построенной моделью.

В четвертом разделе проводится тестирование приложения. Разработанная модель сравнивается с существующими методами.

Содержание

Введение	8
1 Обзор методов моделирования социально-экономических систем	10
1.1 Анализ параметров социально-экономических систем	10
1.2 Анализ методов прогнозирования временных рядов	12
1.3 Описание алгоритма работы нечетких когнитивных карт	15
1.4 Интеграция нечетких когнитивных карт и нейросетей для предсказания времен- ных рядов	17
1.5 Выводы	18
1.6 Постановка задачи на ВКР	18
2 Разработка модели для нейро-нечеткого моделирования социально-экономических систем	20
2.1 Разработка нейро-нечеткой модели с использованием рекуррентный нейросетей	20
2.2 Построение FCM-LSTM	21
2.3 Алгоритм обучения FCM-LSTM	22
2.4 Алгоритм вычисления FCM-LSTM	23
2.5 Возможные проблемы алгоритма FCM-LSTM	23
2.6 Выбор метрик для оценки качества работы моделей	24
2.7 Выводы	25
3 Проектирование нейро-нечеткой модели	26
3.1 Разработка функциональных требований к системе	26
3.2 Архитектура системы непрерывной интеграции	26
3.3 Проектирование интерфейса модуля для нейро-нечеткого моделирования	27
3.4 Проектирование структуры когнитивной карты для моделирования количества продаж социально-экономической системы	29
3.5 Выбор инструментальных средств	32
3.6 Выводы	33

4	Реализация и экспериментальная проверка модуля для нейро-нечеткого моделирования	34
4.1	Состав и структура модуля для нейро-нечеткого моделирования	34
4.2	Анализ данных социально-экономической системы	35
4.3	Построение модели SARIMAX	36
4.4	Построение модели LSTM	38
4.5	Построение модели FCM-LSTM	41
4.6	Выводы	43
	Заключение	45
	Приложения	50
A	Приложение 1. Результаты экспериментов	50
B	Приложение 2. Отчет по тестированию разработанного модуля на Python	52
Г	Приложение 3. Описание исследуемого набора данных	53

Введение

Моделирование социально-экономической системы может быть использовано для предсказания некоторых параметров этой системы в будущем. Для анализа сложных систем, в которых необходимо принимать к сведению не только результат прогнозирования, но и причинно-следственные связи, используются нечеткие когнитивные карты [1, 2]. Однако оригинальный алгоритм работы когнитивных карт можно улучшить, если использовать для оптимизации весов карты нейронные сети. Такие системы называются нейро-нечеткими. Такие системы позволяют снизить нагрузку на экспертов и добиться лучшей точности результата работы системы, но и лучшей интерпретации по сравнению с нейросетевыми методами, которые не используют экспертные знания.

Понимание процессов, происходящих в социально-экономических системах может быть полезно как для бизнеса, так и для государства. Многие организации имеют большое количество статистических данных. С помощью этих данных можно моделировать поведение экономической системы, в которой работает данная организация. Моделирование сложных социально-экономических систем обычно включает в себя построение прогнозов на будущее. Однако не все существующие методы позволяют легко интерпретировать результаты предсказаний. Актуальность данной работы обусловлена тем, что предиктивные экспертные модели позволяют лучше понять процессы, которые происходят в исследуемой системе. Общая идея использования когнитивной карты как способа структурирования экспертных знаний о системе может быть использована так же и для других задач машинного обучения.

Для построения прогноза в работе используются рекуррентные нейронные сети. Нейронные сети — это современный и постоянно развивающийся способ моделирования. Количество исследований на эту тему ежегодно растет.

В аналитической части исследуются существующие методы прогнозирования временных рядов, сравниваются их характеристики, исследуется возможность интерпретации этих моделей. Так же рассматривается модель когнитивных карт. В конце аналитической части определяется цель данной работы и ставятся задачи.

В теоретической части представлена модель нечеткой когнитивной карты с использованием рекуррентных нейросетей, а именно LSTM. Исследуются возможные проблемы разработанной модели.

В инженерном разделе будет спроектирована архитектура модели и система непрерывной

интеграции. Рассмотрены процессы работы с построенной моделью.

В последнем разделе исследуется набор данных для обучения модели. Разработанная нейронечеткая система будет протестирована и сравнена с существующими моделями.

1. Обзор методов моделирования социально-экономических систем

В данной главе приводится анализ предметной области. Проводится сравнительный анализ методов, с помощью которых можно решать задачу прогнозирования временных рядов. Исследуется принцип работы когнитивных карт. В конце раздела формулируются цели и задачи работы.

1.1 Анализ параметров социально-экономических систем

Социально-экономические системы обычно зависят от большого количества параметров. Эти параметры могут неявно влиять друг на друга. Для описания состояния такой системы можно использовать временные ряды. Для того, чтобы изучить такую систему, нужно построить ее модель.

Задача моделирования сложных систем - заключается в определении возможных параметров системы и значений этих параметров. После того, как параметры модели были определены, можно исследовать поведение системы по полученной модели. Одним из возможных применений полученной модели может быть прогнозирование поведения модели в будущем. Также модель может помочь проверить гипотезы, которые выдвигают эксперты. Кроме того, эксперты могут исследовать причинно-следственные связи на основе моделирования. Важно отметить, что корреляция в данных не всегда означает причинно-следственную связь. Поэтому для определения причинно-следственных связей нужен эксперт.

Важной частью моделирования социально-экономической системы является прогнозирование будущих значений параметров системы. Прогноз строится на основании истории одного или нескольких временных рядов, которые соответствуют историческим данным.

Определение 1. (Временной ряд) Временным рядом называются последовательно измеренные через некоторые промежутки времени данные.

Основные явления в эконометрических временных рядах [3]:

- тренды = очищенная от случайностей основная тенденция временного ряда
- сезонности = периодические отклонения от тренда
- разладки = резкое изменение свойств наблюдаемого ряда

Сезонности тоже можно разделить на несколько типов:

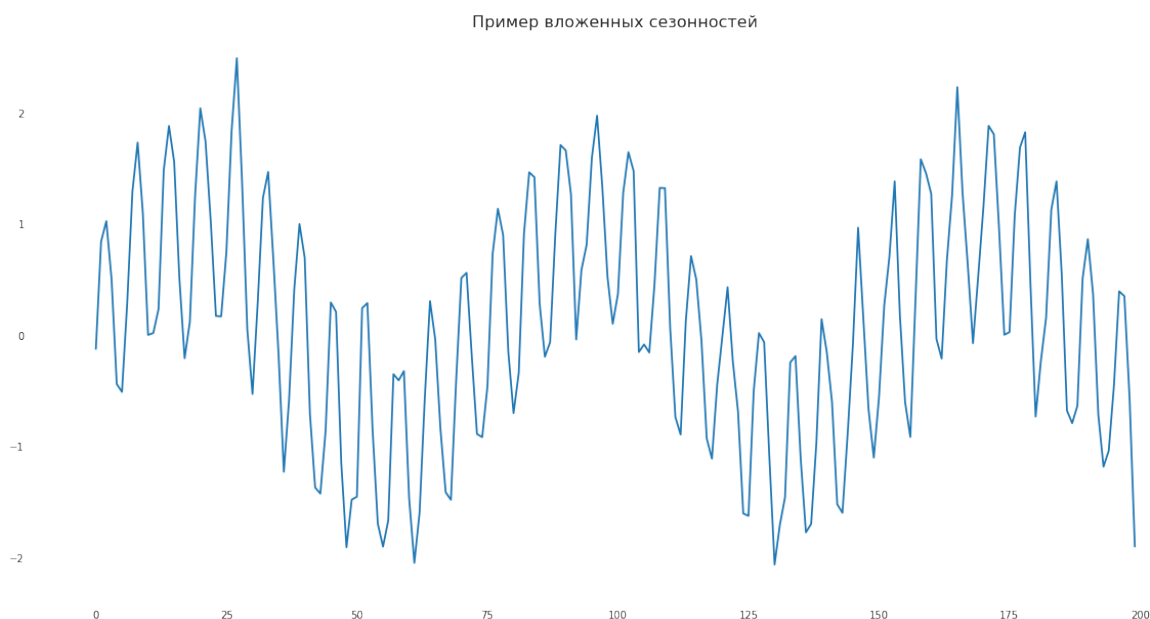


Рис 1.1 – Пример данных со вложенной сезонностью

- аддитивная сезонность
- мультипликативная сезонность

Кроме того, сезонности могут быть вложенными 1.1. Например, готовая сезонность, месячная, недельная, ежедневная.

Так же при анализе временных рядов нужно учитывать возможные выбросы. Некоторые временные ряды могут принимать аномальные значения в праздничные дни. Для таких случаев обычно модель проще обучить реагировать на отдельный входной параметр, который бы характеризовал о наличии или отсутствии праздников в определенный день.

Так как в системе может быть много параметров (и, соответственно, временных рядов), при прогнозировании модель может вычислять будущие значения параметров разными способами:

- предсказывать новые значения временного ряда на основе его прошлых значений;
- предсказывать новые значения временного ряда на основе предыдущих значений; нескольких параметров, которые могут повлиять на данный параметр
- предсказывать сразу несколько временных рядов на основании их предыдущих значений;

Эти способы перечислены в порядке усложнения модели. Сложность модели влияет на качество ее предсказаний. Однако выбор слишком сложной модели может привести к тому, что такую модель будет сложно обучить или наоборот, модель может оказаться переобученной. Переобучение ведет к тому, что модель теряет способность к обобщению. Кроме того, слож-

ные модели нужно дольше обучать. И нужно больше данных для ее качественного обучения. Сложные модели сложнее интерпретировать.

Временные ряды, которые учитываются в модели могут быть двух типов: эндогенные и экзогенные. Эндогенные — это временные ряды, которые требуется предсказать. Экзогенные ряды могут быть вычислены заранее и не зависят от состояния исследуемой системы и эндогенных временных рядов. Экзогенные переменные могут быть определены заранее и для прогнозируемого участка временного ряда. Примером экзогенных переменных может служить индикатор праздников при прогнозировании временных рядов: временной ряд этой экзогенной переменной будет принимать значение 1 в случае, если в заданный день есть праздник и 0 в других случаях.

Математическая модель. Дан набор временных рядов $A_i : a_{i,0}, a_{i,1}, \dots, a_{i,n}$ и набор экзогенных временных рядов $X_j : x_{j,0}, x_{j,1}, \dots, x_{j,n}, x_{j,n+1}, \dots, x_{j,n+h}$. Где h — горизонт прогнозирования, количество значений временного ряда, которые требуется предсказать. Необходимо с помощью значений A_i и X_j определить значения $a_{i,n+1}, \dots, a_{i,n+h}$.

Основные виды моделей для предсказаний временных рядов [3]:

- Авторегрессионные модели — значения временного ряда в данный момент линейно зависят от предыдущих значений этого же ряда
- Адаптивные модели — самонастраивающиеся модели, которые способны отражать изменяющиеся во времени условия
- Нейросетевые модели

1.2 Анализ методов прогнозирования временных рядов

Адаптивные модели хорошо работают на большом количестве временных рядов: в случаях, когда необходимо прогноз нужно получить быстро, а учитывать нужно большое количество факторов. Суть адаптивных методов заключается в том, что на каждой итерации, после того, как стали известны новые данные, параметры модели обновляются. Параметры модели вычисляются на основе исторических данных. Однако такие модели не могут описать сложные зависимости.

Простота адаптивных методов компенсируется селективными и композиционными моделями. Можно обучить несколько разных моделей и в зависимости от качества предсказаний на пошедшие моменты времени для прогнозирования можно использовать ту модель, качество предсказаний которой выше - в этом заключается идея селективных моделей. В композиционных моделях, результирующим предсказанием выступает взвешенная сумма предсказаний моделей. Вес каждой модели тоже адаптивный: вычисляются на основании ошибки данной

модели.

Обычно адаптивные модели используются для краткосрочного моделирования. Такие модели при прогнозировании временных рядов учитывают значения только одного временного ряда. То, что на один временной ряд может влиять значения других временных рядов никак не учитывается.

Преимуществом адаптивных моделей можно считать их простоту и то, что они могут подстраиваться под изменяющиеся параметры временного ряда.

Простейшим примером адаптивной модели является модель экспоненциального сглаживания:

$$\hat{y}_{t+1} = \alpha y_t + (1 - \alpha) \hat{y}_t \quad (1.2.1)$$

Где α - параметр сглаживания. Наблюдения учитываются с убывающими весами. Чем больше значение α , тем больше сглаживается результат вычислений модели.

Авторегрессионные модели могут учитывать влияние большого количества параметров временного ряда [4]. Но чем больше параметров будет иметь модель, тем больше вычислений потребуется для оценки значений этих параметров.

ARMA - модель авторегрессии скользящего среднего [5]. Используется для прогнозирования стационарных временных рядов. ARIMA - это интегрированная модель авторегрессии [6]. Позволяет объяснить тренд во временном ряде за счет интегрирования: после того, как посчитаны разности исходного временного ряда, делается предположение, что эти разности - это стационарный ряд и предсказывается поведение интегрированного ряда. ARIMAX - модификация ARIMA [7], в которой в модели регрессии могут учитываться не только предыдущие значения текущего временного ряда, но и другие экзогенные переменные. SARIMAX - ARIMAX с учетом сезонной компоненты [8]. Для учета сезонной компоненты берутся параметры авторегрессии, интегрирования и скользящего среднего с отставанием на определенный период — показатель сезонности.

Рассмотрим модель SARIMAX подробнее. Данная модель имеет 7 гиперпараметров:

- p - степень авторегрессионной части модели
- d - степень интегрирования
- q - степень модели скользящего среднего
- P - степень авторегрессионной части модели для сезонной компоненты регрессии
- D - степень интегрирования для сезонной компоненты регрессии
- Q - степень модели скользящего среднего для сезонной компоненты регрессии

- S - параметр, определяющий лаг сезонности

Также отдельно стоит учитывать экзогенные переменные. Каждая переменная добавляет еще 1 коэффициент в уравнение регрессии модели.

Определить перечисленные параметры можно перебором различных комбинаций этих параметров, или с помощью визуального анализа.

Параметр d предназначен для того, чтобы избавиться от тренда во временном ряде. После дифференцирования порядка d график временного ряда должен выглядеть как шум.

Для определения параметра p , следует изучать график частичных автокорреляций. Если на есть статистически значимые отклонения, то индекс такого отклонения должен соответствовать значению данного параметра.

Для определения параметра q необходимо найти статистически значимые отклонения для автокорреляции ряда.

Важно отметить, что параметры p должен оцениваться для дифференцированного ряда.

Тогда формула для модели *ARIMAX*:

$$\Delta^d X_t = c + \sum_{i=1}^p a_i \Delta^d X_{t-i} + \sum_{j=1}^q b_j \varepsilon_{t-j} + \varepsilon_t \quad (1.2.2)$$

Где Δ^d — оператор разности временного ряда. a_j — авторегрессионные коэффициенты. b_j — коэффициенты скользящего среднего. c — константа. Для модели с учетом сезонностей добавляются такие же слагаемые, но в них оператор разности заменен на сезонный оператор разности. Разности начинают вычисляться начиная не для текущего значения временного ряда, а начиная с шага $t - S$. Где t — номер текущей итерации. S — показатель лага сезонности.

В нейросетевых моделях тоже можно учесть влияние одного параметра системы на другие. Суть данных методов похожа на предыдущие: сначала настраиваются начальные значения параметров модели, а потом эти параметры обновляются, если добавляются новые данные. Основным отличием можно считать то, что в предсказании одного временного ряда можно учитывать значения других временных рядов в разные промежутки времени.

Нейросетевые методы можно разделить на 2 вида:

- сети прямого распространения [9]
- рекуррентные [10]

Сети прямого распространения не подразумевают какого-либо состояния. Поэтому передать информацию о предыдущем шаге итерации можно только явно: предыдущие значения временного ряда должны передаваться на вход модели. К сожалению, при таком подходе, количество параметров сети возрастает очень быстро при увеличении количества предыдущих

значений временного ряда. Это сильно усложняет модель, она легко переобучается. И не может адекватно предсказать новые значения для временного ряда, потому что теряет способность к обобщению.

Рекуррентные сети наравне с параметрами весов имеют внутреннее состояние, которое обновляется на каждой итерации. Это означает, что предсказание будущих значений рядов зависит от "контекста", то есть от предыдущих вычислений. Данные сети также могут учитывать влияние других временных рядов на текущий даже с учетом нескольких предыдущих значений.

Но при описании влияния нескольких временных рядов в нейросетевых моделях нельзя описать, на какой именно ряд влияет другой ряд. Такие знания могут быть у экспертов. И потенциально, эти знания могли бы упростить модель и уменьшить количество параметров в ней.

1.3 Описание алгоритма работы нечетких когнитивных карт

Когнитивная карта — схема причинно-следственных связей в системе. Когнитивная карта представляет из себя ориентированный граф. Вершины этого графа — концепты, а ребра — причинно-следственные связи между соответствующими концептами. Когнитивные карты строятся для того, чтобы понять и проанализировать структуру сложной системы. Каждое ребро когнитивной карты имеет свой вес, который характеризует степень влияния одного концепта на другой.

Обычно при прогнозировании с использованием нечетких когнитивных карт существует один целевой концепт, значение которого необходимо спрогнозировать.

Стандартная формула для пересчета значений концептов нечетких когнитивных карт:

$$A_j(t+1) = p(A_j(t) + \sum_{i=1, i \neq j}^n w_{ij} A_i(t)) \quad (1.3.1)$$

Где $A_i(t)$ - значение концепта A_i на шаге t . $p(x)$ - функция активации:

$$p(x) = \frac{1}{1 + e^{-mx}} \quad (1.3.2)$$

Параметр m определяет, на сколько сигмоида будет похожа на пороговую функцию.

В зависимости от задачи и данных, с которыми работает эксперт могут использоваться и другие функции активации.

После того, как эксперт построил карту, он может приступить к моделированию поведения системы и изучением причинно-следственной связи. Таким образом, эксперт может посмотреть, как изменение одного концепта может повлиять на поведение системы в целом.

Если значения концептов карты расходятся, эксперту нужно или попробовать изменить функцию активации или изменить значения весов концептов. Возможно, карта несбалансированна из-за того, что в ней отсутствует еще один компонент, который вносит значительный вклад в поведение системы.

Пересчет значений концептов проходит итеративно. Имея исторические данные эксперт может подобрать значения весов таким образом, чтобы смоделированное поведение системы как можно меньше отличалось от экспериментальных данных. В случае классических когнитивных карт эксперту нужно это делать вручную.

После того, как эксперт оптимизировал значения весов, можно приступить к изучению причинно-следственных связей.

Когнитивные карты можно классифицировать:

- Нечеткие
- Нейро-нечеткие
- Расширенные-нечеткие
- Четкие
- Нейро-четкие
- Расширенные четкие

Нечеткие когнитивные карты используются для тех задач, в которых нельзя явно оценить значения концептов. Для того, чтобы получить численное представление нечеткого термина, используются функции фаззификации. Обратная процедура (преобразование нечеткого числа в лингвистические переменные) называется дефаззификацией.

Нейро-нечеткие когнитивные карты позволяют настроить веса карты с помощью искусственных нейронных сетей. Это позволяет ускорить процесс построения карты.

В расширенных нечетких когнитивных картах появляется возможность задавать отложенную активацию весов и более гибко формировать связи между концептами с помощью правил "ЕСЛИ-ТО".

Четкие когнитивные карты отличаются от нечетких тем, что вместо нечетких чисел, в них используются четкие. Это позволяет делать получить более точные модели. Четкие карты являются частным случаем нечетких [11]: отличаются только отсутствием процедур фаззификации и дефаззификации.

Таким образом, решение использовать ли четкие числа или нечеткие определяется прежде всего предметной областью и данными, которые необходимо исследовать. Предпочтительным является использование четких чисел. Но можно строить модели, в которых бы использовались

как четкие, так и нечеткие числа, так как четкие когнитивные карты — это частный случай нечетких.

1.4 Интеграция нечетких когнитивных карт и нейросетей для предсказания временных рядов

Нейро-нечеткая система - это система, которая включает в себя методы нейронных сетей и методы нечеткой логики.

Поведение объекта в сложных системах прогнозировать непросто. Часто необходимо знать не только результат прогноза, но и причинно-следственные связи, которыми был обусловлен данный прогноз [1, 12]. Для выявления таких причинно-следственных связей используются нечеткие когнитивные карты. Следует отметить, что корреляция между временными рядами позволяет качественно оценить влияние разных компонент системы на другие компоненты. Однако, оригинальный алгоритм для пересчета значений концептов не может описать сложные зависимости между отдельными компонентами. Поэтому для того, чтобы увеличить мощность множества зависимостей, которые может описать модель системы, можно использовать нейросетевые методы. В таком случае когнитивная карта все еще хранит данные о причинно-следственных связях.

В нейросетевых подходах установить причинно-следственные связи сложнее. Другие системы, например, деревья решений, наоборот, очень легко интерпретируются, легко обучаются, но имеют более слабую способность к обобщению.

Идея нечетких когнитивных карт, предложенных Коско [2] имеет следующие преимущества:

- Для проведения вычислений требуются небольшие вычислительные мощности.
- Простота в использовании.
- Можно легко добавлять и удалять концепты.
- Можно объединять готовые модели.

Однако есть и недостатки:

- Значение концепта зависит только от значений связанных концептов карты только на предыдущей итерации. Невозможно определить связанные во времени концепты больше, чем на одну итерацию.
- Для создания карты требуется большая работа по подбору весов.

Эти недостатки можно считать существенными. С этими недостатками можно бороться, если использовать модификации нечетких когнитивных карт, например, TTR NFCM (Thee Term

Relation Neural Fuzzy Cognitive Map) [13].

В отличие от классических нечетких карт TTR NFCM позволяет определять нелинейные функции для весов и при перерасчете значений концептов может учитывать то, как этот концепт менялся во времени. Возможность использовать нелинейные веса достигается с помощью многослойных прецептронов. А влияние предыдущих значений концептов определяется размером временного окна, за период которого проводится перерасчет значений концептов.

1.5 Выводы

1. Адаптивные модели хорошо работают на большом количестве временных рядов, но не учитывают влияние других рядов на текущий. Они хорошо предсказывают простые зависимости. И имеют возможность адаптироваться к новому уровню процесса.
2. Модификации регрессионных моделей могут позволить описать значительную часть информации о временном ряде. Они более интерпретируемы, чем нейросетевые методы, более просты.
3. Сети прямого распространения плохо подходят для предсказания временных рядов.
4. Рекуррентные сети хорошо подходят для предсказания временных рядов, но с помощью них сложно представить знания о зависимостях между временными рядами.
5. Четкие когнитивные карты являются частным случаем нечетких [11]. Допустимо использовать как четкие, так и нечеткие числа в одной когнитивной карте. Тип параметров карты (четкий или нечеткий) определяется прежде всего из данных, с которыми работает эксперт. Нечеткость — это способ формализовать лингвистические переменные.

1.6 Постановка задачи на ВКР

В рассмотренных методах не хватает возможности структурировать экспертные знания. Для решения этой проблемы могут быть использованы когнитивные карты. Кроме этого, когнитивные карт разных экспертов можно объединять. Обычно объединенная карта представляет данные более объективно, потому что возможные ошибки одного эксперта могут быть компенсированы знаниями другого эксперта.

Целью данной работы является создание нечеткой когнитивной карты для моделирования социально-экономических систем с использованием нейросетевых моделей по данным открытых источников. Необходимо протестировать разработанную модель и сравнить с существующими решениями.

Задачи данной работы:

- Разработать алгоритм для интеграции нейросетевых моделей с когнитивными картами.

- Спроектировать и реализовать нейро-нечеткую модель для когнитивного картирования.
- Предобработать и проанализировать тестовый набор данных, на котором будет обучена исследуемая модель.
- Протестировать полученную модель на данных открытых источников.
- Сравнить разработанный алгоритм нейро-нечеткого моделирования с существующими решениями.

2. Разработка модели для нейро-нечеткого моделирования социально-экономических систем

В данной главе описывается модифицированная модель когнитивных карт с использованием рекуррентных нейросетей. Производится выбор метрик оценки качества работы системы.

2.1 Разработка нейро-нечеткой модели с использованием рекуррентных нейросетей

Можно усовершенствовать модель TTR LSTM, рассмотренную в первой главе, если вместо сетей прямого распространения использовать рекуррентные сети, например, LSTM [14]. Такую модель будем называть FCM-LSTM 2.1. Выбор рекуррентных сетей обусловлен тем, что такие сети позволяют естественно сохранять информацию об истории концептов в своем скрытом состоянии. В то время как для того, чтобы учитывать значения концептов на предыдущих итерациях в сетях прямого распространения, каждый предыдущий шаг должен быть описан как отдельный параметр. Это сильно увеличивает количество параметров модели и усложняет ее обучение.

Сравнение FCM-LSTM с полносвязными сетями: У FCM-LSTM меньше параметров по сравнению с сетью прямого распространения, которая бы учитывала больше количество предыдущих значений во временном ряде. FCM-LSTM дольше обучается, но имеет значительно лучшую способность к предсказанию на длинных временных рядах. Модель будет дольше обучаться, потому что нельзя распараллелить процесс обучения LSTM сетей, так как в сети учитывается, что внутреннее состояние меняется последовательно. FCM-LSTM имеет лучшую интерпретируемость, так как содержит экспертную информацию о взаимосвязи концептов. Но для построения FCM-LSTM требуется эксперт, а как следствие, ручная работа.

Сравнение FCM-LSTM с TTR NFCM: FCM-LSTM дольше обучается, но на сложных наборах данных может иметь более точные прогнозы.

Сравнение FCM-LSTM с LSTM: LSTM сеть, которая бы соответствовала FCM-LSTM, будет иметь больше параметров в случае, если граф FCM-LSTM не слишком много связей. FCM-LSTM лучше интерпретируема. Скорость обучения, при условии оптимизаций в FCM-LSTM, с помощью которых обучение LSTM моделей в каждой вершине бы проходило параллельно, будет примерно одинаковой. Предсказательная способность должна быть лучше у FCM-LSTM. Потому что такая модель более проста и структурирована. LSTM модель может оказаться излишне

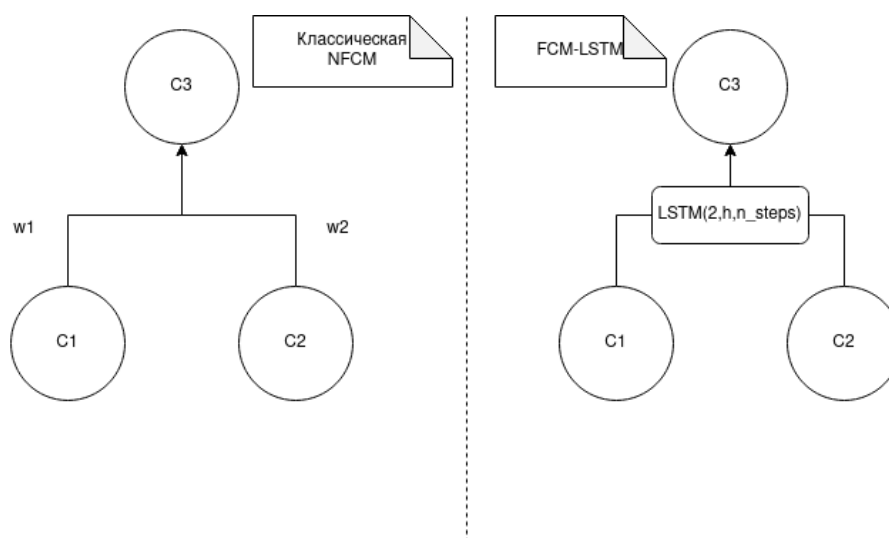


Рис 2.1 – Сравнение архитектуры классической когнитивной карты и FCM-LSTM.

усложненной, так как в ней будет большое количество параметров. Но для построения FCM-LSTM требуется эксперт, а как следствие, ручная работа. Для построения FCM-LSTM требуется эксперт.

Кроме того, так как карта представляет из себя ансамбль более маленьких моделей, в случае ошибки в исходных данных в одной части модели, можно переобучать не всю карту целиком, а только ту часть в которой была ошибка. Так как процесс обучения может занимать долгое время, эта оптимизация может существенно ускорить процесс. Такой же прием можно применить и для случая, когда нужно дообучить карту на новом наборе данных: можно дообучать карту частями.

2.2 Построение FCM-LSTM

Для того, чтобы создать модель, нужно определить параметры системы, концепты. Каждый концепт будет соответствовать вершине графа карты. После того, как концепты определены, нужно описать взаимодействие между концептами. Одним из вариантов может быть полно-связный граф, когда каждый концепт имеет влияние на каждый другой концепт. Теоретически, при увеличении количества связей в модели, количество возможных зависимостей, которые она может описать увеличивается. Однако, усложняется и сама модель так, что ее становится сложно обучить. Слишком сложные модели легко переобучаются на тестовых данных. Задача эксперта как раз заключается в том, что он должен описать только необходимые и достаточные взаимосвязи.

Так же, как и большое количество взаимосвязей, недостаточное количество связей или неверные связи, могут ухудшить качество модели. Модель может оказаться недостаточно мощной, чтобы описать исследуемый процесс. Поэтому важно найти именно существенные связи.

После того, как карта построена, необходимо "заморозить карту". Это означает, что новые связи в карте больше появляться не будут. Это необходимо для того, чтобы рассчитать параметры моделей, которые лежат в каждой вершине графа карты: размерность данных на входе, размерность внутреннего состояния, размерность данных на выходе модели. Перед заморозкой для каждого концепта должна быть определена модель, которая будет использована при моделировании данного концепта.

Для того, чтобы добавить новый концепт после заморозки карты, потребуется переобучить карту. Как описано в предыдущем абзаце, нет необходимости переобучать всю карту целиком. Нужно переобучить только концепт, к которому добавилась новая связь.

После "заморозки", должны быть инициализированы модели в каждой вершине карты. Эксперт имеет возможность задать создать любую модель заданной размерности для каждого концепта отдельно. Выбор модели должен быть основан на характере данных. Можно использовать как авторегрессионные модели, так и нелинейные, в том числе и нейросетевые. Также можно использовать различные комбинации этих подходов. В данной работе будет использоваться LSTM.

2.3 Алгоритм обучения FCM-LSTM

Каждый концепт имеет исторические данные, которые используются для обучения модели концепта.

Процесс обучения можно разбить на несколько шагов:

- Подготовка данных.
- Тренировка сети.
- Валидация обученной сети.

Подготовка данных. На этом этапе происходит разбиение исторических данных на части для обучения и для валидации. Это необходимо, чтобы не допустить переобучения сети: когда на выборке для обучения показатели качества сети отличные, но на валидационной выборке плохие. В подготовку данных также нужно включить преобразование данных в последовательности необходимой размерности. На вход LSTM сеть принимает матрицу размерности $seq_len \times batch \times input_size$. seq_len - длина последовательности, непрерывный кусок данных. $batch$ - размер пакета на каждой итерации, для которого будет вычисляться градиент. $input_size$ - количество связей, которые входят в обучаемый концепт.

Тренировка сети. На данном шаге происходит оптимизация весов. Для модель LSTM, соответствующая каждому концепту будет обучена на данных, которые для данного концепта определил эксперт. сети каждого концепта. В качестве функции потерь будет использоваться

MSE. В качестве оптимизатора будет использован ADAM [15].

Валидация сети подразумевает проверку качества работы обученной сети на данных, которых не было в выборке для обучения.

2.4 Алгоритм вычисления FCM-LSTM

Алгоритм вычисления LSTM-NFCM не сильно отличается от алгоритма вычисления классической нечеткой когнитивной карты.

Предсказание одной итерации в LSTM-NFCM ничем не будет отличаться от предсказаний тех моделей, которые заложены в вершины графа. Можно провести несколько итераций вычисления карты, с учетом вычисленных данных. На следующем шаге модель будет обрабатывать данные, которые сгенерировала на предыдущем шаге. Таким образом можно получить более долгосрочное предсказание. Такой метод называется динамическое предсказание.

Кроме того, в такой модели должны проявиться влияния обратных связей, которые могут быть заложены в архитектуре карты, если карта содержит циклы. Таким образом, в LSTM-NFCM есть 2 уровня обратных связей: на уровне модели концепта (LSTM) и на уровне структуры карты, если карта содержит обратные связи. Обратные связи на уровне структуры карты интересны тем, что через них можно было бы выразить циклически меняющееся состояние системы.

Другой способ получить более долгосрочные предсказания - обучить LSTM предсказывать на большее количество итераций вперед. Такой подход требует больше памяти.

2.5 Возможные проблемы алгоритма FCM-LSTM

Так как в предложенной нейро-нечеткой модели значительно усложнилась структура весов, то анализ весов такой карты тоже заметно усложнился. Вес влияния концептов в такой системе не статичен, а зависит от истории изменения концептов, то есть в разные моменты времени степень влияния одного концепта на другой может быть разной.

Для обучения нейросети требуется, чтобы данные для обучения были репрезентативными. То есть описывали все возможные значения, которые может принимать система. Если в предсказании появляются данные, которых не было в обучающей выборке, модель нужно переобучать. Можно предположить, что эту проблему можно решить, если использовать версию LSTM без функции активации или, возможно, с другой функцией активации, например, ReLU. ReLU просто вычислять. И в отличие от сигмоиды, она не ограничена сверху. Также можно использовать другую модель. При краткосрочном прогнозировании данный недостаток не является существенным. Кроме того, такой проблемы можно избежать, если избавиться от тренда с помощью интегрированной регрессионной модели так, чтобы на вход нейросети поступал стационарный ряд.

Если внутреннее состояние будет иметь большую размерность, модель будет потреблять много памяти. Возможно, модель можно декомпозировать на меньшие задачи и обойтись несколькими концептами с меньшей размерностью внутреннего состояния. Но данный метод не относится к методам, в которых нужно бездумно пользоваться готовой моделью, а нацелен на то, чтобы наоборот получить больше понимания об изучаемом процессе.

При достаточно большом количестве временных рядов, становится сложно экспертно определить взаимодействие между концептами. Для задач с большим количеством рядов для исследования, без модификаций, данный метод не будет эффективен. Так как потребует большое количество вычислительных ресурсов. Для таких задач можно использовать, например, градиентный бустинг [16].

2.6 Выбор метрик для оценки качества работы моделей

Кросс-валидация - это метод оценки модели на данных, не участвовавших в обучении модели. Особенностью кросс-валидации для задачи прогнозирования временных рядов заключается в том, перемешивать данные нельзя, так как порядок данных во временном ряде - это его неотъемлемая составляющая.

В качестве критерия качества результата работы системы для предсказаний для одного временного ряда, можно взять среднеквадратическую масштабированную ошибку (RMSSE — Root Mean Squared Scaled Error) [17, 18]:

$$RMSSE = \sqrt{\frac{\frac{1}{h} \sum_{t=n+1}^{n+h} (Y_t - \hat{Y}_t)^2}{\frac{1}{n-1} \sum_{t=2}^n (Y_t - Y_{t-1})^2}} \quad (2.6.1)$$

Данная метрика обладает рядом преимуществ:

- Ошибка не зависит от масштаба и может быть эффективно использована для предсказаний временных рядов разных масштабов.
- Функция может быть вычислена безопасно. Деление на ноль может произойти только если временной ряд состоял из одного повторяющегося числа.
- Одинаково штрафуются и положительные, и отрицательные отклонения, метрика симметрична.

Для того, чтобы оценить получить ошибку по предсказанию для всех временных рядов, можно использовать взвешенную среднеквадратическую масштабированную ошибку. Она вычисляется по формуле:

$$WRMSSE = \sum_{i=1}^k w_i * RMSSE_i \quad (2.6.2)$$

Так как для некоторых значение $RMSE$ зависит от длины ряда, на котором обучается модель, то для сравнения между собой моделей, обученных на выборках разной длины, будет использоваться сумма квадратов ошибок:

$$MSE = \frac{1}{n} \sum_{i=1}^n (Y_i - \hat{Y}_i)^2 \quad (2.6.3)$$

Кроме того, в некоторых методах обучения в процессе оптимизации весов будет использоваться именно эта метрика.

Для определения оптимальных гиперпараметров ARIMAX модели будет использоваться критерий Акаике [19, 20, 21].

Это критерий используется для сравнения одинаковых моделей с разными гиперпараметрами. Чем меньше значение этого критерия, тем оптимальнее набор гиперпараметров. Для случая, когда модели обучены на выборках одинаковой длины, этот критерий можно вычислить по формуле:

$$AIC = 2k - 2\ln(L) \quad (2.6.4)$$

Где L - значение функции правдоподобия, а k - количество параметров:

$$k = p + q + P + Q + 1 \quad (2.6.5)$$

Чем больше значения гиперпараметров (p, q, P, Q) для ARIMAX модели, тем более лучшие результаты будут получены на тестовой выборке. Поэтому эти гиперпараметры нельзя выбирать из принципа максимального правдоподобия.

2.7 Выводы

В данной главе была рассмотрена модель когнитивных карт. Когнитивные карты являются перспективным методом для структурирования экспертных знаний. Была разработана модель когнитивных карт с использованием нейросетей (рекуррентных и прямого распространения). Были выбраны методы оценки качества работы разрабатываемой системы и методы оптимизации параметров разработанной модели.

3. Проектирование нейро-нечеткой модели

В данной главе разрабатываются функциональные требования к разрабатываемой системе. Проектируется интерфейс модуля. Проектируется структура когнитивной карты. Проводится выбор инструментальных средств разработки.

3.1 Разработка функциональных требований к системе

Так как разрабатываемый модуль предназначен для решения задачи, которая требует интенсивные вычисления, система должна быть реализована с возможностью оптимизации данных вычислений. Распараллеливать вычисления можно с помощью видеокарт. Это можно сделать с использованием фреймворка *pytorch*.

Для проверки качества кода и его работоспособности, должны быть написаны тесты. Авто-тесты помогают быстрее находить ошибки и регрессии в функциональности программы.

Эксперт должен иметь возможность задать произвольные концепты, описать взаимосвязь между концептами явно или неявно, с помощью обученной карты. Для того, чтобы обучить карту, эксперт должен иметь возможность загрузить в нее исторические данные.

Эксперт должен иметь возможность построить графики зависимостей концептов как по историческим данным, так и по предсказанным данным.

Таким образом, функциональные требования:

- Возможность распараллеливания вычислений, с использованием фреймворка *pytorch*.
- Система непрерывной интеграции и автоматизированное тестирование.
- Поддержка заданных экспертом зависимостей между концептами.
- Поддержка зависимостей между концептами, вычисляемых с помощью LSTM.
- Модуль должен быть расширяемым. Нужна возможность определить пользовательскую функцию для постобработки данных, сгенерированных картой.
- Возможность динамических предсказаний. Динамические предсказания — это предсказание, построенное на данных, которые были сгенерированы моделью на предыдущих шагах цикла.
- Возможность задать экзогенные параметры для карты.

3.2 Архитектура системы непрерывной интеграции

При отправке новых изменений в исходных текстах модуля, удаленный репозиторий запускает обработчик тестов. В этом обработчике запускаются команды, которые проверяют ра-

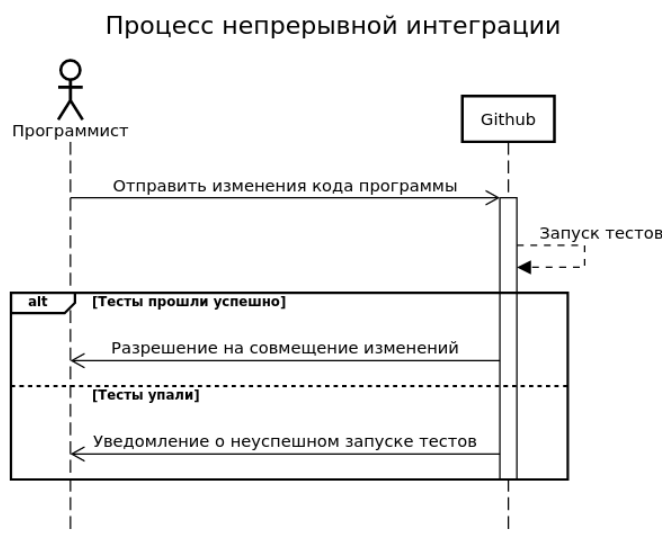


Рис 3.1 – Диаграмма последовательности процесса непрерывной интеграции

ботоспособность и качество кода в проекте. В случае ошибок, pull-request блокируется, новые изменения не могут попасть в ветку *master*. На почту разработчика приходит уведомление, что тесты не прошли. Процесс непрерывной интеграции представлен на диаграмме последовательности 3.1.

Когда тестов становится слишком много, для того, чтобы было проще ориентироваться, где именно произошла ошибка, можно пользоваться отдельными приложениями для построения отчетов о тестировании. К таким отчетам можно прикреплять дополнительную отладочную информацию, какую пожелает разработчик тестов. Однако на данном этапе тестов не будет очень много и это пока что не целесообразно.

3.3 Проектирование интерфейса модуля для нейро-нечеткого моделирования

Для проектирования используется методология ООАП (объектно-ориентированный анализ и проектирование) [22].

Работа с модулем должна состоять из следующих шагов:

- Эксперт описывает карту и загружает в нее исходные данные.
- Карта обучается.
- Выполняется предсказание.
- Эксперт оценивает результат работы модели.
- На основании результатов карта модифицируется, дополняется.

Описание концептов. При описании карты разработчик может задать любое количество концептов и связать их любым образом. Для моделирования будущих значений концепта бу-

дет использоваться LSTM. Для того, чтобы обучить эту модель предсказывать будущие значения данного концепта нужны исторические данные. Это необходимый параметр для определения нового концепта. Также должна быть возможность задать произвольную модель, которую эксперт может задать самостоятельно.

Обучение. Теоретически, этот этап может быть распараллелен за счет того, что исторические данные каждого концепта не меняются и обучение каждого отдельного концепта происходит независимо. Все модели, поддерживаемые картой должны иметь общий интерфейс для обучения и вычисления, чтобы не нарушать инкапсуляцию. Метод обучения заданной экспертом модели должен быть скрыт внутри модели.

Вычисление карты. Во время вычисления карты должны быть определены приоритеты вычислений тех или иных концептов. В зависимости от типа модели должна быть возможность получить график функции потерь для модели, чтобы оценить качество предсказаний.

Модификация карты. При добавлении новых концептов, нужно заново обучить связанные концепты. При удалении концептов, потребуется переобучение зависимых от удаленного концепта вершин.

Моделирование с помощью когнитивной карты. Суть моделирования заключается не только в том, чтобы подобрать оптимальные параметры для выборке для обучения и провалидировать обученную модель. Эксперт может выдвинуть гипотезу, подстроить параметры модели и исследовать, как будет вести себя система, анализировать остатки, которые дает модель. Добавлять или удалять концепты с целью найти объяснение тем или иным процессам. Так же эксперт может добавлять в модель экзогенные параметры, которые могут спрогнозировать поведение системы. Экзогенные параметры должны быть определены экспертом не только на выборке для обучения, но и на тестовой выборке. Предполагается, что эти параметры можно вычислить заранее. В итоге эксперт должен получить граф, который описывает исследуемый процесс. Этот граф можно интерпретировать в соответствии с названиями каждой вершины, концепта. Каждый концепт имеет свою историю значений. И может зависеть или влиять на другие концепты. Таким образом, он отражает знания эксперта об исследуемой системе. И с помощью функций, преобразований, которые эксперт задал на этом графе, может быть получено предсказание определенных значений концептов.

Повторное обучение. Так же как и для модификации карты. Не обязательно переобучать всю карту целиком. Если в этом есть необходимость, можно переобучить только часть карты.

Диаграмма классов спроектированного модуля представлена на рисунке 3.2. Классы моделей, которые могут быть использованы в качестве модели для вычисления концептов, должны имплементировать 3 метода:

- `fit()` — производится настройка модели на основании исторических данных концепта;
- `freeze_check()` — перед заморозкой карты некоторым моделям может потребоваться проверить валидность данных, сохраненных в концептах.
- `forward()` — процесс предсказаний или вычислений будущих значений концептов;

Эксперт может сам задать произвольную функцию для вычисления будущих значений концептов, если определит свой класс модели.

Класс FCM-LSTM содержит 4 атрибута:

- `graph` — объект графа библиотеки *networkx*;
- `n_steps_in` — количество временных точек, на основании которых будет делаться предсказание;
- `n_steps_out` — количество временных точек, которые требуется предсказать (горизонт предсказаний);
- `learning_rate` — коэффициент, определяющий скорость размер шага оптимизации;

и 8 методов:

- `add_new_concept` — добавить новый концепт к карте;
- `connect_concepts` — создает связь между концептами;
- `fit_on_history` — производит обучение моделей каждого концепта на основе исторических данных;
- `fcm_evalute` — производит предсказания на основе исторических данных;
- `plot_concept_history` — выводит график значений исторических данных построенной карты. Результат вывода метода A.2;
- `plot_map` — метод для отображения построенной карты 3.3;
- `_get_pred_concepts` — приватный метод для получения списка концептов, которые оказывают влияние на текущий концепт;
- `_split_sequences` — приватный метод для создания данных для обучения LSTM: разделяет непрерывный временной ряд на части заданного размера с заданным шагом (по умолчанию 1);

3.4 Проектирование структуры когнитивной карты для моделирования количества продаж социально-экономической системы

Исследуемые данные представляют из себя несколько временных рядов за 5 лет по количеству продаж в 3 разных магазинах по 3 разным категориям:

- Еда

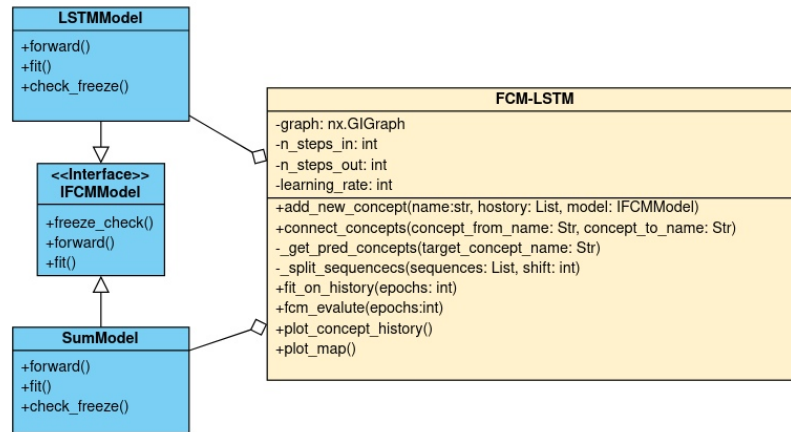


Рис 3.2 – Диаграмма классов проектируемого модуля

- Товары для дома
- Товары для хобби

Процесс покупки продуктов — это социально-экономический процесс. Количество продаж может зависеть от локальных или национальных праздников. Продажи по конкретным товарам могут возрастать, если на эти товары есть скидки или акции. Но так же стоит помнить, что в случае, если товар не завезли, продажи такого товара будут нулевыми. Но это не значит, что этот товар никому не нужен.

В данной работе будет построена упрощенная модель карты для предсказания количества продаж. Но влияние экзогенных параметров будет исследовано на других моделях.

Общее количества продаж складывается из сумм продаж по в каждом магазине:

$$STORE_SALES_i = \sum_{j=1}^M CATEGORY_SALES_{i,j} \quad (3.4.1)$$

А продажи в каждом магазине складываются из продаж товаров каждой категории:

$$STORE_SALES_i = \sum_{j=1}^M CATEGORY_SALES_{i,j} \quad (3.4.2)$$

Тогда когнитивная карта будет содержать следующие вершины:

- *total_sales* — общее количество продаж
- *store_id_TX_1* — количество продаж в первом магазине
- *store_id_TX_1_cat_id_FOODS* — количество продаж в первом магазине в категории ”еда”
- *store_id_TX_1_cat_id_HOBBIES* — количество продаж в первом магазине в категории ”товары для хобби”

- $store_id_TX_1_cat_id_HOUSEHOLD$ — количество продаж в первом магазине в категории "товары для дома"
- $store_id_TX_2$ — количество продаж во втором магазине
- $store_id_TX_2_cat_id_FOODS$ — количество продаж во втором магазине в категории "еда"
- $store_id_TX_2_cat_id_HOBBIES$ — количество продаж во втором магазине в категории "товары для хобби"
- $store_id_TX_2_cat_id_HOUSEHOLD$ — количество продаж во втором магазине в категории "товары для дома"
- $store_id_TX_3$ — количество продаж в третьем магазине
- $store_id_TX_3_cat_id_FOODS$ — количество продаж в третьем магазине в категории "еда"
- $store_id_TX_3_cat_id_HOBBIES$ — количество продаж в третьем магазине в категории "товары для хобби"
- $store_id_TX_3_cat_id_HOUSEHOLD$ — количество продаж в третьем магазине в категории "товары для дома"

Для моделирования продаж по каждой категории в конкретном магазине будет использоваться LSTM. Таким образом имея предсказания продаж для товаров каждой категории в каждом магазине мы можем явно вычислить количество продаж в магазине целиком и в сумме для трех магазинов. Для этого необходимо просуммировать результат предсказаний каждой категории в каждом магазине — $CATEGORY_SALES_{i,j}$. Для этого при построении карты для значений концептов

- $total_sales$
- $store_id_TX_1$
- $store_id_TX_2$
- $store_id_TX_3$

будет использоваться простая модель для суммирования. Особенность ее заключается в том, что модель LSTM в вершинах FCM-LSTM работает с предыдущими историческими значениями концептов. А модель по суммированию должна работать с уже предсказанными результатами модели LSTM.

Граф спроектированной карты представлен на рисунке 3.3.

Все параметры в данной карте представляют из себя четкие числа. Для исследуемого процесса нет возможности найти данные, которые можно было бы использовать как нечеткие па-

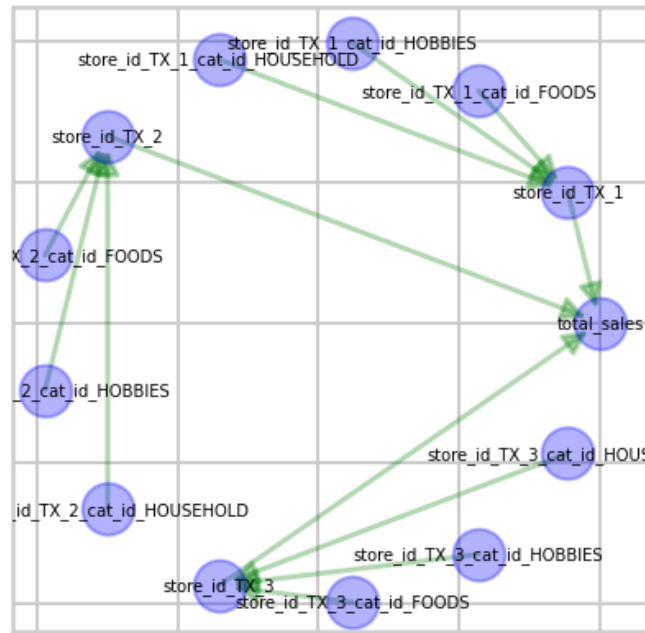


Рис 3.3 – Граф когнитивной карты для моделирования общего количества продаж с разбивкой по магазинам и категории товаров

параметры. Однако стоит заметить, что использование нечетких параметров в разработанной системе все еще возможно: четкие карты — это частный случай нечетких.

3.5 Выбор инструментальных средств

В качестве языка программирования для проведения экспериментов и анализа данных будет использоваться python. Это язык с динамической типизацией, но зато с большим количеством хорошо протестированных и отлаженных фреймворков и библиотек для работы с данными (*numpy* [23], *pandas* [24]), статистическим анализом (*statmodels* [25]) и фреймворками для построения нейросетей (*pytorch* [26], *tensorflow* [27]).

Для распараллеливания работы с данными и использовании аппаратных ускорений для умножений матриц для реализации модуля подсистемы обработки данных и подсистемы для вычисления нечеткой карты будет использоваться *pytorch*. Это фреймворк для работы с данными и созданию нейросетей.

Конкурентом *pytorch* является *tensorflow*, но за счет того, что в *pytorch* не статический граф вычислений, а динамический, *pytorch* предоставляет больше свободы, но и больше мест, где можно ошибиться.

Недостатком python можно считать динамическую типизацию. Динамическая типизация не позволяет найти многие ошибки на стадии компиляции. Многие ошибки можно обнаружить только во время работы программы. Для того, чтобы бороться с этим недостатком, вместе с кодом нужно писать автоматические тесты — они позволяют быстро проверить работоспо-

способность программы.

Для написания автоматических тестов был выбран фреймворк *pytest* [28]. В этом фреймворке вводится понятие фикстур. Это помогает разделить тело теста (логику тести) и подготовку данных для тестирования. Фикстуры позволяют сохранять тело теста более чистым. Это позволяет быстро определить, в чем именно заключается ошибка. Кроме того, благодаря возможности параметризации фикстур и теста, можно запустить один и тот же тест с разными наборами данных.

Для работы с графами был выбран фреймворк *networkx* [29]. Данный фреймворк обладает удобным интерфейсом для построения и работы с графами и позволяет визуально представить построенный граф.

Для разработки, отладки разрабатываемых модулей а так же для анализа данных использовалась интерактивная среда вычислений Jupyter Notebook [30].

3.6 Выводы

В данном разделе:

- были рассмотрена архитектура системы непрерывной интеграции;
- выставлены функциональные требования к разрабатываемой модели;
- спроектирован интерфейс модуля;
- спроектирована архитектура когнитивной карты для моделирования количества продаж.
- произведен выбор инструментальных средств;

4. Реализация и экспериментальная проверка модуля для нейро-нечеткого моделирования

В данной главе приводятся детали разработки и экспериментальной проверки разработанной системы. Описаны результаты экспериментов для разработанной модели сравниваются с результатами аналогичных экспериментов для существующих моделей. Описаны данные, которые использовались в экспериментах.

4.1 Состав и структура модуля для нейро-нечеткого моделирования

Реализованное ПО — это модуль на python, который позволяет помочь эксперту исследовать моделируемую систему и структурировать знания о системе.

Так как разработка проводилась методикой Test-Driven-Development [31, 32], были написаны автотесты, которые запускаются при обновлении кода в удаленном репозитории. Кроме автотестов, в среде по запуску тестов проверяется качество кода и возможные ошибки с помощью линтеров на соответствие кода стандарту *pep8*. Автотесты необходимы проекту на языке с динамической типизацией, так как в интерпретируемых языках с динамической типизацией интерпретатор на стадии компиляции программы не может проверить или вывести, чтобы проверить типы переменных и аргументов функций самостоятельно. Поэтому код, который не покрыт тестами может не работать и об этом разработчик узнает только после того, как программа упадет. Но раннее обнаружение этих ошибок с помощью автотестов поможет избежать проблем с использованием модуля в продакшне и поможет сократить время доведения продукта до состояния успешного релиза. Для того, чтобы было проще искать ошибки в коде и в тестах, тело тестов должно быть максимально простым. Разработанные автотесты покрывают как негативные, так и позитивные сценарии работы программы. Для разработанного модуля было реализовано 6 автотестов:

- `test_fit_on_history` — проверяет, что обучение карты на исторических данных с использованием оптимизатора *SGD* проходит без ошибок
- `test_fit_on_history_with_exogen_param` — проверяет, что обучение карты на исторических данных с использованием экзогенных параметров проходит без ошибки
- `test_sum_model` — проверяет, работу модели по суммированию концептов
- `test_double_freeze` — проверяет, что генерируется ошибка при неправильном использовании метода `freeze`.

- `test_freeze_history_mismatch` — проверяет, что генерируется ошибка при несовпадении размерностей исторических данных в концептах
- `test_train_lstm` — проверяет, что обучение карты на исторических данных с использованием оптимизатора *ADAM* проходит без ошибок

Отчет по тестированию представлен в приложении В.

4.2 Анализ данных социально-экономической системы

Анализ и предобработка данных были проведены с использованием библиотек *pandas* и *statmodels*. Набор данных, на которых будет проводиться тестирование разработанной системы представляет из себя несколько *csv* файлов: файл с количеством продаж продуктов разных категорий в 3 магазинах за 5 лет начиная с 2011-01-29 и файл с описанием праздников, которые выпадают на каждый день.

Обработка исходных данных включала в себя:

- Разбивка категориальных признаков на несколько числовых (индикаторных) с помощью `one-hot-encoding`;
- Объединение набора данных с праздниками и набора данных к продажам товаров в одну таблицу по ключевому столбцу `date`;
- Группировка исходных данных по дате с последующим суммированием внутри группы. Это преобразование позволяет получить общее количество продаж за один день для заданного набора данных.

описанные преобразования производились с помощью библиотеки *pandas*. Описание результирующего набора данных Г.2 и количественные характеристики Г.1 исследуемого набора данных представлены в приложении.

Требуется предсказать количество продаж на 30 дней вперед.

Для начала можно провести качественный визуальный анализ данных. Общее количество продаж можно оценить на графике 4.1. Из этого графика видно, что до середины 2012 года количество проданных товаров увеличивалось. Но после тренд пропал. В данных можно наблюдать как годовые, так и месячные и даже недельные сезонности. Каждый год в Рождество количество проданных товаров равно 0, потому что магазин не работает. По выходным количество проданных товаров больше, чем в будние дни.

Количество проданных товаров по каждому магазину представлено на следующем графике 4.2. В количестве проданных товаров отдельно по каждому магазину нет значительных отличий. Все магазины продают примерно одинаковое количество товаров.

Количество продаж по категориям для каждого магазина на графике 4.3 Во всех трех мага-

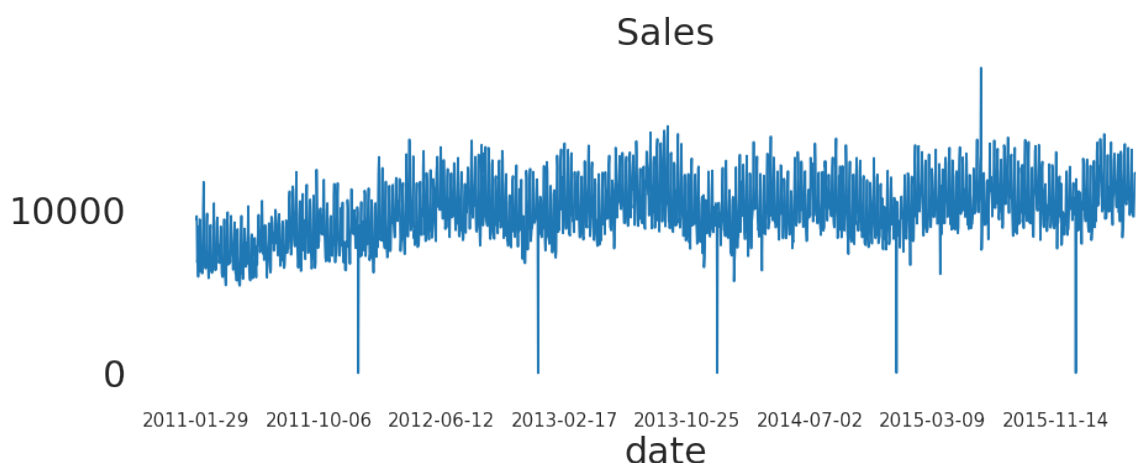


Рис 4.1 – Количество продаж

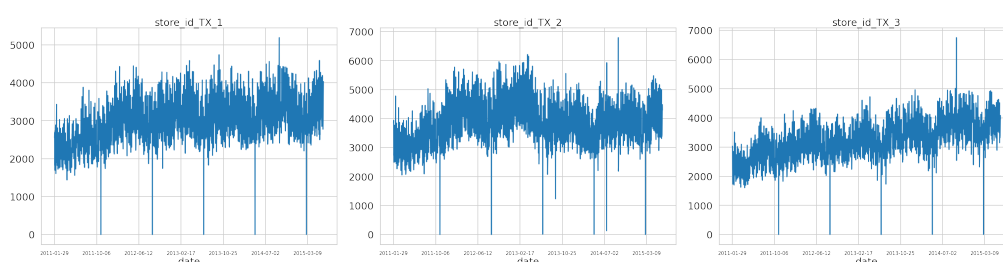


Рис 4.2 – Общее количество продаж по каждому магазину

знах количество продаж товаров в категории "еда" больше всего, а товаров для дома меньше всего. Из данного графика можно заметить, что сезонность присутствует в графике по продаже товаров в категории "еда" и для товаров для дома, но для товаров для хобби сезонности не наблюдаются.

Рассмотрим графики автокорреляции и частичной автокорреляции для общего количества продаж 4.4. Для лагов 1, 2, 3, 6, 7 присутствует статически значимая частичная автокорреляция. Это может быть использовано при построении модели SARIMAX. На графики автокорреляции видны синусоидальные колебания. Это подтверждает, что в данных есть сезонность.

Для обучения моделей использовался набор данных полностью за исключением последних 30 точек. Оставшиеся 30 точек использовались для валидации предсказаний исследуемых моделей.

4.3 Построение модели SARIMAX

Данная модель разрабатывалась с использованием библиотеки *statmodels*. Для оптимального подбора гиперпараметров был проведен визуальный анализ исходных данных, поиск перебором модели с минимальным значением критерия Акаике. Оптимизация параметров на полном наборе данных занимает примерно 10 секунд при 8 параметрах модели.



Рис 4.3 – Количество продаж товаров определенной категории для каждого магазина

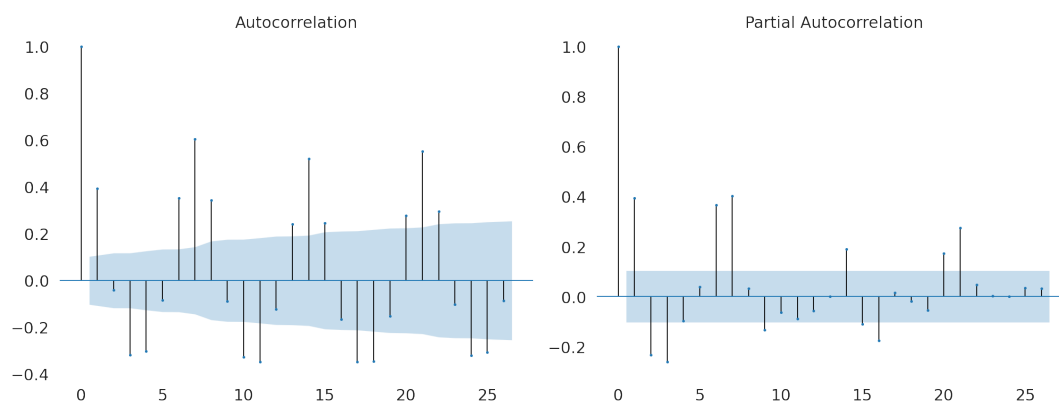


Рис 4.4 – Автокорреляция и частичная автокорреляция для общего количества продаж

Оптимальными гиперпараметрами оказались:

$$p = 2, d = 0, q = 3, P = 0, D = 1, Q = 1, S = 7 \quad (4.3.1)$$

Сезонность недельная - это логично, что в текущий день товаров будет продано столько же, как неделю назад.

На графике остатков 4.5 все еще прослеживается годичная сезонность и выбросы. Данные выбросы - возможно предсказать, если добавить экзогенные параметры в виде индикатора праздников.

В качестве экзогенных параметров рассмотрим индикатор дня недели. В исходные данные добавится 7 колонок. Каждому дню недели будет соответствовать одна колонка. И еще одним экзогенным параметром возьмем индикатор праздника — Рождества. Экзогенные параметры можно вычислить заранее, так как наперед можем поставить в соответствие индикатор дня недели определенному календарному дню и индикатор Рождества. Оптимизация параметров модели с 8 экзогенными переменными потребовало 20 секунд.

На графике остатков модели с экзогенными переменными 4.5 видно, что удалось избавиться от большого значения остатков на Рождество. Это получилось сделать благодаря столбцу-индикатору Рождественских праздников.

Но если проанализировать значения коэффициентов обученной модели A.1, можно заметить, что индикаторы дня недели слишком слабо влияют на предсказание модели. Значения

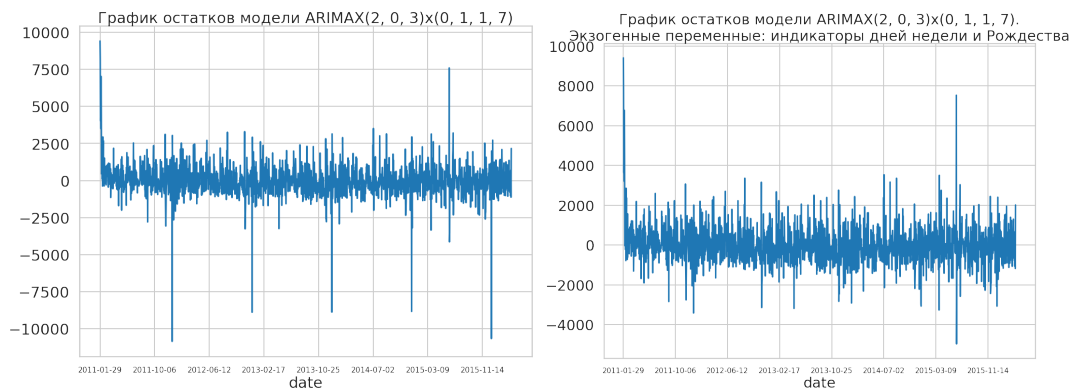


Рис 4.5 – График остатков модели ARIMAX с соответствующими параметрами

коэффициентов очень маленькие, вероятность того, что эти параметры будут иметь значение больше критического равно единице, дисперсия большая. Из этого можно сделать вывод, что использование этих экзогенных переменных неоправданно. Но зато значение коэффициента для индикатора Рождества ($xmas$) большое. Этот экзогенный параметр действительно объясняет некоторую полезную часть данных. Так, значение теста Харке-Бера [33] для предыдущей модели было равно 37658.88, а после выявления зависимости между количеством продаж и днем рождества стало равно 1488.02. Чем ближе значение этого теста к нулю, тем больше остатки модели похожи на нормальное распределение. Этот тест сверяет асимметрию и эксцесс остатков с значениями этих моментов для нормального распределения. И хотя с таким абсолютным значением теста нельзя говорить о том, что остатки распределены нормально, но относительно предыдущей модели, новая модель определенно лучше.

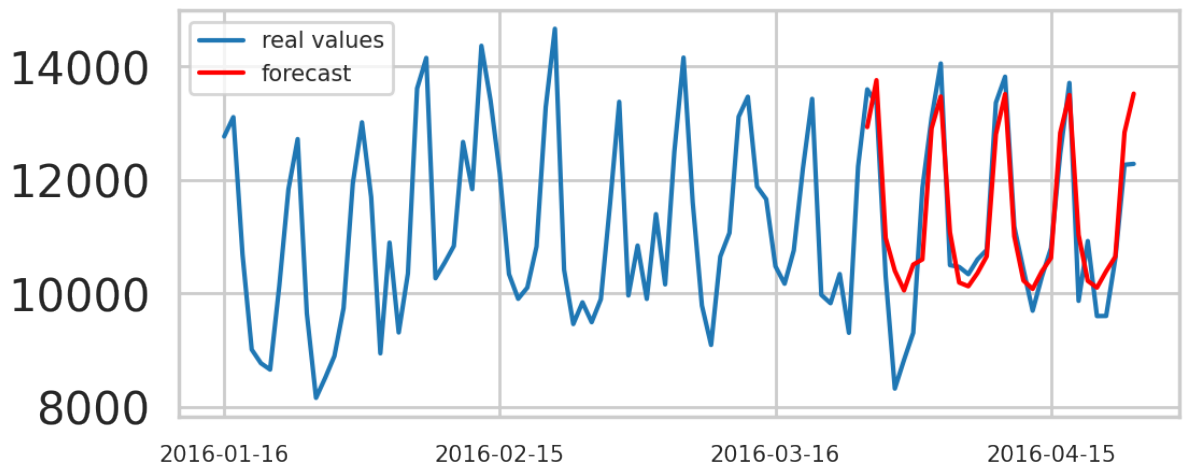
Но на предсказании на 30 дней вперед найденные экзогенные переменные не повлияли 4.6, так как в исследуемый момент времени не было Рождества.

Значение $RMSSE$ для модели с экзогенными параметрами ($RMSSE = 0.000210$) незначительно меньше модели без экзогенных переменных ($RMSSE = 0.000208$).

4.4 Построение модели LSTM

Данная модель разрабатывалась с использованием фреймворка *pytorch*. В отличие от таких статистических методов как ARIMAX, предсказание данных с помощью нейросетей требует более тщательно предобработки данных. Перед тем, как передать данные на вход нейросети, их нужно нормализовать. Для того, чтобы нормализовать данные, вычтем из данных за каждый день среднее количество продаж и разделим на стандартное отклонение. После того, как нейросетевая модель посчитает предсказание, полученные в предсказании числа нужно будет умножить на стандартное отклонение и прибавить среднее. Такая нейросеть будет работать с нечеткими числами. Которые после обучения и предсказания будут восстановлены в соответ-

Предсказание на 30 дней. ARIMAX((2, 0, 3)x(0, 1, 1, 7)).



Предсказание на 30 дней.
ARIMAX((2, 0, 3)x(0, 1, 1, 7)).
Экзогенные: Friday, Monday, Saturday, Sunday, Thursday, Tuesday, Wednesday, xmas

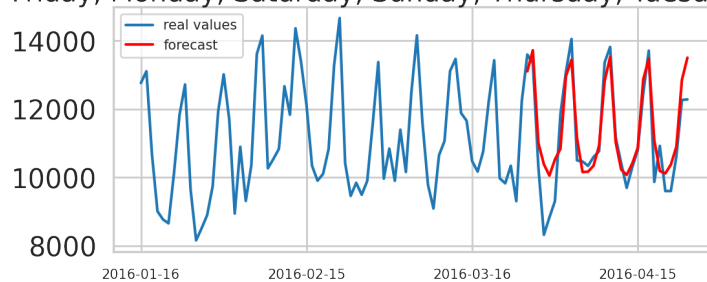


Рис 4.6 – Предсказания на 30 дней вперед

ствующие предсказанию значения. Важно, чтобы обучающая выборка была репрезентативной. Так как модель может вести себя неадекватно, если входные данные слишком сильно отличаются от данных, которые были на обучающей выборке.

К сожалению, предсказания, которые мы получили с помощью LSTM менее интерпретируемы, чем те, которые мы получили с помощью SARIMAX. Однако LSTM — это намного более мощная модель. И может быть использована в тех случаях, если мощности SARIMAX не хватает. Также можно использовать эти модели вместе. Например, сначала с помощью SARIMAX создать модель, которую можно интерпретировать, а потом с помощью LSTM можно предсказывать остатки, которые будет давать SARIMAX. И эти остатки можно прибавлять к исходному предсказанию SARIMAX. Такой алгоритм не позволит повысить интерпретируемость предсказания, но позволит увеличить его точность. Другой вариант — это использовать не остатки предсказаний ARIMAX модели, а сами предсказания в качестве одного из параметров LSTM модели 4.7. Теоретически, это должно позволить LSTM сети быстрее обучиться. В результате эксперимента было замечено, что действительно качество предсказаний на малых итерациях

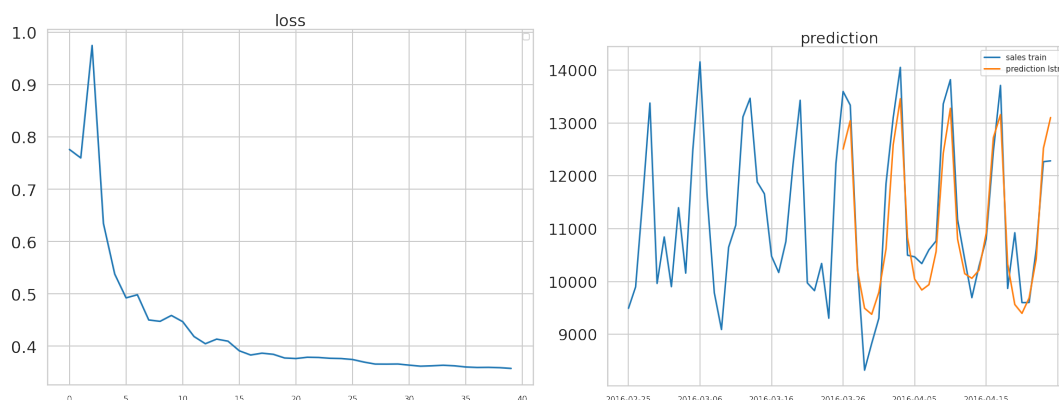


Рис 4.7 – Кривая потеря и предсказание LSTM-сети с использованием результатов предсказаний ARIMAX модели

при использовании предсказаний ARIMAX модели в LSTM модели позволили быстрее получить качественных результат. Но при увеличении количества итераций получить лучший результат по сравнению с использованием чистой LSTM модели не удалось. $RMSSE = 0.000174$ — это даже больше, хотя и незначительно, чем у обычной LSTM.

Результаты предсказаний с помощью модели LSTM очень похожи на результаты предсказаний с помощью SARIMAX. Стоит заметить, что LSTM была обучена и вычислялась с помощью GPU. В то время как оптимизация параметров модели SARIMAX производились на процессоре и для получения примерно одинаковых результатов требовалось столько же времени. Если бы LSTM обучалась на процессоре, время обучения бы исчислялось не секундами, а часами, так как это затратный по вычислительным мощностям процесс.

В результате эксперимента была обучена LSTM сеть с размерностью скрытого слоя равной 200. Результат обработки LSTM-сети после ее вычислений попадал в полносвязную сеть с размерностью тоже 200. Кривая потеря в процессе обучения и график с предсказаниями представлены на графиках 4.8. Значение $RMSSE = 0.000167$. Это немного меньше, чем у модели SARIMAX, но незначительно.

Кроме того, важным недостатком такой модели может быть то, что она не способна предсказывать данные из новой области значений. Для того, чтобы предсказать временной ряд с трендом, нужно или делать декомпозицию или преобразовать исходный ряд так, чтобы в нем не было тренда. После того, как получили предсказания, нужно проделать обратные преобразования. Это не всегда удобно. Возможно, этого недостатка можно избежать, если использовать модифицированную LSTM. Предположительно, если не использовать функции активации, то такой проблемы не будет. Но у такой модели в некоторых случаях больше вероятность получить взрыв градиентов, когда веса сети будут слишком большими. Такая ситуация ведет к

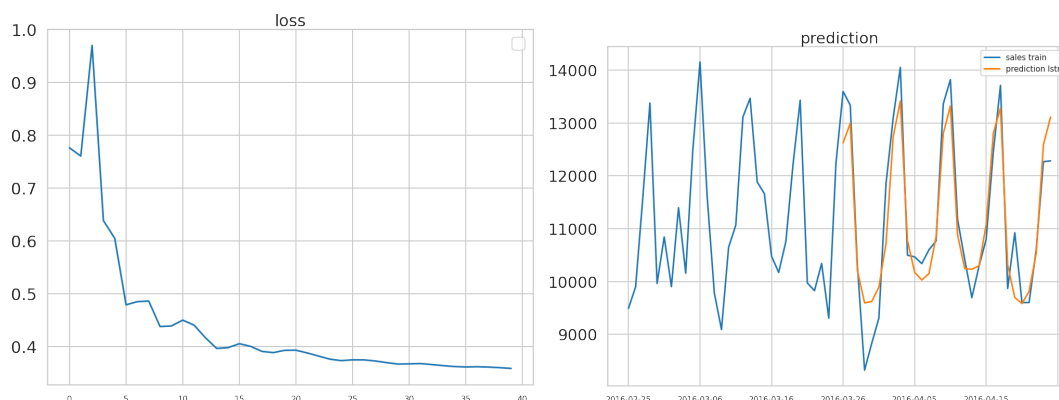


Рис 4.8 – Кривая потерь и предсказание LSTM-сети

невозможности обучить модель.

4.5 Построение модели FCM-LSTM

Данная модель разрабатывалась с использованием фреймворка *pytorch*. Для исследования модели была построена карта 3.3. Данные были разбиты на 9 временных рядов. Они были сгруппированы по магазину и по категории. Экзогенные переменные не использовались, так как для данной задачи не было найдено таких экзогенных параметров, которые могли бы значительно повлиять на краткосрочные предсказания.

В качестве модели в вершине карты были использованы сети LSTM с размерностями внутреннего состояния равными 50, 100, 200, 300. Каждая модель обучалась 40 эпох. С увеличением размерности скрытого состояния увеличивалось и время обучения.

Наилучшее значение $WRMSSE$ было получено для модели с размерностью 50 4.1. Веса ошибок каждого ряда были взяты единичные, так как ошибки для всех рядов расцениваются одинаково. Можно заметить, что слишком большие значения размерности скрытого состояния модели привели к ухудшению метрики $WRMSSE$. Это можно объяснить слишком сложной моделью. Не смотря на это, прогнозы исследуемых моделей отличаются незначительно. Поэтому разница в значениях ошибки предсказаний может быть даже просто случайной. Но по сравнению с моделью чистой LSTM и чистой ARIMAX, данная модель показала худшие результаты. Но зато модель с когнитивной картой может предсказывать разные временные ряды отдельно. Для сравнения предсказаний данной модели с моделями *LSTM* или *SARIMAX*, в таблице представлен столбец со значением $RMSSE$ для суммы предсказанных рядов по всем категориям и всем магазинам. Ошибка оказалась больше, чем у *LSTM* и *SARIMAX* моделей.

Результаты предсказаний модели с $hidden_size = 200$ представлены на графике 4.9. Можно заметить, что ряды с малыми колебаниями данная модель предсказывает плохо. Результаты предсказания выглядят сглаженными. Это можно объяснить тем, что в данных есть выбросы

Таблица 4.1 – Зависимость $WRMSSE$ и $RMSSE$ для суммы рядов в зависимости от размерности скрытого состояния сети

Размерность скрытого слоя	WRMSSE	RMSSE of sum
50	0.000370	0.000242
100	0.000405	0.000243
200	0.000388	0.000226
300	0.000416	0.000237



Рис 4.9 – Предсказания по каждой категории товара для каждого магазина с помощью FCM-LSTM с размерностью скрытого состояния 200, обученной на выборке с выбросами

(например, нет продаж на каждое Рождество). Для того, чтобы избежать этой проблемы, можно просто исключить выбросы из выборки. Или обучить модель на том участке данных, на котором выбросов нет. Уменьшение размера выборки может улучшить результат предсказаний для любой модели. Так как если данных очень много, то слишком старые показатели временного ряда могут быть не так полезны для модели, потому что более актуальная информация о процессе будет содержаться в части ряда, которая ближе к промежутку, который требуется предсказать. Кроме того, для борьбы с выбросами можно бороться и другими методами. Можно прологарифмировать ряд. Но такое преобразование сгладит пики при больших значениях временного ряда. И при обратном преобразовании маленькие отклонения исходного ряда могут превратиться в гигантские скачки или выбросы. Использование экзогенных параметров, таких как индикатор Рождества, не позволило бы справиться с проблемой.

Результаты предсказаний для карты с размерностью скрытого состояния 200, обученной на 110 предшествующих днях представлены на графиках 4.10.

В таблице 4.2 приведены ошибки для всех исследуемых типов моделей. Эксперимент был проведен еще раз для каждой модели. Следует отметить, что при повторном запуске значения ошибок могут варьироваться. Это объясняется тем, что некоторые веса в нейросетевых методах инициализируются случайным образом. Значения ошибок представленных в этой таблице можно считать средними.

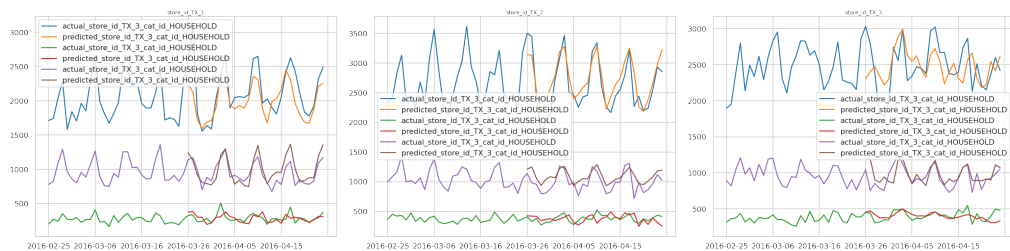


Рис 4.10 – Предсказания по каждой категории товара для каждого магазина с помощью FCM-LSTM с размерностью скрытого состояния 200, обученной на выборке без выбросов

Таблица 4.2 – Ошибки в предсказании общего количества продаж на 30 дней для исследованных типов моделей

Модель	RMSSE	MSE
SARIMAX(2, 0, 3)x(0, 1, 1, 7) с эндогенными переменными	0.0002077	526786
LSTM (hidden_size = 200)	0.0002312	398844
FCM-LSTM (hidden_size = 200), обученная на выборке с выбросами	0.0002354	676686
FCM-LSTM (hidden_size = 200), обученная на выборке без выбросов	0.0002129	553416

4.6 Выводы

В данной главе была реализована система для нейро-нечетного когнитивного картирования с использованием методов нейронных сетей, которая удовлетворяет требованиям, определенным в предыдущей главе.

Исследованы самостоятельные модели для прогнозирования временных рядов: *SARIMAX*, *LSTM*, *SARIMAX+LSTM*. Исследовано влияние выбросов на предсказание модели и границы применимости каждой модели. Разработанная модель была сравнена с другими моделями для предсказания временных рядов. Также описаны инструменты, с помощью которых была реализована система. Описана структура программного обеспечения, проведено тестирование разработанной системы.

Модель когнитивной карты, обученная на выборке с выбросами, имела самые большие показатели ошибок. Но такая же модель обученная на выборке без выбросов оказалась конкурентноспособной с другими моделями, не использующими когнитивные карты. Но у модели с когнитивной картой есть ряд преимуществ, описанных во второй и третьей главе данной работы:

- Разработанная модель на основе карты более интерпретируема;
- Предсказание такой модели более детальное: предсказаны значения каждой категории

товаров для каждого магазина;

- Больше возможностей для моделирования благодаря простому добавлению новых параметров;
- Разработанная модель гибка в плане дообучения;
- Для добавления новых параметров системы не требуется полное переобучение модели;

Однако есть и недостатки:

- Суммарная ошибка по предсказанию общего количества продаж больше, чем у методов, которые предсказывают сразу целевое значение объема продаж, а не продажи по категориям. Это можно объяснить тем, что количество прогнозируемых рядов больше. И ошибки при прогнозировании каждого ряда в сумме получаются больше, чем в предсказаниях самостоятельных моделей.
- Модель имеет большое количество параметров и требует больших вычислительных мощностей. Это ожидаемо, так как такая модель прогнозирует большее количество временных рядов. Для предсказания нескольких временных рядов можно использовать и одну LSTM сеть. И если сравнивать FCM-LSTM с LSTM для прогнозирования нескольких временных рядов, то FCM-LSTM будет требовать меньше ресурсов и будет проще обучаться. Потому что в FCM-LSTM строго определены, какие концепты могут влиять друг на друга, а какие — нет. В обычной LSTM для предсказания нескольких временных рядов, будет создано столько весов, сколько бы потребовалось для FCM-LSTM в случае, если бы такая карта представляла из себя полносвязный граф.

Заключение

В ходе работы были решены следующие задачи:

- Исследованы существующие методы прогнозирования временных рядов для моделирования социально-экономических;
- Разработан алгоритм для интеграции нейросетевых моделей с когнитивными картами;
- Спроектирована и реализована модель для когнитивного картирования социально-экономической системы с использованием нейросетевых методов;
- Предобработан и проанализирован тестовый набор данных, на котором были обучены исследуемые модели;
- Разработанная модель протестирована на открытых данных социально-экономических систем;
- Разработанный алгоритм нейро-нечеткого моделирования был сравнен с существующими решениями: SARIMAX, LSTM;

Разработанная модель хорошо подходит для моделирования сложных систем. При открытии нового магазина, в текущую модель потребуется просто добавить еще 4 концепта (один для магазина, и 3 для каждой категории товара). В то время как с помощью других исследованных моделей добавление еще одного параметра привело бы к необходимости их переобучения. Результирующая ошибка разработанной модели была незначительно больше, чем для других моделей (LSTM, SARIMX), но модель на основе когнитивной карты была более интерпретируема.

Для дальнейших исследований можно исследовать интеграцию когнитивной карты и авторегрессионной модели, например, SARIMAX. Если использовать алгоритмы автоматической настройки гиперпараметров для SARIMAX модели, эксперт может не тратить на это время и сосредоточиться на проектировании архитектуры модели, а не на деталях. Однако для более детального изучения модели эксперту стоит исследовать значения подобранных параметров а также исследовать значения коэффициентов полученных моделей. Такая модель будет легче в обучении. Количество параметров будет тоже меньше, чем у FCM-LSTM.

Также в будущем могут быть исследованы рекуррентные связи: если граф карты будет содержать циклы, то результат вычислений концепта, обработанный другими концептами будет поступать на вход самому себе. Это может как привести к взрыву градиентов, так и к получению более качественных предсказаний за счет того, что рекуррентность будет заключаться не

только в ячейках модели, (концептах, LSTM модели), но и в глобальной архитектуре модели. Рекуррентные модели проще поддаются Curriculum Learning [34, 35]. А рекуррентные модели для решения задачи классификации способны совершать предсказания, основанные на таксономии [35]. но эти предсказания можно уточнить в будущем, если провести больше вычислений. Однако для такой модели, скорее всего, простое добавление и удаление концептов будет плохо работать. Такая карта будет менее гибкая для моделирования и исследования следственно-причинных связей. При удалении или добавлении новых концептов такую карту нужно будет переобучать полностью сначала (как минимум ту часть, в которой содержится обратная связь).

Еще одним из направлений для дальнейших исследований является то, как распространяется градиент при обучении модели. В разработанной модели каждый концепт обучался изолированно относительно других концептов. Если позволить градиенту во время обучения распространяться на другие концепты, теоретически, можно повысить качество предсказаний такой модели.

Список литературы

1. Osoba Osonde, Kosko Bart. Beyond DAGs: Modeling Causal Feedback with Fuzzy Cognitive Maps. — 2019. — 1906.11247.
2. Kosko Bart. Fuzzy cognitive maps // International journal of man-machine studies. — 1986. — Vol. 24, no. 1. — P. 65–75.
3. К.В. Воронцов. Прогнозирование временных рядов [Электронный ресурс] // ШАД. — 2014. — Режим доступа: <http://www.machinelearning.ru/wiki/images/archive/c/cb/20160412121749!voronml-forecasting-slides.pdf>.
4. Akaike Hirotugu. Fitting autoregressive models for prediction // Annals of the institute of Statistical Mathematics. — 1969. — Vol. 21, no. 1. — P. 243–247.
5. Forecast of hourly average wind speed with ARMA models in Navarre (Spain) / Jose Luis Torres, Almudena Garcia, Marian De Blas, Adolfo De Francisco // Solar energy. — 2005. — Vol. 79, no. 1. — P. 65–77.
6. ARIMA models to predict next-day electricity prices / Javier Contreras, Rosario Espinola, Francisco J Nogales, Antonio J Conejo // IEEE transactions on power systems. — 2003. — Vol. 18, no. 3. — P. 1014–1020.
7. Peter Ďurka, Silvia Pastoreková. ARIMA vs. ARIMAX—which approach is better to analyze and forecast macroeconomic time series // Proceedings of the 30th International Conference Mathematical Methods in Economics, Karviná, Czech Republic. — 2012. — P. 11–13.
8. Omenzetter Piotr, Brownjohn James MW. A seasonal ARIMAX time series model for strain-temperature relationship in an instrumented bridge // Structural Health Monitoring 2005. — 2005.
9. Svozil Daniel, Kvasnicka Vladimir, Pospichal Jiri. Introduction to multi-layer feed-forward neural networks // Chemometrics and intelligent laboratory systems. — 1997. — Vol. 39, no. 1. — P. 43–62.
10. Zaremba Wojciech, Sutskever Ilya, Vinyals Oriol. Recurrent neural network regularization // arXiv preprint arXiv:1409.2329. — 2014.

11. Когнитивные карты с интерпретацией концептов и связей между ними совокупностью аргументов двузначной логики / АИ Каяшев, МИ Шарипов, ЕА Муравьёва, КА Багров // XII всероссийское совещание по проблемам управления ВСПУ-2014. — 2014. — Р. 4126–4131.
12. Kreinovich Vladik, Stylios Chrysostomos. Why Fuzzy Cognitive Maps Are Efficient // International Journal of Computers Communications & Control. — 2015. — 10. — Vol. 10. — P. 65.
13. Puheim Michal, Vascak Jan, Madarasz L. Three-term relation neuro-fuzzy cognitive maps // CINTI 2014 - 15th IEEE International Symposium on Computational Intelligence and Informatics, Proceedings. — 2015. — 01. — P. 477–482.
14. Hochreiter Sepp, Schmidhuber Jürgen. Long Short-term Memory // Neural computation. — 1997. — 12. — Vol. 9. — P. 1735–80.
15. Diederik P. Kingma Jimmy Ba. Adam: A Method for Stochastic Optimization. — 2014. — 1412.6980.
16. Friedman Jerome H. Stochastic gradient boosting // Computational statistics & data analysis. — 2002. — Vol. 38, no. 4. — P. 367–378.
17. Hyndman Rob J, Koehler Anne B. Another look at measures of forecast accuracy // International journal of forecasting. — 2006. — Vol. 22, no. 4. — P. 679–688.
18. A survey of forecast error measures / Maxim Vladimirovich Shcherbakov, Adriaan Brebels, Nataliya Lvovna Shcherbakova et al. // World Applied Sciences Journal. — 2013. — Vol. 24, no. 24. — P. 171–176.
19. Sakamoto Yosiyuki, Ishiguro Makio, Kitagawa Genshiro. Akaike information criterion statistics // Dordrecht, The Netherlands: D. Reidel. — 1986. — Vol. 81.
20. Akaike H. A new look at the statistical model identification // IEEE Transactions on Automatic Control. — 1974. — Vol. 19, no. 6. — P. 716–723.
21. Wagenmakers Eric-Jan, Farrell Simon. AIC model selection using Akaike weights // Psychonomic bulletin & review. — 2004. — Vol. 11, no. 1. — P. 192–196.
22. Маклафлин Б, Поллайс Г. Объектно-ориентированный анализ и проектирование. — 2013.
23. Oliphant Travis E. A guide to NumPy. — Trelgol Publishing USA, 2006. — Vol. 1.
24. pandas development team The. pandas-dev/pandas: Pandas. — 2020. — Feb. — Access mode: <https://doi.org/10.5281/zenodo.3509134>.

25. Seabold Skipper, Perktold Josef. statsmodels: Econometric and statistical modeling with python // 9th Python in Science Conference. — 2010.
26. PyTorch: An Imperative Style, High-Performance Deep Learning Library / Adam Paszke, Sam Gross, Francisco Massa et al. // Advances in Neural Information Processing Systems 32 / Ed. by H. Wallach, H. Larochelle, A. Beygelzimer et al. — Curran Associates, Inc., 2019. — P. 8024–8035. — Access mode: <http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf>.
27. Abadi Martín, Agarwal Ashish, Barham Paul et al. TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems. — 2015. — Software available from tensorflow.org. Access mode: <https://www.tensorflow.org/>.
28. Krekel Holger, Oliveira Bruno, Pfannschmidt Ronny et al. pytest x.y. — 2004. — Access mode: <https://github.com/pytest-dev/pytest>.
29. Hagberg Aric A., Schult Daniel A., Swart Pieter J. Exploring network structure, dynamics, and function using NetworkX // Proceedings of the 7th Python in Science Conference (SciPy2008). — Pasadena, CA USA, 2008. — Aug. — P. 11–15.
30. Jupyter Notebooks – a publishing format for reproducible computational workflows / Thomas Kluyver, Benjamin Ragan-Kelley, Fernando Pérez et al. // Positioning and Power in Academic Publishing: Players, Agents and Agendas / Ed. by F. Loizides, B. Schmidt ; IOS Press. — 2016. — P. 87 – 90.
31. Beck Kent. Test-driven development: by example. — Addison-Wesley Professional, 2003.
32. Janzen David, Saiedian Hossein. Test-driven development concepts, taxonomy, and future direction // Computer. — 2005. — Vol. 38, no. 9. — P. 43–50.
33. Thadewald Thorsten, Büning Herbert. Jarque–Bera test and its competitors for testing normality—a power comparison // Journal of applied statistics. — 2007. — Vol. 34, no. 1. — P. 87–105.
34. Curriculum learning / Yoshua Bengio, Jérôme Louradour, Ronan Collobert, Jason Weston // Proceedings of the 26th annual international conference on machine learning. — 2009. — P. 41–48.
35. Wu Te-Lin, Shen Bo-Kui. Feedback based Neural Networks. — 2016.

А. Приложение 1. Результаты экспериментов

Таблица А.1 – Значения коэффициентов параметров ARIMAX(2, 0, 3)x(0, 1, 1, 7) с экзогенными переменными: индикатор дня недели и индикатор Рождества

Параметр	Значение коэффициента	Дисперсия	P-value
intercept	3.7294	1.686	0.027
Friday	0.0159	2440.530	1.000
Monday	0.0045	5202.932	1.000
Saturday	-0.0011	7994.566	1.000
Sunday	0.0055	4295.359	1.000
Thursday	-0.0069	5951.614	1.000
Tuesday	-0.0007	7381.431	1.000
Wednesday	0.0078	3557.219	1.000
xmas	-8738.4506	399.352	0.000
ar.L1	0.0149	0.067	0.824
ar.L2	0.8291	0.056	0.000
ma.L1	0.3286	0.074	0.000
ma.L2	-0.5438	0.062	0.000
ma.L3	-0.1352	0.038	0.000
ma.S.L7	-0.9363	0.017	0.000
sigma2	1.264e+06	3.87e+04	0.000

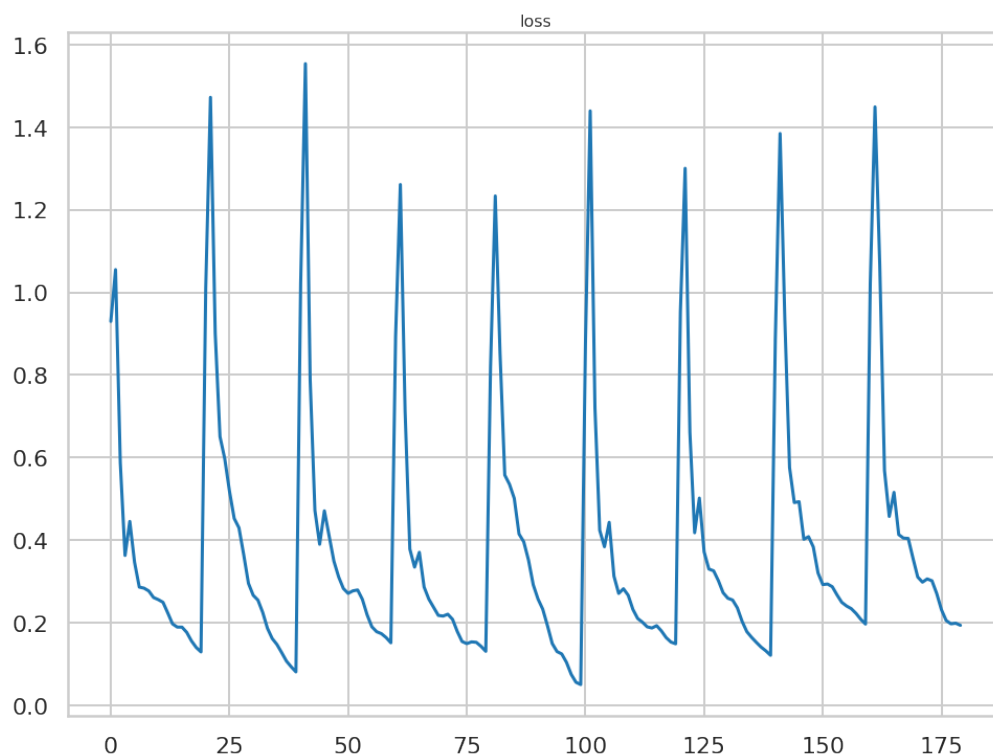


Рис А.1 – Графики значений функции потерь для FCM-LSTM. Каждый концепт обучался 20 эпох.

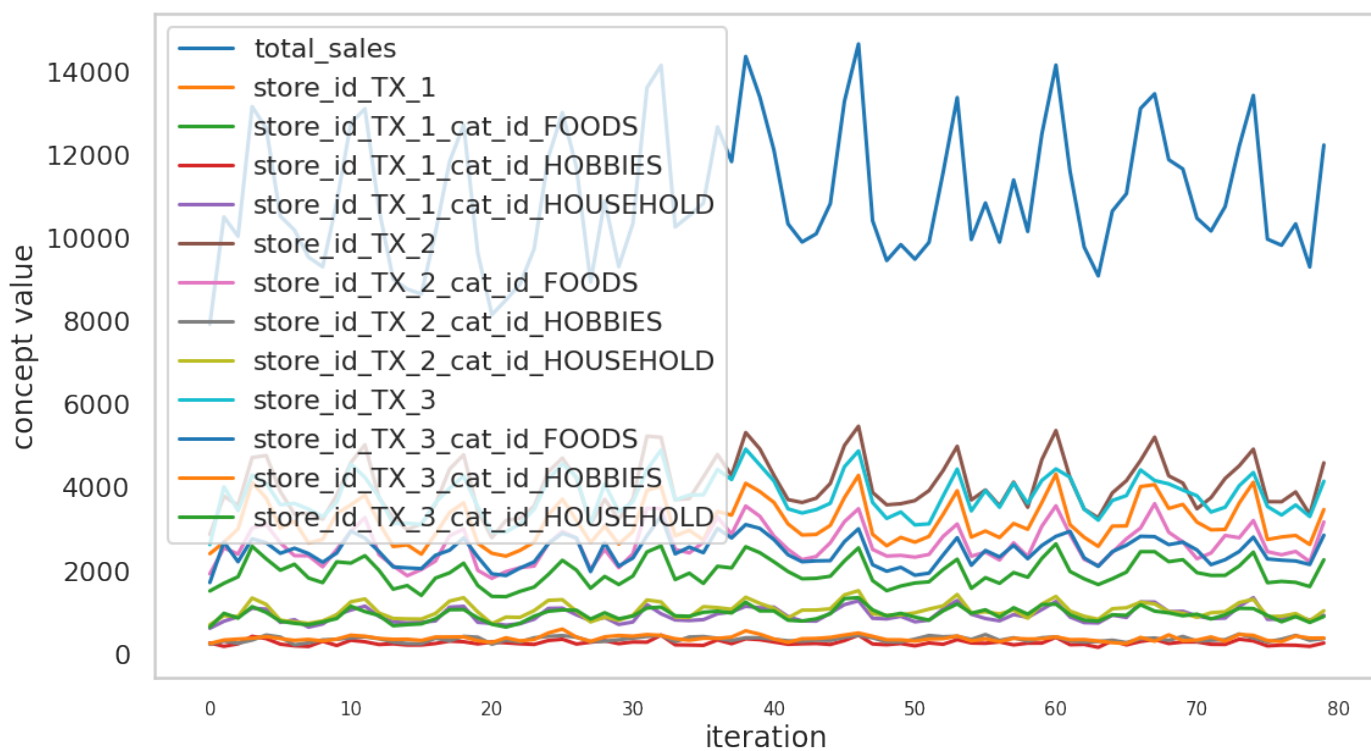


Рис А.2 – Результат работы метода plot_concept_history для карты, обученной на 80 точках исследуемых данных.

В. Приложение 2. Отчет по тестированию разработанного модуля на Python

```
===== test session starts =====  
  
platform linux -- Python 3.7.6, pytest-5.3.5, py-1.8.1, pluggy-0.13.1 -- /home/d.tarasov  
collected 6 items  
  
fcm_test.py::TestFcm::test_fit_on_history PASSED [ 16%]  
fcm_test.py::TestFcm::test_fit_on_history_with_exogen_param PASSED [ 33%]  
fcm_test.py::TestFcm::test_sum_model PASSED [ 50%]  
fcm_test.py::TestFcm::test_double_freeze PASSED [ 66%]  
fcm_test.py::TestFcm::test_freeze_history_mismatch PASSED [ 83%]  
fcm_test.py::TestFcm::test_train_lstm PASSED [100%]  
  
===== 6 passed in 3.26s =====
```

Г. Приложение 3. Описание исследуемого набора данных

Характеристика	Значение
Занимаемый объем оперативной памяти	3.7MB
Количество строчек в наборе данных	17217
Количество столбцов	18
Минимальное значение поля "date"	2011-01-29
Максимальное значение поля "date"	2016-04-24

Таблица Г.1 – Количественные характеристики исследуемого набора данных

Название столбца	Тип данных	Описание
date	Дата	"2011-01-29", "2013-10-23"
cat_id_FOODS	Число	1, если товар в категории "Еда", иначе 0
cat_id_HOBBIES	Число	1, если товар в категории "Товары для хобби", иначе 0
cat_id_HOUSEHOLD	Число	1, если товар в категории "Товары для дома", иначе 0
store_id_TX_1	Число	1, если товар был в магазине "TX_1", иначе 0
store_id_TX_2	Число	1, если товар был в магазине "TX_2", иначе 0
store_id_TX_3	Число	1, если товар был в магазине "TX_3", иначе 0
weekday_Monday	Число	1, если дата — понедельник, иначе 0
weekday_Tuesday	Число	1, если дата — вторник, иначе 0
weekday_Wednesday	Число	1, если дата — среда, иначе 0
weekday_Thursday	Число	1, если дата — четверг, иначе 0
weekday_Friday	Число	1, если дата — пятница, иначе 0
weekday_Saturday	Число	1, если дата — суббота, иначе 0
weekday_Sunday	Число	1, если дата — воскресенье, иначе 0
event_type_Cultural	Число	1, если дата — культурный праздник, иначе 0
event_type_National	Число	1, если дата — национальный праздник, иначе 0
event_type_Religious	Число	1, если дата — религиозный праздник, иначе 0
event_type_Sporting	Число	1, если дата — спортивный праздник, иначе 0

Таблица Г.2 – Описание набора данных по количеству продаж товаров разных категорий.