

ПРАВИТЕЛЬСТВО РОССИЙСКОЙ ФЕДЕРАЦИИ  
ФГАОУ ВО НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ  
«ВЫСШАЯ ШКОЛА ЭКОНОМИКИ»

Факультет компьютерных наук  
Образовательная программа «Прикладная математика и информатика»

УДК 51-77

**Отчет об исследовательском проекте на тему:**  
**Моделирование развития городов с помощью клеточных автоматов**

**Выполнил:**

студент группы БПМИ226  
Леонтенков Егор Евгеньевич



(подпись)

26.04.2024

(дата)

**Принял руководитель проекта:**

Кандидат технических наук, доцент,  
Родригес Залепинос Рамон Антонио  
Старший преподаватель  
Факультета компьютерных наук НИУ ВШЭ

(подпись)

(дата)

**Москва 2024**

# Содержание

|   |           |
|---|-----------|
| <b>Аннотация</b>  | <b>4</b>  |
| <b>1 Введение</b>   | <b>5</b>  |
| 1.1 Описание предметной области . . . . .                     | 5         |
| 1.2 Постановка задачи . . . . .                               | 5         |
| 1.3 Структура работы . . . . .                                | 6         |
| 1.4 Обзор литературы . . . . .                                | 6         |
| 1.4.1 Задача моделирования развития городов . . . . .         | 6         |
| 1.4.2 Статьи с использованием клеточных автоматов . . . . .   | 6         |
| <b>2 Рассматриваемая область и датасет</b>                    | <b>7</b>  |
| 2.1 Обозреваемая область . . . . .                            | 7         |
| 2.2 Выбор и обработка датасета . . . . .                      | 7         |
| 2.2.1 Работа в ГИС . . . . .                                  | 7         |
| 2.2.2 Обработка в Jupyter Notebook . . . . .                  | 9         |
| <b>3 Анализ и построение моделей для алгоритма</b>            | <b>11</b> |
| 3.1 Обзор . . . . .   | 11        |
| 3.2 Модель . . . . .  | 13        |
| 3.3 Реализация алгоритма . . . . .                            | 14        |
| 3.3.1 Использование мультиномиального распределения . . . . . | 14        |
| 3.3.2 Предлагаемый метод «Basic» . . . . .                    | 15        |
| 3.3.3 Предлагаемый метод «Update» . . . . .                   | 15        |
| 3.3.4 Предлагаемый метод «Neighbors» . . . . .                | 16        |
| 3.4 Обучения моделей . . . . .                                | 18        |
| <b>4 Тестирование алгоритма</b>                               | <b>18</b> |
| 4.1 Метрики для тестирования . . . . .                        | 18        |
| 4.2 Проведение тестового предсказания . . . . .               | 20        |
| 4.2.1 Тест метода Basic . . . . .                             | 21        |
| 4.2.2 Тест метода Update . . . . .                            | 22        |
| 4.2.3 Тест метода Neighbors с mode=0 . . . . .                | 22        |
| 4.2.4 Тест метода Neighbors с mode=1 . . . . .                | 23        |
| 4.3 Итог . . . . .  | 23        |

|  |           |
|--|-----------|
| <b>5 Моделирование развития города</b> | <b>24</b> |
| 5.1 Применение модели . . . . .        | 24        |
| 5.2 Анализ результатов . . . . .       | 24        |
| <b>Заключение</b>                      | <b>25</b> |
| <b>Список литературы</b>               | <b>27</b> |

## **Аннотация**

Изучение изменения планировки в городских и региональных системах резко изменилось за годы в связи с появлением и применением моделей клеточных автоматов. Модели клеточных автоматов способны имитировать изменения в городской планировке. Несмотря на заметные достижения в этой области, остаются значительные пробелы между городскими процессами, смоделированными в моделях, и фактической динамикой развивающихся городских систем. Целью данного исследования является мониторинг, оценка, моделирование и анализ динамики пространственно-временного роста в городе Москва, Россия, с помощью клеточной автоматической модели СА-Markov (cellular automata Markov). Модель СА-Markov может быть эффективно использована для изучения городской динамики в быстро растущих городах. Будучи эффективным инструментом для представления пространственных структур, информация, генерируемая им, может быть использована для прогнозирования сценариев развития городов для обеспечения устойчивого роста.

Исследование показывает, что модели клеточных автоматов способны предсказывать развитие городов с точностью до 89%. Моделирование предсказывает четкую тенденцию трансформации различных классов землепользования в области расширения городских застроек. Сделан вывод, что использование клеточного автомата с другими моделями для классификации может повысить эффективность работы алгоритма и стать инструментом поддержки принятия решений для политиков в области проектирования устойчивых городских сред обитания.

## **Ключевые слова**

СА-Markov, клеточный автомат, моделирование развития города, многомерные процессы городских изменений, геоинформационная система, большие данные.

# **1 Введение**

## **1.1 Описание предметной области**

Урбанизация является не только неизбежным процессом из-за экономического развития и быстрого роста населения, но и серьезной проблемой многих регионов мира. Точная и актуальная информация об изменении покрова необходима для понимания и оценки экологических последствий изменений, потому что незапланированная урбанизация создает такие проблемы, как загрязнение и перегруженность дорог транспортом. Таким образом, становится понятно, что исследование градостроения будет не только всегда актуально, но и полезно для дальнейшего планирования при хороших результатах.

По определению, клеточные автоматы - это динамические модели, дискретные во времени, пространстве и состоянии. Клеточные автоматы используются при решении многих задач. Примерами применения клеточных автоматов можно назвать анализ и прогнозирование физических процессов, генерация случайных чисел в криптографии, а также помочь в имитировании химических реакций в химии. Сложность использования клеточного автомата заключается в поиске подходящего алгоритма для его реализации в применяемой области. Также при моделировании возникает проблема в поиске актуальных и правильных данных, к которым будет применен алгоритм.

Гибридная модель, состоящая из цепи Макрова и клеточных автоматов, была разработана для повышения производительности стандартной модели логистической регрессии для имитации городского расширения в 1980 году. Поэтому в последнее время такие модели стали часто использоваться для прогнозирования развития городов. Использование ГИС при создании модели помогает при мониторинге и оценки динамических изменений трансформации землепользования.

## **1.2 Постановка задачи**

В этом исследовании предпринимается попытка отобразить будущее состояние землепользования и земельного покрова города с использованием векторных геоданных и одной из моделей клеточных автоматов. При исследовании на вход поступают векторные геоданные и геопространственные инструменты для количественной оценки и анализа пространственно-временного изменения городского землепользования и ландшафтного покрова в течение всего периода исследования. Задача заключалась в построении модели и предсказании изменений городской планировки, которые могут произойти в ближайшие несколько лет. Также

исследовалась точность предсказаний в зависимости от качества растровых изображений.

### 1.3 Структура работы

Работа организована следующим образом. В разделе 1.4 дается обзор литературы по изученной теме. Далее, в разделах 2 и 3 описываются обработка данных и создание разных моделей для решения задачи, в частности, описание моделей и их обучение. В разделе 4 приводятся метрики и их применение для анализа моделей. В итоговом разделе 5 происходит моделирование развития города с помощью выбранной модели.

### 1.4 Обзор литературы

#### 1.4.1 Задача моделирования развития городов

Задача моделирования развития городов была актуальна всегда, поэтому существует множество разносторонних подходов для ее решения. Обозревая статьи, можно часто встретить применение машинного обучения, например в статье Yisa Ginath Yuh [4]. При использовании ML моделей чаще всего делают с помощью алгоритмов k-ближайших соседей (kNN), случайных лесов (RF) или искусственных нейронных сетей. Но стоит учитывать, что в результате статьи выше, пришли к выводу, что модели ML не широко применяются в некоторых уголках земли из-за методологических проблем, которые возникают при использовании спутниковых изображений с грубым разрешением. Однако, для решения задачи чаще используются клеточные автоматы, которые стали популярны в этой области ближе к 2010 году. Далее, проводится рассмотрение нескольких статей на эту тему и позиционирование данной работы относительно них.

#### 1.4.2 Статьи с использованием клеточных автоматов

Авторы статьи Modelling urban change with cellular automata [3] решили рассмотреть проблемы данного метода и пришли к выводу, что несмотря на заметные достижения в этой области, остаются значительные пробелы между городскими процессами, смоделированными в моделях СА, и фактической динамикой развивающихся городских систем. Однако, они выделили, что возможное расширение входных данных в качестве аналитической основы сможет улучшить итоговые исследования.

В статье Urban sprawl modeling using cellular automata [1] была реализована модель CA-Markov, но для входных данных использовались фото с камеры LISS-IV. LISS-IV - мно-

госпектральная камера высокого разрешения, работающая в трех спектральных диапазонах. Использование спутниковых изображений как входных данных упрощает работу для калибровки и тестирования модели. В заключении, Shikhar Deep приходит к выводу, что такое моделирование может стать инструментом системы поддержки принятия решений для политиков при разработке планов городского расширения с подходом к устойчивому развитию среды обитания.

Нельзя не выделить одну из последних работ в этой области [2], проводимой Ergo Beyene. В работе был успешно запущен алгоритм, было выделено, что фокус модели CA-Markov сосредоточен на временных рядах, пространстве, а также CA имеет ограничение в предоставлении эмпирических описаний изменений в землепользовании.

Таким образом, настоящее исследование запускало несколько моделей CA-Markov с использованием векторных геоданных. Нужно подчеркнуть, что ранее исследования с данной изучаемой областью и таким датасетом не проводились.

## 2 Рассматриваемая область и датасет

### 2.1 Обозреваемая область

Данное исследование посвящено городу Москва, Россия. Координаты географического центра города:  $55^{\circ}33'31,46''$  северной широты и  $37^{\circ}22'43,84''$  восточной долготы. Общая площадь составляет  $2511 \text{ км}^2$ .

### 2.2 Выбор и обработка датасета

Для нашего датасета были выбраны наборы векторных геоданных для 2015, 2020 и 2024 годов. Векторные геоданные – это тип географических данных, в которых информация хранится в виде набора точек, линий или полигонов, а также атрибутивных данных этих объектов. Сложность состояла в том, чтобы из наших данных [2.1] сделать удобные входные для кода. Для этого была сделана обработка данных, описанная ниже.

#### 2.2.1 Работа в ГИС

Слои данных (коллекции географических данных) относятся к разным классам, но можно заметить, что многие из них содержат похожие типы покрова. Чтобы уменьшить вычислительную нагрузку и усилия, а также улучшить предсказания изменений землепользования из года в год, категории слоев были реклассифицированы в ГИС руками. Была

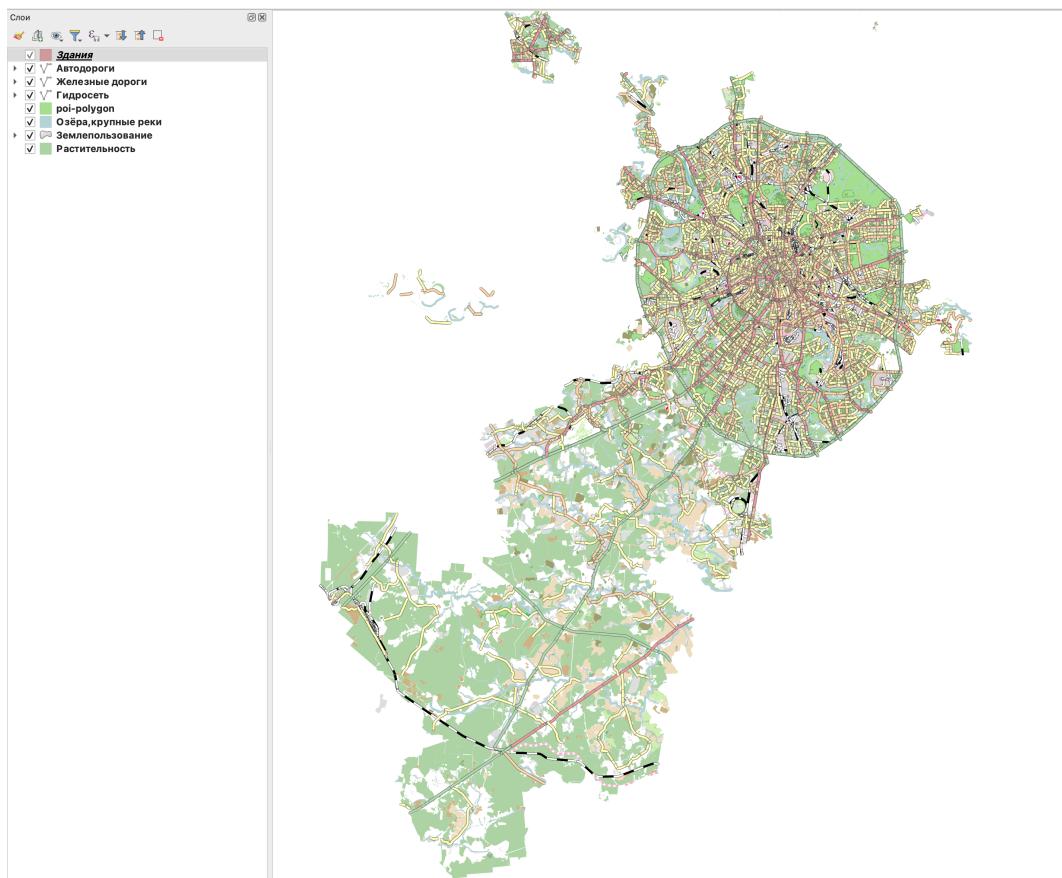


Рис. 2.1: Начало работы с данными.

получена новая классификация [2.1].

Таблица 2.1: Реклассификация

| Новый класс землепользования | Содержит классы                                       |
|------------------------------|---|
| Water                        | Озера + Реки + Бассейны водостока + Гидросети         |
| Urban                        | Застройки + Сдания + Гаражи + Промышленные территории |
| Road                         | Автодороги + Железная дорога + Подвесные дороги       |
| Land                         | Пустые области + Территория неклассифицированной суши |
| Vegetation                   | Леса + Парки + Луга + Фермы + Кустарники + Песок      |

Далее, чтобы легче было читать данные, было принято решение преобразовать карты в растровые изображения с разным качеством. Растворная графика — это графическое изображение, состоящее из массива сетки пикселей, или точек различных цветов, которые имеют одинаковый размер и форму. Таким образом, мы имели для каждой карты по два изображения: одно с 200dpi, другое с 1000dpi. DPI — сокращение для англ. dots per inch, количество точек на дюйм. Процесс преобразования можно увидеть на изображении [2.2].

Основываясь на цели исследования, а также на общих условиях исследуемой области, растровые изображения содержали в себе пять классов наземного покрова (рисунок 2.2).

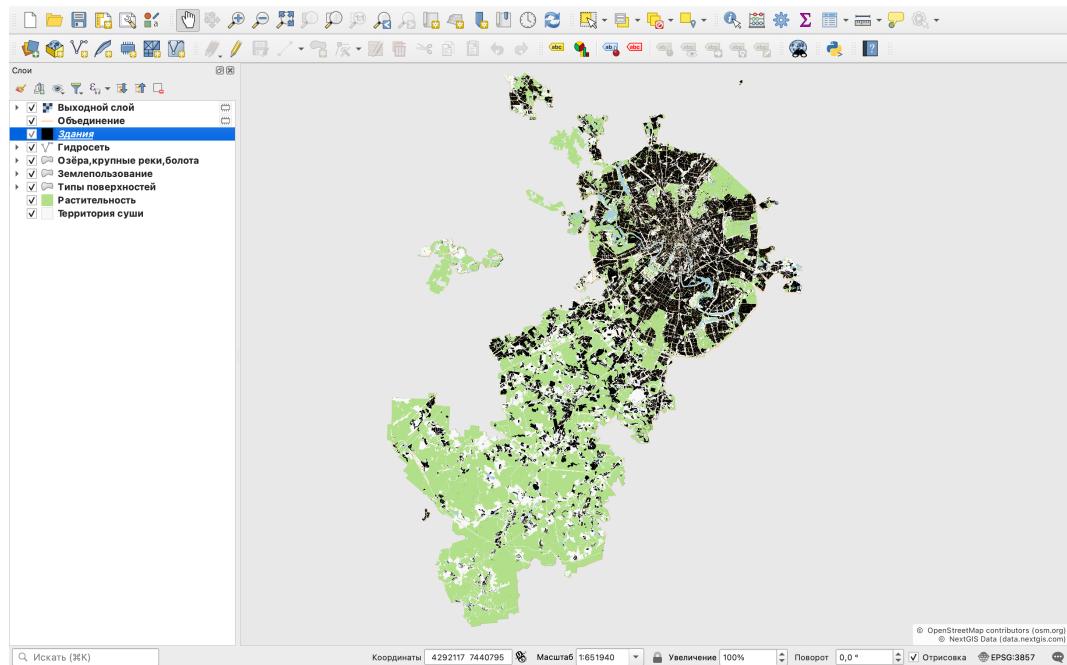


Рис. 2.2: Процесс обработки данных.

### 2.2.2 Обработка в Jupyter Notebook

Так как на предыдущем шаге мы получили изображения, то некоторые пиксели, характеризовавшие классы, стали размытыми из-за свойств преобразования карт в изображения, что в дальнейшем могло повлиять на работу алгоритма. Во избежание проблем, было решено, что нужно обновить каждое изображение. То есть для каждого изображения отнести пиксели к «идеальным» цветам (характеризовавшим классы) с помощью формулы цветового отличия [1].

$$D = \sqrt{(R_{base} - R)^2 + (G_{base} - G)^2 + (B_{base} - B)^2}, \quad (1)$$

где:

- R,G,B - каналы проверяемого цвета;
- $_base$  - это каналы «идеального» цвета, относительно которого проверяем;
- D - удаленность проверяемого цвета от идеального.

Чтобы отнести полученный цвет к некоторому классу: производится сортировка D по возрастанию для каждого цвета и выбирается наименьший по цветовому отличию класс. Таблицу «идеальных» цветов можно наблюдать ниже [2.2].

После предварительного просмотра полученных карт обнаружилось, что на изображениях карты расположены со смещениями. Например: верная точка карты для 2015 года

Таблица 2.2: Идеальные цвета

| Класс (номер)  | Идеальный цвет (представление в RGB) |
|----------------|--------------------------------------|
| Empty (0)      | Серый (222, 222, 222)                |
| Water (1)      | Синий (174, 208, 220)                |
| Urban (2)      | Черный (0, 0, 0)                     |
| Road (3)       | Оранжевый (255, 184, 65)             |
| Land (4)       | Зеленый (168, 228, 160)              |
| Vegetation (5) | Белый (255, 255, 255)                |

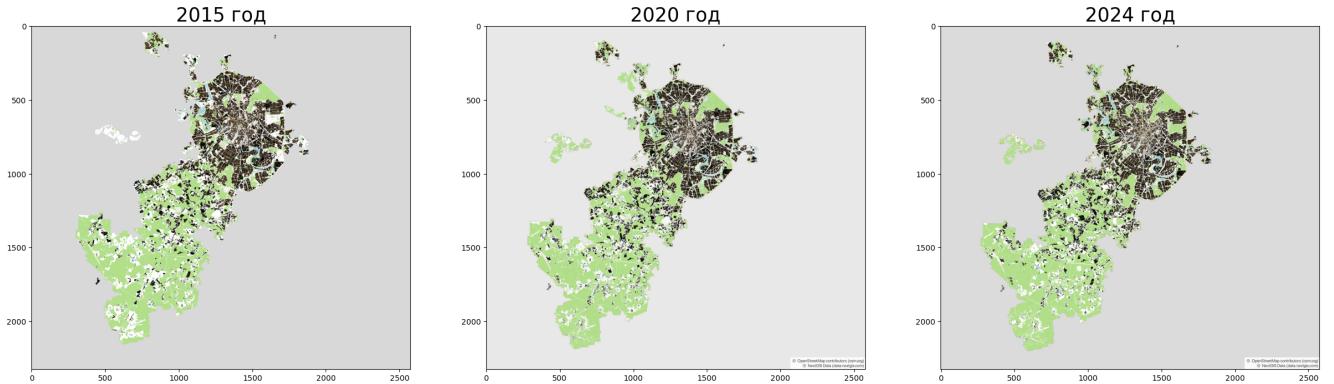


Рис. 2.3: Изображения до обработки для 200dpi.

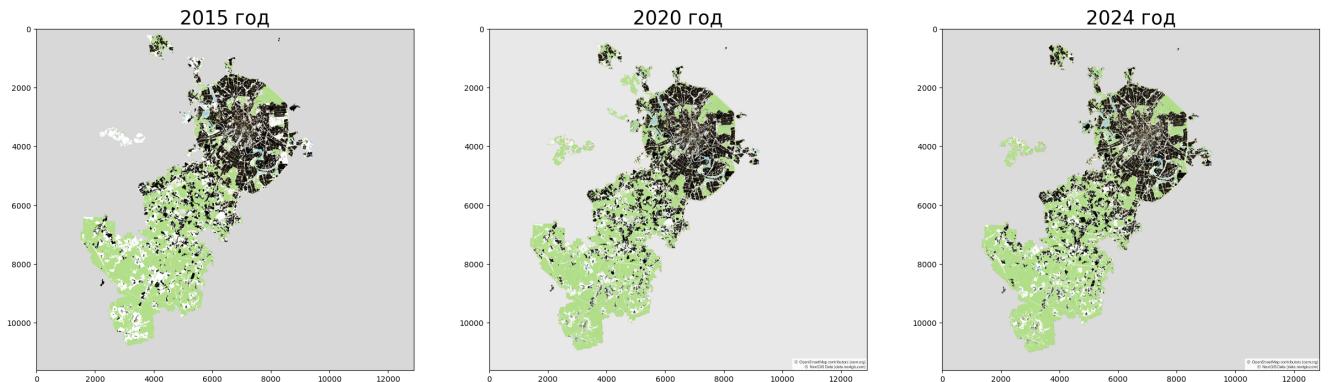


Рис. 2.4: Изображения до обработки для 1000dpi.

была расположена выше, чем верхняя точка для 2020 года. Это также могло повлиять на результат работы алгоритма, из-за чего пришлось централизовать карты. Также стало известно, что карты имеют разные размеры. Но так как их размеры различаются максимум на 40 клеток для 200dpi и 150 клеток для 1000dpi по одной оси, то мы можем считать это погрешностью для работы алгоритма. В итоге размеры карт изменились с  $2575 \times 2325$  на  $2150 \times 1600$  для 200dpi, а также  $12875 \times 11625$  на  $10800 \times 8100$  для 1000dpi. Результаты до и после расположены на рисунках [2.3], [2.4], [2.5] и [2.6]. Как и ожидалось, изображения с 1000dpi намного детальнее.

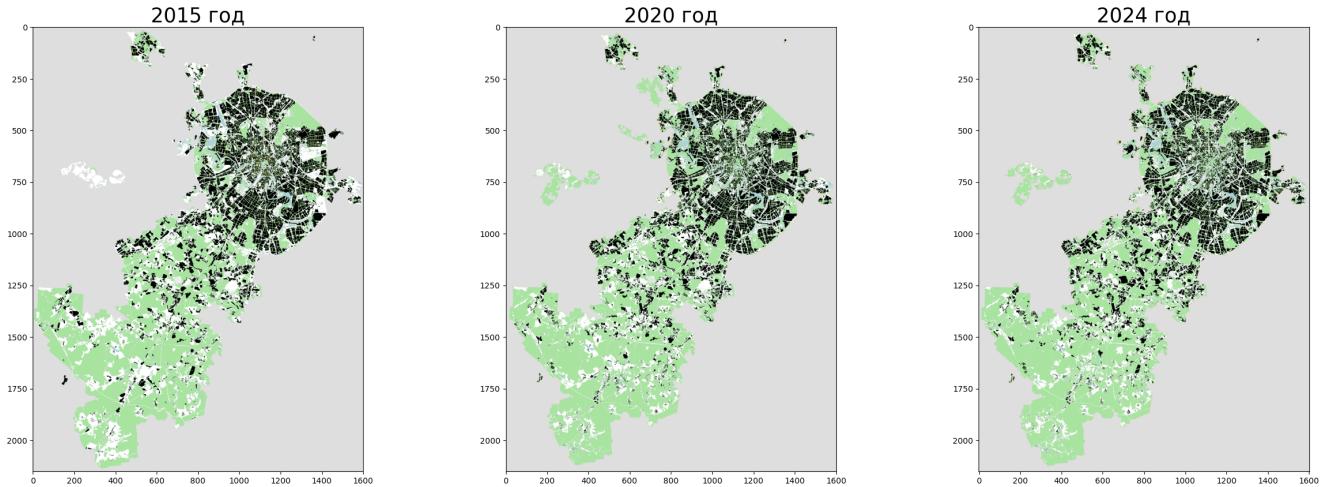


Рис. 2.5: Изображения после обработки для 200dpi.

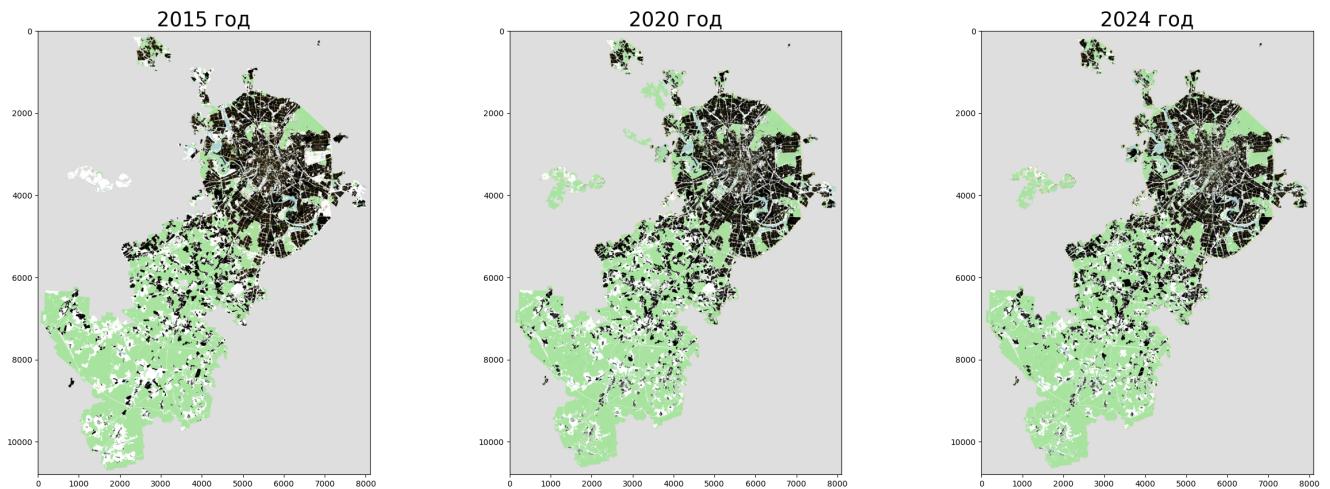


Рис. 2.6: Изображения после обработки для 1000dpi.

### 3 Анализ и построение моделей для алгоритма

#### 3.1 Обзор

В настоящее время существует не так много моделей для LULC с использованием клеточных автоматов, среди них можно выделить: Markov Chain, FLUS и SLEUTH. Рассмотрим модели далее.

FLUS (Future Land Use Simulation) основана на анализе структуры самого последнего землепользования, а не на изменениях в землепользовании в течение двух сроков. В отличие от классических клеточных автоматов, где состояние клетки определяется жесткими правилами, в FLUS состояние клетки зависит от гибких локальных правил и неопределенности. FLUS использует искусственную нейронную сеть для учета как деятельности человека, так и естественных экологических воздействий путем выявления сложных взаимосвязей между моделями землепользования и различными человеческими и природными движущими сила-

ми. Эти изменения происходят в соответствии с заданными вероятностными правилами, что создает возможность эволюции системы в неопределенном направлении.

SLEUTH (Slope, Landuse, Exclusion, Urban, Transportation и Hillshade) - это модель прогнозирования, используемая для моделирования городского роста и моделирования землепользования с использованием модели Deltatron. В SLEUTH есть четыре правила роста: правило спонтанного роста, правило роста центра распространения, правило роста периферии и правило роста, влияющих на дорогу. Все эти четыре правила роста связаны с пятью коэффициентами, называемыми коэффициентом дисперсии, коэффициентом породы, коэффициентом спреда, коэффициентом сопротивления уклона и коэффициентом гравитации дороги. Наряду с этими правилами роста в этой модели применяются вторичные «правила самоизменения», которые используются для изменения значений коэффициентов роста для достижения типичной S-кривой для темпов роста городского расширения . Правила само-модификации, такие как бум, спад, критические низкие и критические высокие константы, отвечают за нелинейное моделирование моделей городского роста. Городские клетки - это живые организмы, которые регулируются с помощью переходных правил, которые тренируются в пределах СА в виде вложенных петель.

Модель CA-Markov основывается на цепи Маркова для прогнозирования изменений землепользования. Она вычисляет величину изменений, вывод переходных матриц и клеточных автоматов; распределяет эти изменения пространственно на основе переходных матриц. Марковский процесс использует вероятности перехода для управления временной динамикой между категориями занятости. Локальные правила регулируют пространственную динамику с помощью механизма клеточных автоматов, который учитывает либо конфигурацию окрестностей, либо вероятности перехода. Этот процесс продолжается путем выполнения анализа цепи Маркова на картах LULC (Land Use and Land Cover) из обучающего датасета, что приводит к матрицам вероятности перехода и поверхностям перехода. Карты пригодности построены на основе матриц для указания прогнозируемого пространственного распределения LULC.

В данном исследовании будем использовать модель Markov Chain-Cellular Automata, чтобы спрогнозировать расширение города на 2029-2030 годы. Выбранна именно данная модель, потому что существует мало реализаций алгоритмов с клеточным автоматом, использующим цепь Макрова.

### 3.2 Модель

Создание модели клеточного автомата Маркова (САМ) для прогнозирования изменений землепользования включает в себя оценку матрицы вероятности перехода и моделирование изменений землепользования с течением времени. Цепь Маркова — последовательность случайных событий с конечным или счётным числом исходов, где вероятность наступления каждого события зависит только от состояния, достигнутого в предыдущем событии. Характеризуется тем свойством, что при текущем настоящем состоянии системы, её будущее состояние не зависит от прошлого. Модель САМ может быть математически представлена следующим образом:

$$L_{t+k} = P_{ij} \cdot L_t, \text{ для типов землеиспользования } i, j = 1, 2, \dots, n \quad (2)$$

Где  $L_t$  и  $L_{t+k}$  - карты землепользования в год  $t$  и  $t+k$  соответственно, а  $P_{ij}$  - это матрица вероятности перехода, выражающая вероятность того, что каждая ячейка изменится с типа землепользования  $i$  в год  $t$  на тип землепользования  $j$  в год  $t+k$ . Таким образом, эту матрицу можно выразить следующим образом:

$$P_{ij} = \begin{bmatrix} P_{11} & P_{12} & \dots & P_{1n} \\ P_{21} & P_{22} & \dots & P_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ P_{n1} & P_{n2} & \dots & P_{nn} \end{bmatrix}, \text{ где } 0 \leq P_{ij} \leq 1, \sum_{j=1}^n P_{ij} = 1 \quad (3)$$

Практически, матрица вероятности перехода оценивается путем «перекрестной таблицы» двух карт разных лет  $(t, t+k)$ , и она определяет вероятность перехода пикселя в класс землепользования  $i$  в другой класс  $j$  в течение этого времени. Чтобы оценить эти вероятности, нам нужно сначала узнать количество пикселей, которые со временем изменились с каждого землепользования  $i$  на другой тип  $j$ . А именно, этот процесс возвращает каждый элемент матрицы изменения (не как вероятность, т.е. не матрицу  $P_{ij}$ , а как изменения ячеек).

Имея матрицы вероятностей перехода, модель САМ может быть применена для прогнозирования будущего землепользования, как описано в уравнении [2], используя результаты, полученные для уравнения [3].  $L_t$  в уравнении [2] будет исторической картой землепользования.

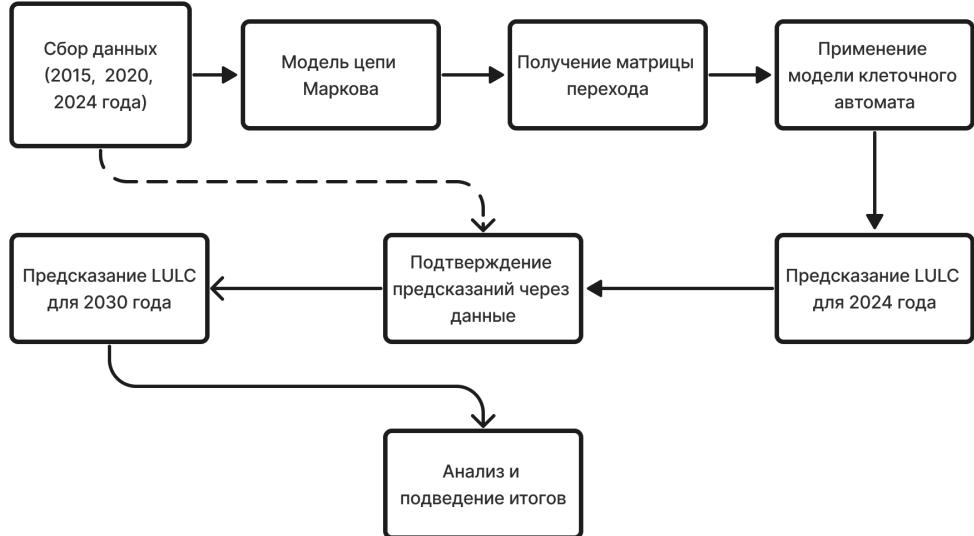


Рис. 3.1: Организация рабочего процесса, принятая для проведения исследования.

### 3.3 Реализация алгоритма

При построении модели использовалось несколько разных подходов для реализации, которые модифицируют модель клеточного автомата Маркова. Описание каждой из них представлено ниже.

#### 3.3.1 Использование мультиномиального распределения

Мультиномиальное распределение – это распределение вероятностей дискретной случайной величины, которая может принимать несколько различных значений с определенными вероятностями. Оно обобщает биномиальное распределение на случай нескольких категорий. Мультиномиальное распределение описывается формулой:

$$\mathbb{P}(\nu^n = x^n) = \frac{N!}{x_1! \cdot \dots \cdot x_n!} p_1^{x_1} \cdot \dots \cdot p_n^{x_n}, \quad \sum_{l=1}^n p_l = 1, \quad (4)$$

где  $\nu^n$ ,  $x^n$ ,  $p^n$  - векторы размера  $n$ ,  $N$ - количество испытаний,  $p^n$  - вероятности,  $n$  - количество взаимоисключающих событий.

Для  $n$  независимых испытаний, каждое из которых приводит к успеху ровно для одной из  $k$  категорий, при этом каждая категория имеет заданную фиксированную вероятность успеха, многономиальное распределение дает вероятность любой конкретной комбинации чисел успехов для различных категорий. Использование мультиномиального распределения в модели CA-Markov поможет получить нужный класс для прогнозирования.

### 3.3.2 Предлагаемый метод «Basic»

Во время обучения Basic рассчитывает «перекрестную таблицу» двух карт разных лет  $(t, t+k)$ , и строит матрицу вероятностей, разделяя каждый элемент строки на сумму элементов в строке, в итоге получает матрицу (нумерация с нуля), где в  $i$  строке  $j$  столбце описана вероятность перехода из класса  $i$  в класс  $j$  за период  $k$  (пустой класс всегда переходит в пустой). Если даны несколько карт для годов  $t, t+k, t+2k, \dots, t+nk$ , то метод рассчитывает матрицы вероятностей для карт из пар  $(t, t+k), (t+k, t+2k), \dots, (t+(n-1)k, t+nk)$ , суммирует матрицы и делит на общее количество пар, из чего получает более точную матрицу вероятностей для моделирования на  $k$  лет вперед.

Моделирование развития города происходит следующим образом:

- 1 На вход поступает изображение в виде матрицы  $data$  для года  $t$  и матрица вероятностей для моделирования на  $k$  лет вперед. Создаем пустую матрицу  $predict$  с размером матрицы  $data$ ;
- 2 Выполняем циклы по  $i$  и  $j$ , которые отвечают за строку и колонку в матрице соответственно;
- 3 Для элемента  $data[i][j]$  находим класс (в виде числа), которому он принадлежит;
- 4 Проводится эксперимент с мультиномиальным распределением [формула 4] с параметрами  $N = 1, n = 6, p^n$  - строка матрицы вероятностей с номером полученным из пункта 3. В итоге из некоторой реализации в виде вектора мы получаем индекс  $index$  (число от 0 до 5) для максимального элемента в векторе, то есть номер класса, который встречался максимальное количество раз в нашем эксперименте.
- 5 Присваиваем элементу  $predict[i][j]$  цвет для класса  $index$  из пункта 4.
- 6 Продолжаем цикл по  $i$  и  $j$  пока не обработаем всю матрицу  $data$ ;
- 7 В итоге получаем матрицу  $predict$  - прогноз на год  $t + k$ .

### 3.3.3 Предлагаемый метод «Update»

Предсказание и обучение для метода Update происходит аналгично Basic. Главное отличие в расчете матрицы вероятностей.

После того как мы получили матрицу вероятностей из обучения Basic, мы повышаем вероятность перехода класса в самого себя до 0,75 (если она меньше этого числа). Также

вероятность перехода непустого класса в пустой понижаем до нуля. Обуславливается этот прием логикой и различием в масштабах карт из датасета. Например: Москва-река располагается на одном месте на карте\_1, но на карте\_2 Москва-река расположена немного выше, из чего не следует смещение позиции реки. Таким образом, для каждой строки мы получаем число difference равное разности вероятности перехода непустого класса в пустой и вероятностью, на которую мы повысили переход класса в самого себя. В каждой строке  $i$  распределим вероятность difference между классами (кроме класса 0 и  $i$ ) пропорционально их доле. Далее проводим моделирование с новой матрицей вероятностей аналогично методу Basic.

### 3.3.4 Предлагаемый метод «Neighbors»

Для метода Neighbors была реализована вспомогательная функция GetNeighbors, которая для элемента из матрицы с изображением находила шестизначное число (сумма цифр 9), где каждый элемент отвечает за количество соседей (включая сам элемент) соответствующего класса. Работу функции можно наблюдать на рисунке [3.2]. На данном рисунке получается строка '003222', которая показывает, что у некоторого элемента соседей класса 0 было 0, класса 1 - 0, класса 2 - 3, класса 3 - 2, класса 4 - 2, класса 5 - 2.

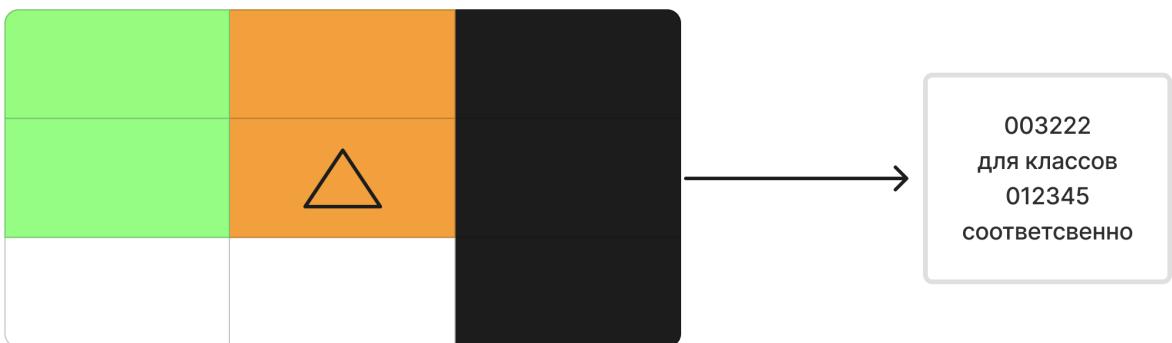


Рис. 3.2: Пример работы функции GetNeighbors для клетки.

Описание обучения в методе Neighbors:

- На вход поступает словарь dict, data\_1 - матрица с изображением для карты с годом  $t$ , data\_2 - матрица с изображением для карты с годом  $t + k$ . Ключом словаря dict является шестизначное число, а значением - массив из 6 элементов, где для индекса  $m$  хранится количество переходов данной комбинации соседей в класс  $m$ ;

2 Выполняем циклы по  $i$  и  $j$ , которые отвечают за строку и колонку в матрицах соответственно;

3 Для элемента  $\text{data\_1}[i][j]$  получаем число  $\text{num}$  из функции `GetNeighbors`;

- Если ключ  $\text{num}$  есть в словаре, то продолжаем;
- Иначе создаем массив длины 6, заполненный нулями, переходим к шагу 4;

4 Для элемента  $\text{data\_2}[i][j]$  находим класс  $c$  (в виде числа), которому он принадлежит;

5 Инкрементируем  $\text{dict}[\text{num}][c]$ ;

6 Продолжаем цикл по  $i$  и  $j$  пока не обработаем матрицы;

7 Для каждого ключа преобразовываем массив с количеством в массив вероятностей перехода  $\text{num}$  в класс, поделив каждый элемент массива на сумму элементов в массиве.

Предсказание развития города для метода в зависимости от режима `mode`:

1 На вход поступает изображение в виде матрицы  $\text{data}$  для года  $t$  и словарь  $\text{dict}$  на  $k$  лет вперед, полученный при обучении. Создаем пустую матрицу  $\text{predict}$  с размером матрицы  $\text{data}$ ;

2 Выполняем циклы по  $i$  и  $j$ , которые отвечают за строку и колонку в матрице соответственно;

3 Для элемента  $\text{data}[i][j]$  получаем число  $\text{num}$  из функции `GetNeighbors`;

- Если ключ  $\text{num}$  есть в словаре, то продолжаем;
- Иначе  $\text{predict}[i][j] = \text{data}[i][j]$ , переходим к шагу 5;

4 Элементу  $\text{predict}[i][j]$  присваивается нужный цвет в зависимости от режима для предсказаний:

- $\text{mode}=0$ . Проводится эксперимент с мультиномиальным распределением (формула 4) с параметрами  $N = 1$ ,  $n = 6$ ,  $p^n = \text{dict}[\text{num}]$ . Присваиваем элементу цвет индекса, отвечающему максимальному элементу в реализации;
- $\text{mode}=1$ . Присваиваем элементу цвет индекса, отвечающему максимальному элементу в  $\text{dict}[\text{num}]$  (класс с максимальной вероятностью);

5 Продолжаем цикл по  $i$  и  $j$  пока не обработаем всю матрицу  $\text{data}$ ;

6 В итоге получаем матрицу  $\text{predict}$  - прогноз на год  $t + k$ .

### 3.4 Обучения моделей

Обучение методов происходило на основе изображений карт за 2015, 2020 и 2024 года с разными dpi. По итогу для метода Neighbors получился 1501 ключ в словаре для 200dpi и 2001 ключ в словаре для 1000dpi. Матрицы вероятностей (МВ) для Basic и Update представлены ниже.

Таблица 3.1: МВ для Basic с 200dpi

| Классы         | 0          | 1          | 2          | 3          | 4          | 5          |
|----------------|------------|------------|------------|------------|------------|------------|
| Empty (0)      | 1          | 0          | 0          | 0          | 0          | 0          |
| Water (1)      | 0.17775849 | 0.2641028  | 0.16682537 | 0.00238367 | 0.2723295  | 0.11660019 |
| Urban (2)      | 0.09175695 | 0.02600726 | 0.6000267  | 0.00290225 | 0.21959394 | 0.05971286 |
| Road (3)       | 0.25200775 | 0.02840196 | 0.3119591  | 0.06131608 | 0.2869258  | 0.05938938 |
| Land (4)       | 0.10548466 | 0.0158468  | 0.08519843 | 0.00159151 | 0.72396505 | 0.06791352 |
| Vegetation (5) | 0.1685963  | 0.03111871 | 0.09306659 | 0.00182586 | 0.32615858 | 0.379234   |

Таблица 3.2: МВ для Basic с 1000dpi

| Классы         | 0          | 1          | 2          | 3          | 4          | 5          |
|----------------|------------|------------|------------|------------|------------|------------|
| Empty (0)      | 1          | 0          | 0          | 0          | 0          | 0          |
| Water (1)      | 0.06470516 | 0.34881398 | 0.16664949 | 0.0129631  | 0.22673595 | 0.18013231 |
| Urban (2)      | 0.05916454 | 0.0138588  | 0.64528567 | 0.03581949 | 0.09868465 | 0.14718685 |
| Road (3)       | 0.09910166 | 0.01770399 | 0.4906144  | 0.05948385 | 0.18222728 | 0.15086883 |
| Land (4)       | 0.05296428 | 0.01181774 | 0.05746873 | 0.0069345  | 0.7638397  | 0.10697501 |
| Vegetation (5) | 0.07844578 | 0.02472901 | 0.1845358  | 0.01228448 | 0.29518688 | 0.40481803 |

Таблица 3.3: МВ для Update с 200dpi

| Классы         | 0 | 1          | 2          | 3          | 4          | 5          |
|----------------|---|------------|------------|------------|------------|------------|
| Empty (0)      | 1 | 0          | 0          | 0          | 0          | 0          |
| Water (1)      | 0 | 0.75       | 0.07286732 | 0.00107543 | 0.12264213 | 0.05341512 |
| Urban (2)      | 0 | 0.02052825 | 0.75       | 0.00246418 | 0.18109617 | 0.0459114  |
| Road (3)       | 0 | 0.01054883 | 0.10484993 | 0.75       | 0.11025366 | 0.02434758 |
| Land (4)       | 0 | 0.02263064 | 0.12034386 | 0.00235879 | 0.76026505 | 0.09440167 |
| Vegetation (5) | 0 | 0.01728325 | 0.05045337 | 0.00102013 | 0.18124324 | 0.75       |

## 4 Тестирование алгоритма

### 4.1 Метрики для тестирования

Проверка прогнозируемых карт землепользования является важным шагом для оценки точности моделей и «безопасных» прогнозов в будущем. Наша задача - задача мультиклассовой классификации. Существует множество метрик, которые могут быть использованы, например:

Таблица 3.4: МВ для Update с 1000dpi

| Классы         | 0 | 1          | 2          | 3          | 4          | 5          |
|----------------|---|------------|------------|------------|------------|------------|
| Empty (0)      | 1 | 0          | 0          | 0          | 0          | 0          |
| Water (1)      | 0 | 0.75       | 0.0685221  | 0.00561784 | 0.09750548 | 0.07835458 |
| Urban (2)      | 0 | 0.01136641 | 0.75       | 0.03225249 | 0.08227168 | 0.12410942 |
| Road (3)       | 0 | 0.00521691 | 0.1460208  | 0.75       | 0.05416987 | 0.04459241 |
| Land (4)       | 0 | 0.01432067 | 0.06883924 | 0.00846593 | 0.7802783  | 0.1280958  |
| Vegetation (5) | 0 | 0.01198852 | 0.08960548 | 0.00599166 | 0.14241433 | 0.75       |

- Процент точных результатов (Accuracy)

Общая точность - это простое измерение, которое рассчитывает процент правильно классифицированных пикселей по сравнению с общим количеством пикселей. Он обеспечивает четкую и легко интерпретируемую метрику точности модели (0 = плохо, 1 = идеально). Практически это процент правильно классифицированных точек данных (пикселей, в данном случае) по сравнению с общим количеством точек данных :

$$Accuracy = \frac{\text{Number of correctly classified data points}}{\text{total number of data points}} \quad (5)$$

- Матрица ошибок для предсказаний

Матрица ошибок - это матрица чисел, которая говорит нам, где модель путается. Это классовое распределение прогнозной производительности классификационной модели, то есть матрица ошибок является организованным способом отображения прогнозов в исходные классы, к которым относятся данные.

Такие матрицы не только позволяют рассчитать точность классификатора, будь то глобальная или классовая точность, но также помогают вычислить другие важные метрики.

Матрица помогает отнести каждую клетку к некоторой величине по условию:

- TP (true positives) - классификатор верно отнёс объект к рассматриваемому классу;
- TN (true negative) - классификатор верно утверждает, что объект не принадлежит к рассматриваемому классу;
- FP (false positives) - классификатор неверно отнёс объект к рассматриваемому классу;
- FN (false negative) - классификатор неверно утверждает, что объект не принадлежит к рассматриваемому классу.

Формально, матрица ошибок (СМ) — это квадратная матрица размера  $k \times k$ , где  $CM_{i,j}$  — число объектов класса  $i$ , которые были классифицированы как класс  $j$ , а  $k$  — число классов. Значения ячеек СМ могут быть вычислены по формуле:

$$CM(y, \hat{y})_{i,j} = \sum_{k=1}^n [(y_k = i) \wedge (\hat{y}_k = j)], \quad (6)$$

где  $y$  отвечает за реальную карту, а  $\hat{y}$  за предсказанную.

В этом случае TP, TN, FP и FN считаются относительно некоторого класса  $i$  следующим образом:

$$TP_i = CM_{i,i}; \quad FP_i = \sum_{k \neq i} CM_{i,k}; \quad FN_i = \sum_{k \neq i} CM_{k,i}; \quad TN_i = ALL - TP_i - FP_i - FN_i. \quad (7)$$

Основываясь на этих данных мы можем найти несколько оценок:

- Precision (точность) — показывает долю верно классифицированных объектов среди всех объектов, которые к этому классу отнес классификатор.

$$\text{Precision}_{-i} = \frac{TP_i}{TP_i + FP_i} \quad (8)$$

- Recall (полнота) — показывает отношение верно классифицированных объектов класса к общему числу элементов этого класса.

$$\text{Recall}_{-i} = \frac{TP_i}{TP_i + FN_i} \quad (9)$$

- Specificity — показывает отношение верных срабатываний классификатора к общему числу объектов за пределами класса.

$$\text{Specificity}_{-i} = \frac{TN_i}{FP_i + TN_i} \quad (10)$$

Значения данных метрик должны стремиться к 1.

## 4.2 Проведение тестового предсказания

Тестовые предсказания проводились для 2024 года, основываясь на изображении карты за 2020 год. Полученные предсказания можно наблюдать на рисунках [4.1] и [4.2]. Легко заметить, что предсказания для изображений с 1000dpr работают намного лучше. Далее

рассматривались значения метрик для методов в зависимости от разрешения изображений, чтобы найти лучшие метод и тип изображений для предсказываний.

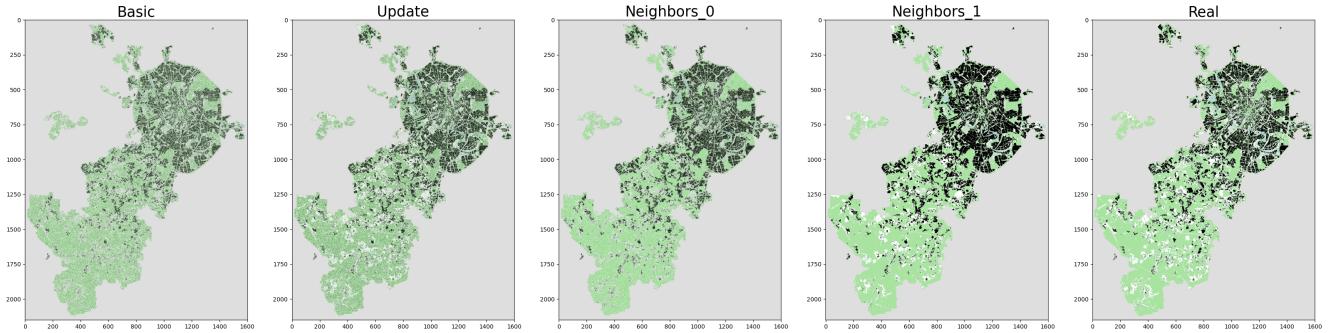


Рис. 4.1: Тестовые предсказания для изображений с 200dpi.

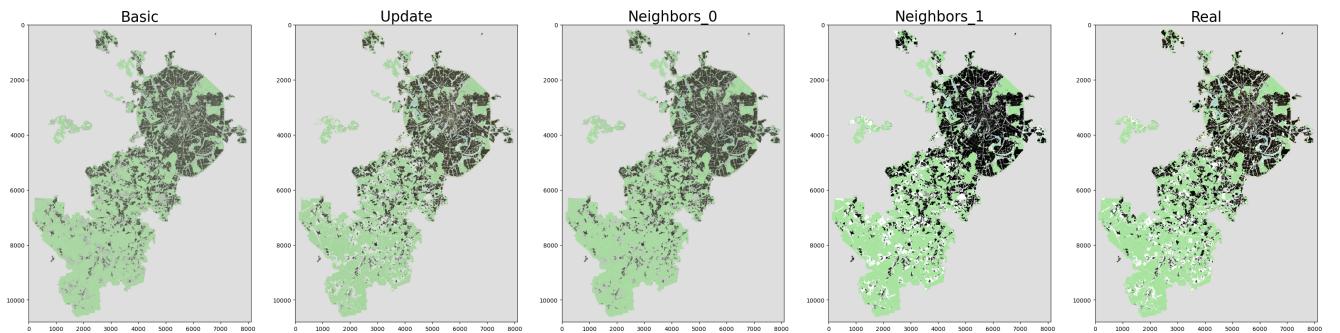


Рис. 4.2: Тестовые предсказания для изображений с 1000dpi.

#### 4.2.1 Тест метода Basic

По данным из таблиц [4.1] и [4.2] можно наблюдать, что данный метод плохо распознает дороги, водную территорию и растительность. Также для изображений с 200dpi имеет низкую точность (precision) для пустых элементов. При работе с разрешением 1000dpi показывает немного более высокую точность и более точные результаты по всем классам. Если рассматривать внешние карты ([4.1] и [4.2]), то наблюдается пониженная насыщенность по причине того, что классы Empty и Land смешались.

Таблица 4.1: Точность и матрица ошибок для Basic с 200dpi

| Accuracy       | 0.8125189655117342 |        |        |             |          |           |             |
|----------------|--------------------|--------|--------|-------------|----------|-----------|-------------|
| Классы         | TP                 | FP     | FN     | TN          | Recall   | Precision | Specificity |
| Empty (0)      | 2.57485e+06        | 654123 | 268593 | 6.83369e+06 | 0.905539 | 0.797421  | 0.912642    |
| Water (1)      | 1.01288e+07        | 83076  | 106728 | 12606       | 0.989573 | 0.991865  | 0.131749    |
| Urban (2)      | 9.25343e+06        | 337743 | 411699 | 328383      | 0.957404 | 0.964786  | 0.492974    |
| Road (3)       | 1.03102e+07        | 6852   | 14172  | 78          | 0.998627 | 0.999336  | 0.0112554   |
| Land (4)       | 7.84976e+06        | 584172 | 777030 | 1.1203e+06  | 0.909928 | 0.930735  | 0.65727     |
| Vegetation (5) | 9.60232e+06        | 270948 | 358692 | 99288       | 0.96399  | 0.972557  | 0.268175    |

Таблица 4.2: Точность и матрица ошибок для Basic с 1000dpi

| Accuracy       |             | 0.8229130557879807 |             |             |          |           |             |
|----------------|-------------|--------------------|-------------|-------------|----------|-----------|-------------|
| Классы         | TP          | FP                 | FN          | TN          | Recall   | Precision | Specificity |
| Empty (0)      | 7.7511e+07  | 9.12783e+06        | 3.8689e+06  | 1.71989e+08 | 0.952459 | 0.894645  | 0.949603    |
| Water (1)      | 2.58518e+08 | 1.64668e+06        | 1.95331e+06 | 378588      | 0.992501 | 0.993671  | 0.186932    |
| Urban (2)      | 2.28833e+08 | 9.97594e+06        | 1.18148e+07 | 1.18728e+07 | 0.950904 | 0.958226  | 0.543409    |
| Road (3)       | 2.59176e+08 | 1.41864e+06        | 1.85623e+06 | 46119       | 0.992889 | 0.994556  | 0.0314858   |
| Land (4)       | 2.07776e+08 | 1.28038e+07        | 1.46542e+07 | 2.72627e+07 | 0.934118 | 0.941954  | 0.680437    |
| Vegetation (5) | 2.34185e+08 | 1.15119e+07        | 1.23373e+07 | 4.46278e+06 | 0.949955 | 0.953146  | 0.279366    |

#### 4.2.2 Тест метода Update

Сравнивая модель с предыдущей, можно увидеть как повышение accuracy, так и общее повышение остальных метрик. Однако проблемы с предсказанием дорог, воды и растительности остаются. При работе с большим разрешением модель проявляет себя лучше. Внешне, для карты наблюдается улучшение ([4.1] и [4.2]), изображение становится более четким.

Таблица 4.3: Точность и матрица ошибок для Update с 200dpi

| Accuracy       |             | 0.829792281730009 |        |             |          |           |             |
|----------------|-------------|-------------------|--------|-------------|----------|-----------|-------------|
| Классы         | TP          | FP                | FN     | TN          | Recall   | Precision | Specificity |
| Empty (0)      | 2.90681e+06 | 322164            | 307434 | 6.79485e+06 | 0.904353 | 0.900227  | 0.954733    |
| Water (1)      | 1.00916e+07 | 120306            | 85764  | 33570       | 0.991573 | 0.988219  | 0.218163    |
| Urban (2)      | 9.1928e+06  | 398373            | 332226 | 407856      | 0.965121 | 0.958465  | 0.505881    |
| Road (3)       | 1.03032e+07 | 13758             | 13305  | 945         | 0.99871  | 0.998666  | 0.0642726   |
| Land (4)       | 7.92344e+06 | 510489            | 751068 | 1.14626e+06 | 0.913417 | 0.939472  | 0.691873    |
| Vegetation (5) | 9.4799e+06  | 393369            | 268662 | 189318      | 0.972441 | 0.960158  | 0.324905    |

Таблица 4.4: Точность и матрица ошибок для Update с 1000dpi

| Accuracy       |             | 0.8387474489536731 |             |             |          |           |             |
|----------------|-------------|--------------------|-------------|-------------|----------|-----------|-------------|
| Классы         | TP          | FP                 | FN          | TN          | Recall   | Precision | Specificity |
| Empty (0)      | 8.25508e+07 | 4.088e+06          | 4.13227e+06 | 1.71726e+08 | 0.952329 | 0.952816  | 0.976748    |
| Water (1)      | 2.58292e+08 | 1.87252e+06        | 1.54287e+06 | 789027      | 0.994062 | 0.992803  | 0.296454    |
| Urban (2)      | 2.29634e+08 | 9.17489e+06        | 1.06382e+07 | 1.30494e+07 | 0.955724 | 0.961581  | 0.587169    |
| Road (3)       | 2.58343e+08 | 2.25175e+06        | 1.76954e+06 | 132813      | 0.993197 | 0.991359  | 0.055697    |
| Land (4)       | 2.10502e+08 | 1.0078e+07         | 1.49381e+07 | 2.69788e+07 | 0.933738 | 0.954311  | 0.728039    |
| Vegetation (5) | 2.30834e+08 | 1.48631e+07        | 9.3073e+06  | 7.49279e+06 | 0.961242 | 0.939506  | 0.33516     |

#### 4.2.3 Тест метода Neighbors с mode=0

Neighbors (mode = 0) обгоняет по точности прошлые методы, но в остальных метриках остается на прежнем уровне. На изображении ([4.1] и [4.2]) можно увидеть, что класс Vegetation полностью перешел в класс Empty (ошибки в предсказании для класса Vegetation). Работа с меленьким разрешением на работу модели не влияет.

Таблица 4.5: Точность и матрица ошибок для Neighbors (mode = 0) с 200dpi

| Accuracy       |             | 0.8430136209034857 |        |             |          |           |             |
|----------------|-------------|--------------------|--------|-------------|----------|-----------|-------------|
| Классы         | TP          | FP                 | FN     | TN          | Recall   | Precision | Specificity |
| Empty (0)      | 2.90631e+06 | 322662             | 307164 | 6.79512e+06 | 0.904414 | 0.900073  | 0.954668    |
| Water (1)      | 1.01221e+07 | 89814              | 92139  | 27195       | 0.990979 | 0.991205  | 0.232418    |
| Urban (2)      | 9.26048e+06 | 330687             | 320607 | 419475      | 0.966538 | 0.965522  | 0.559179    |
| Road (3)       | 1.03075e+07 | 9501               | 13692  | 558         | 0.998673 | 0.999079  | 0.0554727   |
| Land (4)       | 7.85081e+06 | 583119             | 555543 | 1.34178e+06 | 0.933914 | 0.93086   | 0.697066    |
| Vegetation (5) | 9.58719e+06 | 286083             | 332721 | 125259      | 0.966459 | 0.971025  | 0.304513    |

Таблица 4.6: Точность и матрица ошибок для Neighbors (mode = 0) с 1000dpi

| Accuracy       |             | 0.8400119219783114 |             |             |          |           |             |
|----------------|-------------|--------------------|-------------|-------------|----------|-----------|-------------|
| Классы         | TP          | FP                 | FN          | TN          | Recall   | Precision | Specificity |
| Empty (0)      | 8.25497e+07 | 4.08911e+06        | 4.13217e+06 | 1.71726e+08 | 0.95233  | 0.952803  | 0.976742    |
| Water (1)      | 2.58479e+08 | 1.68587e+06        | 1.82478e+06 | 507117      | 0.99299  | 0.99352   | 0.231244    |
| Urban (2)      | 2.28789e+08 | 1.00197e+07        | 1.03282e+07 | 1.33594e+07 | 0.956807 | 0.958043  | 0.571425    |
| Road (3)       | 2.59047e+08 | 1.54711e+06        | 1.83599e+06 | 66363       | 0.992962 | 0.994063  | 0.0411306   |
| Land (4)       | 2.0784e+08  | 1.27396e+07        | 1.21446e+07 | 2.97723e+07 | 0.944794 | 0.942245  | 0.700329    |
| Vegetation (5) | 2.33782e+08 | 1.1915e+07         | 1.17306e+07 | 5.06948e+06 | 0.95222  | 0.951505  | 0.298477    |

#### 4.2.4 Тест метода Neighbors с mode=1

Данный метод показывает очень высокую accuracy для алгоритма. Также наблюдается сильное улучшение при предсказывании не только водных территорий, дорог и растительности, но и в общем для всех классов. Для разрешения в 1000dpi Neighbors (mode = 1) допускает ошибки всего с вероятностью 0.11. Внешне, предсказание данного метода очень похоже на фактическую карту.

Таблица 4.7: Точность и матрица ошибок для Neighbors (mode = 1) с 200dpi

| Accuracy       |             | 0.8784223946504843 |        |             |          |           |             |
|----------------|-------------|--------------------|--------|-------------|----------|-----------|-------------|
| Классы         | TP          | FP                 | FN     | TN          | Recall   | Precision | Specificity |
| Empty (0)      | 2.90675e+06 | 322221             | 307251 | 6.79503e+06 | 0.904402 | 0.900209  | 0.954727    |
| Water (1)      | 1.01976e+07 | 14313              | 84633  | 34701       | 0.991769 | 0.998598  | 0.707981    |
| Urban (2)      | 9.27646e+06 | 314712             | 143490 | 596592      | 0.984767 | 0.967187  | 0.654658    |
| Road (3)       | 1.03158e+07 | 1239               | 13848  | 402         | 0.998659 | 0.99988   | 0.244973    |
| Land (4)       | 7.97763e+06 | 456300             | 434283 | 1.46304e+06 | 0.948373 | 0.945897  | 0.762262    |
| Vegetation (5) | 9.72601e+06 | 147264             | 272544 | 185436      | 0.972742 | 0.985085  | 0.557367    |

### 4.3 Итог

Подробно рассмотрев методы, был сделан вывод, что лучше всего проявляет себя Neighbors с первым режимом предсказывания. Однако для каждой модели повышение разрешения изображения улучшает точность работы алгоритма. Таким образом, было принято

Таблица 4.8: Точность и матрица ошибок для Neighbors (mode = 1) с 1000dpi

| Accuracy       | 0.8905347622594711 |             |             |             |          |           |             |
|----------------|--------------------|-------------|-------------|-------------|----------|-----------|-------------|
| Классы         | TP                 | FP          | FN          | TN          | Recall   | Precision | Specificity |
| Empty (0)      | 8.25508e+07        | 4.08801e+06 | 4.13225e+06 | 1.71726e+08 | 0.952329 | 0.952815  | 0.976748    |
| Water (1)      | 2.59679e+08        | 485946      | 1.45041e+06 | 881490      | 0.994446 | 0.998132  | 0.64463     |
| Urban (2)      | 2.30383e+08        | 8.4257e+06  | 4.91727e+06 | 1.87703e+07 | 0.979102 | 0.964718  | 0.690187    |
| Road (3)       | 2.60567e+08        | 26886       | 1.89589e+06 | 6459        | 0.992777 | 0.999897  | 0.193702    |
| Land (4)       | 2.11785e+08        | 8.79456e+06 | 7.18011e+06 | 3.47368e+07 | 0.967209 | 0.96013   | 0.797972    |
| Vegetation (5) | 2.38783e+08        | 6.91316e+06 | 9.15833e+06 | 7.64175e+06 | 0.963063 | 0.971863  | 0.525029    |

решение использовать Neighbors (mode = 1) для моделирования развития города, прогнозируя на изображении карты с разрешением 1000dpi. Однако, стоит учитывать, что возможны ошибки при работе с некоторыми классами.

## 5 Моделирование развития города

### 5.1 Применение модели

Мы моделировали развитие города Москва, основываясь на методе Neighbors с первым режимом предсказывания. Данные для предсказания были взяты из предыдущих шагов при обучении модели. Прогнозирование проводилось для 2029-2030 годов, используя карту 2024 года с разрешением 1000dpi. Результат исследования расположен на рисунке [5.1].

### 5.2 Анализ результатов

Сравнение [5.1] общей площади территорий, занимаемых классами в прогнозируемой карте, с 2024 годом показывает, что увеличится площадь градостроений в городе за счет снижения территорий с растительностью. Но на рисунке [5.2] можно увидеть как город развивается во всех направлениях, что может предвещать начало крупных застроек в ближайшие годы. Однако, как и в тестировании модели, наблюдаются ошибки при работе с дорогами и водными территориями.

Таблица 5.1: Сравнение территорий 2024 и 2030 годов

| Years                      | Empty       | Water    | Urban       | Road    | Land        | Vegetation  |
|----------------------------|-------------|----------|-------------|---------|-------------|-------------|
| 2024                       | 5.86193e+07 | 777299   | 7.89587e+06 | 634117  | 1.39723e+07 | 5.60003e+06 |
| Change in % predicted 2030 | 4.77658e-05 | -37.2995 | 19.1196     | -98.236 | 6.42181     | -26.6802    |
|                            | 5.86193e+07 | 487370   | 9.40552e+06 | 11186   | 1.48696e+07 | 4.10593e+06 |

Предсказанный 2030 год

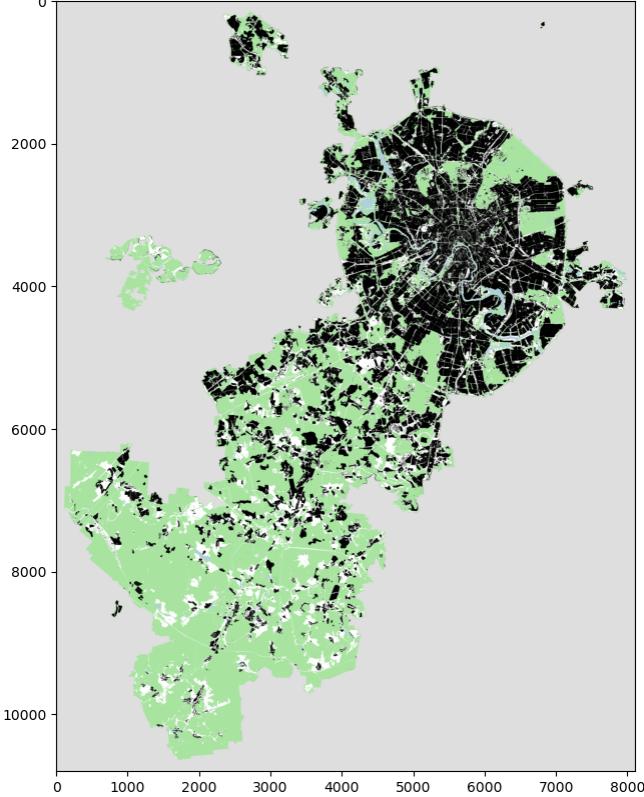


Рис. 5.1: Моделирование развития города на 2029-2030 года

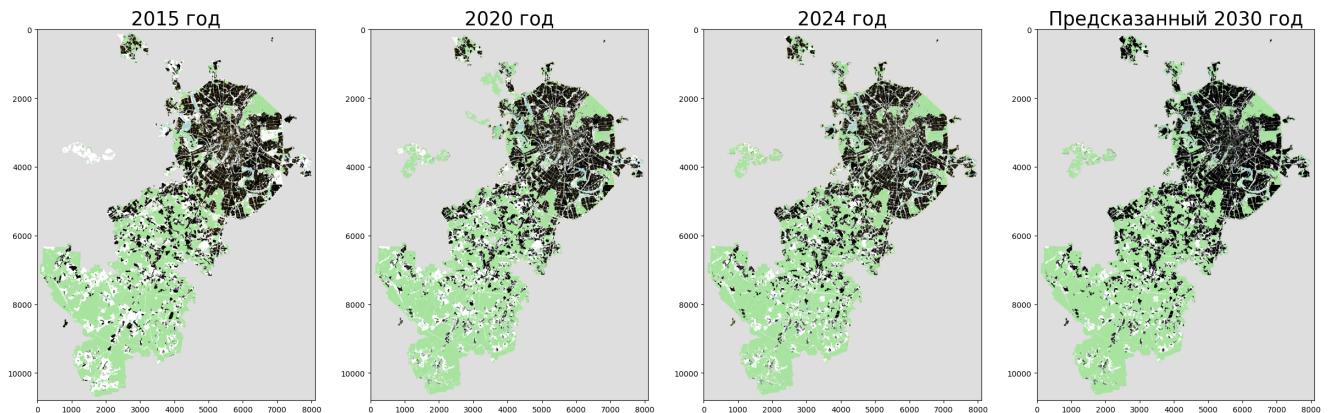


Рис. 5.2: Все карты.

## Заключение

Основное внимание в этом исследовании было уделено анализу и определению роста города. Комбинированный подход к применению векторных геоданных, инструментов ГИС и моделей клеточного автомата с использованием цепи Маркова показал свою эффективность в анализе, оценке, представлении и прогнозировании пространственных и временных моделей городского роста. Результаты исследования показывают, что в ближайшем будущем мы станем свидетелями быстрой урбанизации в рассматриваемом городе. Результаты

этого исследования могут быть полезны планировщикам и законодательным лидерам города Москвы, чтобы разработать лучшие стратегии и планирование реформ для подготовки планов городского развития для содействия устойчивому развитию. Тем не менее, следует отметить, что модели обозреваемые в данном исследовании не способны сами по себе предсказать внезапные изменения, но добавление других подходов, таких как искусственная нейронная сеть или моделирование на основе агентов, может намного улучшить качество модели. Этим исследованием еще раз доказывается, что область связанная с применением клеточных автоматов должна научно развиваться для улучшения моделей.

Ссылка на гитхаб проекта с кодом: [Modeling\\_of\\_Urban\\_Development\\_using\\_CA](#).

## Список литературы

- [1] Shikhar Deep. “Urban sprawl modeling using cellular automata”. B: *The Egyptian Journal of Remote Sensing and Space Sciences* 17 (2014), c. 179—187.
- [2] Ergo Beyene. “Modeling urban land use dynamics using Markov-chain and cellular automata in Gondar City, Northwest Ethiopia”. B: *Chinese Journal of Population, Resources and Environment* 21 (2023), c. 111—120.
- [3] Yan Liu, Michael Batty, Siqin Wang, Jonathan Corcoran. “Modelling urban change with cellular automata: Contemporary issues and future research directions”. B: *Progress in Human Geography* 45 (2019), c. 3—24.
- [4] Yisa Ginath Yuh. “Application of machine learning approaches for land cover monitoring in northern Cameroon.” B: *Ecological Informatics* 74 (2023).