МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

ИНФОРМАТИКА И КИБЕРНЕТИКА

1 (39)

Донецк – 2025

УДК 004.3+004.9+004.2+51.7+519.6+519.7

ИНФОРМАТИКА И КИБЕРНЕТИКА, № 1 (39), 2025,
Донецк, ДонНТУ.

Выпуск подготовлен по материалам, подготовленным для обсуждения на XVI Международной научно-технической конференции «Информатика, управляющие системы, математическое и компьютерное моделирование – 2025» (ИУСМКМ–2025), дата проведения 28 мая 2025 г. в рамках XI Научного форума Донецкой Народной Республики, а также результатам текущей научно-технической деятельности аспирантов, соискателей и научных работников. Статьи посвящены вопросам приоритетных направлений в области информатики, кибернетики, вычислительной техники и интеллектуальных систем.

Материалы предназначены для специалистов народного хозяйства, ученых, преподавателей, аспирантов и студентов высших учебных заведений.

Рекомендовано к печати ученым советом ФГБОУ ВО «Донецкий национальный технический университет» Министерства науки и высшего образования РФ. Протокол № 2 от 31 марта 2025 г.

Свидетельство о регистрации СМИ: серия ААА № 000145 от 20.06.2017.

Приказ МОН ДНР № 135 от 01.02.2019 о включении в Перечень рецензируемых научных изданий ВАК ДНР.

Контактный адрес редакции
РФ, ДНР, 283001, г. Донецк, ул. Артема, 58, ФГБОУ ВО «ДонНТУ»,
4-й учебный корпус, к. 36.
Тел.: +7 (856) 301-07-35, +7 (949) 334-89-11
Эл. почта: infcyb.donntu@yandex.ru
Интернет: http://infcyb.donntu.ru

СОДЕРЖАНИЕ

Информатика и вычислительная техника

Компьютерные науки

Информатика и вычислительная техника

УДК 004.415+794.8

Адаптивная оптимизация Cascaded Shadow Maps с использованием OpenCL для динамических игровых сцен

Н.В. Хомичук[*1], С.А. Зори[*2]
[*1] магистрант, Донецкий национальный технический университет,
kristoll1995@gmail.com
[*2] д.т.н., доцент, Донецкий национальный технический университет,
ik.ivt.rec@mail.ru

Аннотация
В статье представлен адаптивный метод оптимизации Cascaded Shadow Maps (CSM) с использованием технологии OpenCL для повышения качества изображений динамических игровых сцен. Подход динамически регулирует параметры CSM, обеспечивая баланс между качеством и производительностью. Анализ результатов показывает существенное улучшение эффективности рендеринга теней и визуальной детализации. В дальнейших исследованиях планируется интеграция и всесторонняя оценка прототипа непосредственно в Unreal Engine 5, усовершенствование алгоритма адаптации параметров теней, а также проведение сравнительного анализа влияния разработанного решения на различные GPU.

Введение

В современных видеоиграх реалистичная графика и плавный игровой процесс являются ключевыми факторами погружения, делая оптимизацию рендеринга критически важной задачей [1, 2]. Особую сложность представляет рендеринг теней [3] в динамических сценах, где постоянные изменения освещения и геометрии требуют адаптивных решений.

Традиционные методы часто не справляются с поддержанием стабильной производительности и высокого качества изображения. В связи с этим, актуальной задачей является разработка эффективных алгоритмов рендеринга теней, способных динамически адаптироваться к изменяющимся условиям сцены.

Целью работы является разработка и оценка метода, позволяющего достичь оптимального баланса между качеством изображения и частотой кадров в динамических игровых окружениях. Предлагаемый метод направлен на решение проблемы нестабильной производительности и артефактов рендеринга, характерных для традиционных подходов, за счет динамической регулировки параметров CSM на основе анализа характеристик сцены и текущей производительности.

Анализ существующих подходов к оптимизации CSM в контексте OpenCL

Для определения наиболее перспективных направлений разработки адаптивного алгоритма рендеринга теней для динамических игровых сцен, проведён тщательный анализ существующих методов оптимизации Cascaded Shadow Maps (CSM) [4] в контексте OpenCL [5]. Ключевая цель анализа – выявление достоинств, недостатков и ограничений существующих решений для определения пробелов в текущих подходах и формулировки требований к новому адаптивному алгоритму.

Методы фильтрации, такие как Percentage Closer Filtering (PCF) [6], Variance Shadow Maps (VSM) [7] и Exponential Shadow Maps (ESM) [8], широко используются для уменьшения артефактов, связанных с алиасингом и дискретизацией карт теней [9]. PCF реализует сглаживание границ теней путём усреднения значений глубины соседних пикселей в карте теней. VSM использует дисперсию глубины для аппроксимации затенения, создавая более мягкие тени. ESM применяет экспоненциальное преобразование глубины для уменьшения артефактов самозатенения [9].

Однако, увеличение количества выборок при использовании PCF напрямую влияет на производительность. VSM и ESM требуют аккуратной настройки параметров для предотвращения размытия или появления артефактов на границах теней. OpenCL-реализация эффективно распараллеливает процесс фильтрации на GPU, значительно снижая нагрузку на CPU.

Техники направлены на динамическое изменение разрешения карт теней для каждого каскада в зависимости от плотности геометрии и близости к камере. Идея – выделение большего разрешения областям сцены, находящимся ближе к камере или содержащим больше деталей, и использование меньшего разрешения для удаленных областей. К примеру, на рисунке 1 тени на расстоянии (A) имеют соответствующее разрешение, тогда как тени вблизи камеры (B) демонстрируют искажение перспективы [10].

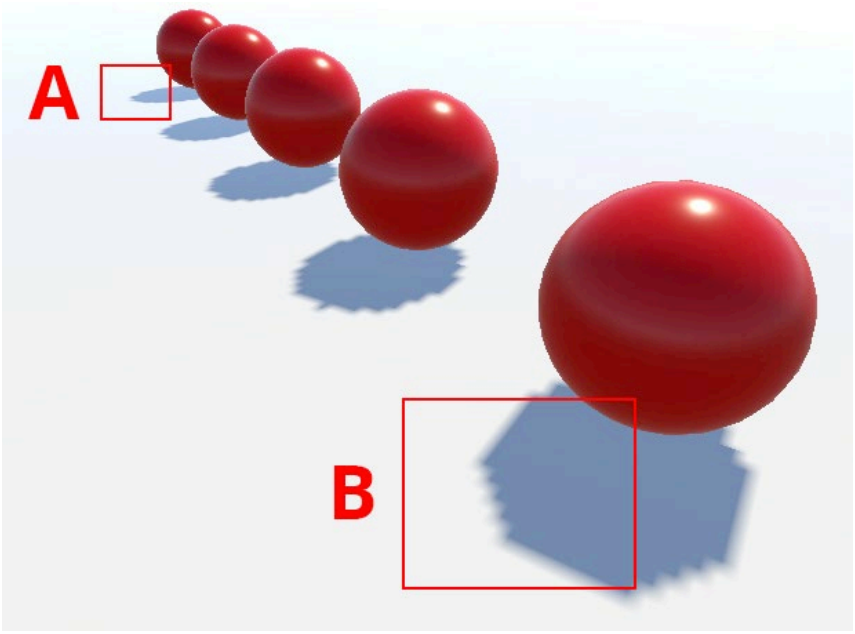

Рисунок 1 - Разрешение теней вблизи камеры и на расстоянии

Объекты, ближайшие к глазу, требуют более высокого разрешения, чем более удаленные объекты. Эффективность адаптивного разрешения каскадов зависит от точности оценки плотности геометрии и от алгоритма распределения разрешения [10]. Если настроить теневые каскады можно использовать 0, 2 или 4 каскада. Чем больше каскадов используется, тем меньше на тени влияет сглаживание перспективы [10]. Некорректная оценка приводит к неоптимальному распределению ресурсов и ухудшению качества теней. OpenCL-реализация эффективно распараллеливает процесс оценки плотности геометрии и динамически изменяет разрешение карт теней на GPU. Наложение перспективы менее заметно при использовании мягких теней и при использовании более высокого разрешения для карты теней. Однако эти решения используют больше памяти и пропускной способности при рендеринге [10].

Методы направлены на динамическую корректировку границ между каскадами CSM для точного распределения разрешения карт теней и минимизации артефактов на границах. Резкие переходы между каскадами с разным разрешением приводят к видимым артефактам. Динамическая корректировка границ смягчает эти переходы и улучшает качество теней [10]. На рисунке 2 представлена динамическая корректировка границ каскадов CSM.

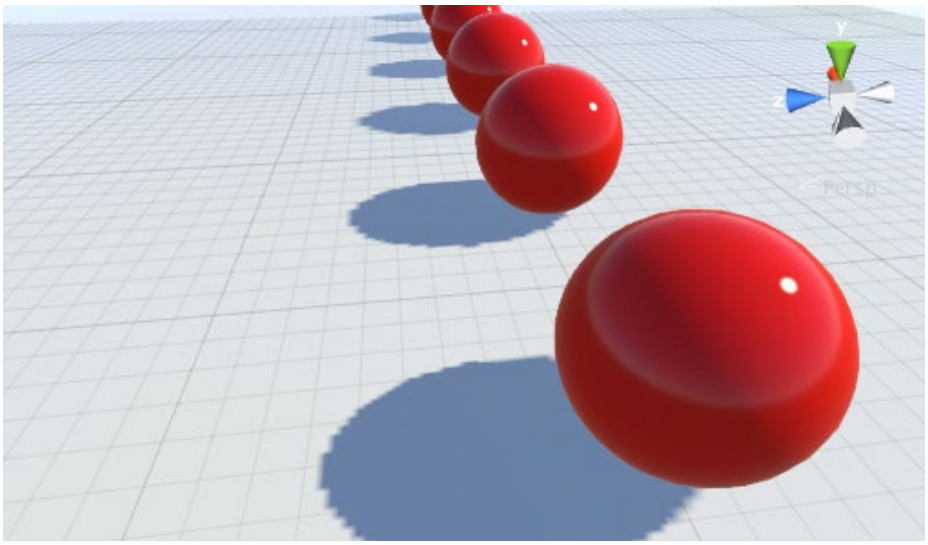

Рисунок 2 - Корректировка границ каскадов

Реализация динамической корректировки границ каскадов требует разработки эффективных алгоритмов определения оптимальных границ, учитывающих распределение геометрии и освещения в сцене [10]. Использование OpenCL позволяет реализовать такие алгоритмы на GPU, снижая нагрузку на CPU и обеспечивая плавную адаптацию границ каскадов к изменяющимся условиям сцены.

Разработка прототипа адаптивного алгоритма CSM на основе OpenCL

На основе анализа существующих подходов и выявленных ограничений, разработан прототип нового адаптивного алгоритма Cascaded Shadow Maps (CSM) для рендеринга теней в динамических игровых сценах с использованием OpenCL. Алгоритм направлен на достижение оптимального баланса между качеством изображения и производительностью за счёт динамической адаптации параметров CSM к изменяющимся условиям сцены (рис. 3 - 6).

Алгоритм начинается с анализа характеристик сцены, включающего оценку плотности геометрии, освещённости и текущей загрузки GPU. На основе этой информации определяется оптимальное количество каскадов, разрешение карт теней для каждого каскада и границы каскадов, стремясь к минимизации артефактов и обеспечению плавного перехода между уровнями детализации. Ключевой особенностью алгоритма является механизм динамической регулировки параметров CSM, основанный на обратной связи между характеристиками сцены, производительностью и качеством изображения.

Алгоритм измеряет частоту кадров (FPS), время рендеринга карт теней и загрузку GPU, а затем использует эти метрики для принятия решений об адаптации параметров CSM:

1) если FPS падает ниже заданного порога, уменьшается разрешение карт теней и/или количество каскадов;

2) если загрузка GPU низкая, увеличивается разрешение карт теней и/или количество каскадов;

3) если наблюдаются артефакты на границах каскадов, корректируются границы каскадов.

Прототип алгоритма реализован с использованием OpenCL для эффективного использования ресурсов GPU. Он включает ядра OpenCL для анализа сцены, рендеринга карт теней, фильтрации карт теней и рендеринга сцены с тенями.

Для минимизации задержек, связанных с передачей данных между CPU и GPU, используется эффективное управление памятью, включающее использование локальной и глобальной памяти GPU, а также минимизацию копирования данных между CPU и GPU с применением техник zero-copy, где это возможно.

```
#include <iostream>
#include <vector>

// Предполагаем, что у нас есть структуры данных для представления сцены, освещения, камеры и т.д.
struct Scene { /* ... */ };
struct Light { /* ... */ };
struct Camera { /* ... */ };
struct ShadowMap { /* ... */ };
struct Cascade { /* ... */ }; // Каскад содержит параметры камеры, разрешение карты теней и пр.

// И структуры для представления OpenCL context и command queue
struct CLContext { /* ... */ };
struct CLCommandQueue { /* ... */ };

// Предполагаем наличие OpenCL функций для:
// - Компиляции и запуска ядер
// - Чтения и записи данных между CPU и GPU
// - Синхронизации
extern void cl_compile_kernel(CLContext& context, const char* kernel_source, const char* kernel_name, cl_kernel& kernel);
extern void cl_launch_kernel(CLCommandQueue& queue, cl_kernel& kernel, size_t global_work_size, size_t local_work_size, /* ... */);
extern void cl_read_buffer(CLCommandQueue& queue, /* ... */);
extern void cl_write_buffer(CLCommandQueue& queue, /* ... */);

// Функции для определения характеристик сцены
float calculate_geometry_density(const Scene& scene, const Cascade& cascade);
float calculate_light_intensity(const Scene& scene, const Cascade& cascade, const Light& light);

// Функция для измерения загрузки GPU (требует специфичной для GPU реализации)
float measure_gpu_load();
float measure_fps();

// Функция для корректировки границ каскадов (нужен алгоритм определения оптимальных границ)
void adjust_cascade_boundaries(std::vector<Cascade>& cascades, const Scene& scene);
```

Рисунок 3 - Прототип адаптивного алгоритма CSM (часть 1)

```
int main() {
    // 1. Инициализация
    CLContext context;
    CLCommandQueue queue;
    cl_compile_kernel(context, "analysis_kernel.cl", "analysisKernel", analysisKernel);
    cl_compile_kernel(context, "shadow_map_kernel.cl", "shadowMapKernel", shadowMapKernel);
    cl_compile_kernel(context, "filter_kernel.cl", "filterKernel", filterKernel);
    cl_compile_kernel(context, "scene_kernel.cl", "sceneKernel", sceneKernel);

    Scene scene;
    Light light;
    Camera camera;

    int numCascades = 3;  // Начальное количество каскадов
    std::vector<Cascade> cascades(numCascades);

    float targetFPS = 60.0f;
    float minResolution = 256.0f;
    float maxResolution = 2048.0f;
    float lowGPULoadThreshold = 0.7f; // 70%

    // 2. Основной цикл рендеринга
    while (true) {
        // 2.1 Анализ характеристик сцены (на CPU, передаём на GPU для параллельной обработки)
        std::vector<float> geometryDensity(numCascades);
        std::vector<float> lightIntensity(numCascades);

        for (int i = 0; i < numCascades; ++i) {
          geometryDensity[i] = calculate_geometry_density(scene, cascades[i]);
          lightIntensity[i] = calculate_light_intensity(scene, cascades[i], light);
        }
```

Рисунок 4 - Прототип адаптивного алгоритма CSM (часть 2)

```
        //  2.2 Получение загрузки GPU (зависит от платформы и API)
        float gpuLoad = measure_gpu_load();
        float currentFPS = measure_fps();

        // 2.3 Определение оптимальных параметров CSM
        if (currentFPS < targetFPS) {
            // Производительность низкая, уменьшаем параметры
            if (cascades[0].shadowMapResolution > minResolution) {
                for (int i = 0; i < numCascades; ++i) {
                    cascades[i].shadowMapResolution /= 2.0f;
                }
            } else if (numCascades > 1) {
                numCascades--;
                cascades.resize(numCascades);
            }
        } else if (gpuLoad < lowGPULoadThreshold) {
            // GPU не загружен, увеличиваем параметры
            if (cascades[0].shadowMapResolution < maxResolution) {
                for (int i = 0; i < numCascades; ++i) {
                    cascades[i].shadowMapResolution *= 2.0f;
                }
            } else if (numCascades < 4) { // Предположим максимум 4 каскада
                numCascades++;
                cascades.resize(numCascades);
            }
        }
```

Рисунок 5 - Прототип адаптивного алгоритма CSM (часть 3)

```
    // 2.4 Корректировка границ каскадов (минимизация артефактов)
    adjust_cascade_boundaries(cascades, scene);

    // 2.5 Рендеринг карт теней (выполняется на GPU с помощью shadowMapKernel)
    std::vector<ShadowMap> shadowMaps(numCascades);
    for (int i = 0; i < numCascades; ++i) {
        // Установить параметры камеры для данного каскада
        // cl_launch_kernel(queue, shadowMapKernel, ..., scene, cascades[i], /* результаты -> shadowMaps[i]*/);
         std::cout << "Рендеринг карты теней для каскада " << i << std::endl;
    }

    // 2.6 Фильтрация карт теней (выполняется на GPU с помощью filterKernel)
     std::vector<ShadowMap> filteredShadowMaps(numCascades);
    for (int i = 0; i < numCascades; ++i) {
        // Установить параметры фильтрации для данного каскада
         //cl_launch_kernel(queue, filterKernel, ..., shadowMaps[i], /* результаты -> filteredShadowMaps[i]*/);
          std::cout << "Фильтрация карты теней для каскада " << i << std::endl;
    }

    // 2.7 Рендеринг сцены с тенями (выполняется на GPU с помощью sceneKernel)
     //cl_launch_kernel(queue, sceneKernel, ..., scene, camera, light, filteredShadowMaps, /* результаты -> frameBuffer*/);
    std::cout << "Рендеринг сцены" << std::endl;

    // 2.8 Отображение изображения на экране (зависит от движка и платформы)
    //display_framebuffer(frameBuffer);
    std::cout << "Отображение изображения" << std::endl;
}

return 0;
```

Рисунок 6 - Прототип адаптивного алгоритма CSM (часть 4)

Асинхронные операции позволяют перекрывать время ожидания на передачу данных и выполнение вычислений, дополнительно повышая производительность.

Блок-схема разработанного прототипа адаптивного алгоритма CSM на основе OpenCL представлена на рисунке 7, визуализируя последовательность этапов и взаимосвязи между ними.

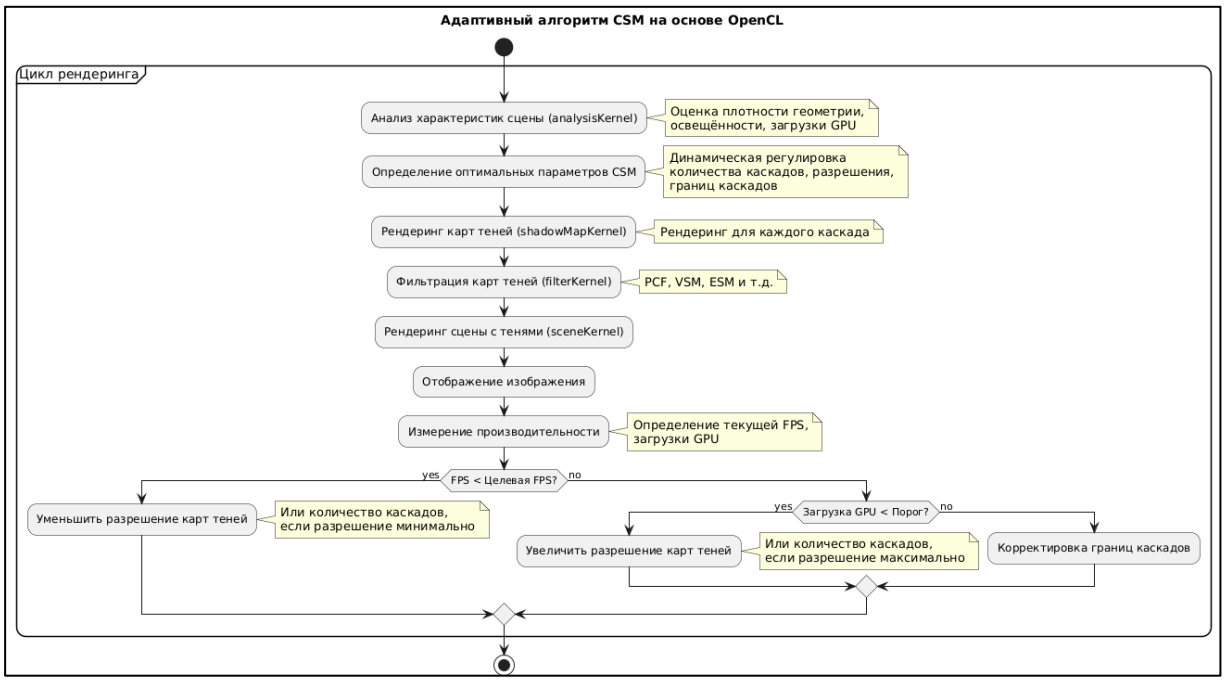

Рисунок 7 - Блок-схема разработанного прототипа адаптивного алгоритма CSM

Прогнозирование результатов внедрения прототипа в Unreal Engine 5

Внедрение разработанного адаптивного алгоритма Cascaded Shadow Maps (CSM) на основе OpenCL в Unreal Engine 5 (UE5) представляет собой перспективное направление для улучшения производительности рендеринга теней в динамических игровых сценах. Прогнозируется, что интеграция прототипа приведёт к ряду положительных результатов, связанных как с производительностью, так и с качеством изображения:

1. Снижение нагрузки на CPU. Перенос значительной части вычислений, связанных с

рендерингом теней (анализ сцены, рендеринг карт теней, фильтрация), на GPU с использованием OpenCL позволит снизить нагрузку на CPU. Это особенно важно для сложных сцен с большим количеством объектов и динамических источников света, где CPU может быть перегружен задачами, не связанными с рендерингом.

2. Улучшение частоты кадров (FPS). Сочетание снижения нагрузки на CPU и эффективного использования ресурсов GPU позволит добиться значительного улучшения частоты кадров в динамических игровых сценах. Это особенно важно для игр, требующих высокой частоты кадров для обеспечения плавного и отзывчивого игрового процесса.

3. Минимизация артефактов. Адаптивная регулировка параметров CSM (количества каскадов, разрешения карт теней, границ каскадов) позволит минимизировать артефакты, связанные с алиасингом, полосатостью и неправильным позиционированием теней.

4. Улучшение детализации теней. Использование более высокого разрешения карт теней в областях сцены с высокой детализацией позволит улучшить детализацию теней и обеспечить более реалистичное отображение мелких деталей.

5. Более плавные тени. Применение эффективных методов фильтрации карт теней (например, PCF, VSM, ESM) позволит сгладить границы теней и сделать их более плавными и реалистичными.

Заключение

Проведенное исследование позволило провести анализ возможностей OpenCL в контексте оптимизации рендеринга в игровых движках. Разработанный прототип с адаптивным алгоритмом рендеринга Cascaded Shadow Maps, оптимизированным с помощью OpenCL, представляет собой перспективное решение для повышения производительности и улучшения качества теней в динамических игровых сценах. Прогнозируется, что внедрение данного прототипа в современные игровые движки, включая Unreal Engine 5, позволит достичь увеличения скорости рендеринга, снижения нагрузки на GPU и, как следствие, более стабильной частоты кадров.

В целом, работа демонстрирует значительный потенциал OpenCL в контексте адаптивной оптимизации рендеринга теней и открывает новые возможности для создания более производительных и визуально привлекательных игр. В дальнейших исследованиях планируется интеграция и всесторонняя оценка прототипа непосредственно в Unreal Engine 5, дальнейшее усовершенствование алгоритма адаптации параметров теней, а также проведение сравнительного анализа влияния разработанного решения на различные GPU.

Литература

1. Зори, С. А. Использование средств аппаратной поддержки для повышения производительности систем 3D-пространственной визуализации / С. А. Зори, А. Я. Аноприенко, Р. В. Мальчева, О. А. Авксентьева // Информатика и кибернетика. - Донецк: ДонНТУ, 2019. - № 1 (15). - С. 5-12.

2. Хомичук, Н.В. Оптимизация производительности рендеринга в игровых движках с помощью технологии OpenCL / С. А. Зори, Н. В. Хомичук // Информатика и кибернетика. - Донецк: ДонНТУ, 2024. - № 4 (38). - С. 5-11.

3. Shadowing in 3D Graphics [Электронный ресурс] – Режим доступа: https://translated.turbopages.org/proxy_u/en-ru.ru.1f4386ae-680e64c9-23fbb5ac-74722d776562/https/www.tutorialspoint.com/computer_graphics/shadowing_in_3d_graphics.htm

4. Cascaded Shadow Maps // Learn [Электронный ресурс] – Режим доступа: https://learn.microsoft.com/en-us/windows/win32/dxtecharts/cascaded-shadow-maps OpenCL Architecture and AMD Accelerated Parallel Processing Technology [Электронный ресурс]. – Режим доступа: https://nov26.readthedocs.io/en/latest/Programming_Guides/Opencl-programming-guide.html#opencl-overview

5. OpenCL Architecture and AMD Accelerated Parallel Processing Technology [Электронный ресурс]. – Режим доступа: https://nov26.readthedocs.io/en/latest/Programming_Guides/Opencl-programming-guide.html#opencl-overview

6. Chapter 11. Shadow Map Antialiasing – 11.2 Percentage-Closer Filtering [Электронный ресурс] – Режим доступа: https://developer.nvidia.com/gpugems/gpugems/part-ii-lighting-and-shadows/chapter-11-shadow-map-antialiasing

7. Variance Shadow Maps (VSM) [Электронный ресурс] – Режим доступа: https://github.com/Delt06/toon-rp/wiki/Variance-Shadow-Maps-(VSM)

8. Тени / Майя // Руководство Вердж3Д [Электронный ресурс] – Режим доступа: https://www.soft8soft.com/docs/manual/ru/maya/Shadows.html

9. Learn OpenGL. Урок 5.3 — Карты теней // Хабр [Электронный ресурс] – Режим доступа: https://habr.com/ru/articles/353956/

10. Каскады теней // UnityHub [Электронный ресурс] – Режим доступа: https://unityhub.ru/manual/shadow-cascades

Хомичук Н.В., Зори С.А. Адаптивная оптимизация Cascaded Shadow Maps с использованием OpenCL для динамических игровых сцен. *В статье представлен адаптивный метод оптимизации Cascaded Shadow Maps (CSM) с OpenCL с использованием технологии OpenCL для повышения качества изображений динамических игровых сцен. Подход динамически регулирует параметры CSM, обеспечивая баланс между качеством и производительностью. Анализ результатов показывает существенное улучшение эффективности рендеринга теней и визуальной детализации. В дальнейших исследованиях планируется интеграция и всесторонняя оценка прототипа непосредственно в Unreal Engine 5, усовершенствование алгоритма адаптации параметров теней, а также проведение сравнительного анализа влияния разработанного решения на различные GPU.*

Ключевые слова: *OpenCL, Cascaded Shadow Maps, рендеринг, динамические сцены, адаптивные алгоритмы.*

Khomichuk N.V., Zori S.A. Adaptive optimization of Cascaded Shadow Maps using OpenCL for dynamic game scenes. *The article presents an adaptive optimization method for Cascaded Shadow Maps (CSM) with OpenCL for image quality enhancement of dynamic game scenes. The approach dynamically adjusts CSM parameters, ensuring a balance between quality and performance. The analysis of the results shows a significant improvement in the efficiency of shadow rendering and visual detail. In further research, it is planned to integrate and comprehensively evaluate the prototype directly into Unreal Engine 5, improve the algorithm for adapting shadow parameters, as well as conduct a comparative analysis of the impact of the developed solution on various GPUs.*

Key words: *OpenCL, Cascaded Shadow Maps, rendering, dynamic scenes, adaptive algorithms.*

Статья поступила в редакцию 20.01.2025
Рекомендована к публикации профессором Мальчевой Р. В.

УДК 005+65.011.56

Исследование 3D-модели матрицы полезностей для принятия решения о целесообразности деятельности интернет-провайдера

Д. А. Бишаров, И. В. Матях, Е. О. Савкова
Донецкий национальный технический университет, г. Донецк
iramatyakh@gmail.com

Аннотация
В работе предлагается представить задачу принятия решения об открытии фирмы или расширения ее сферы деятельности в виде трехмерной модели матрицы полезностей в условиях многокритериальности, а также с учетом случайных факторов, влияющих на прибыль предприятия. Рассмотрено практическое применение алгоритма решения многокритериальной задачи в условиях неопределенности на основе данной модели для организации, предоставляющей интернет-услуги населению. На основании предложенных данных построена трехмерная модель матрицы полезностей. Используя алгоритм решения задачи на основе 3D-модели, исследованы разбиения матрицы на срезы, параллельные оси случайных факторов и оси критериев. В результате каждого среза получены двумерные матрицы полезностей.

Введение

В настоящее время принятие каких-либо решений в бизнесе, на работе и в других сферах жизни неизбежно связано с определенными рисками. Они возникают в связи с тем, что современный мир очень резко меняется под влиянием различных факторов. Сейчас все больше людей стремятся открыть свое дело или же расширить сферу деятельности существующего бизнеса для получения максимально-возможной прибыли. При этом размер прибыли зависит от большого количество факторов, как внешних (например, расположение фирмы, количество конкурентов), так и внутренних (ценовая политика фирмы, спектр предоставляемых услуг).

Значительный рост убыточных предприятий позволяет говорить о том, что без предварительного прогнозирования получаемого дохода от открытия предприятия или реализации нового вида деятельности, при различных возможных ситуациях, в современном мире не обойтись.

В связи с этим можно сказать, что успешная деятельность нового предприятия или расширения сферы деятельности существующего зависит от определения внешних и внутренних факторов и оценки их влияния на прибыль.

Проблема оценки данных факторов остается актуальной на сегодняшний день. В практической деятельности получить оценку можно с помощью методов принятия решений, как однокритериальных, так и многокритериальных. При использовании однокритериальных методов получаем альтернативное решение в зависимости от одного выбранного критерия (например, максимизация прибыли). При использовании многокритериальных методов получаем альтернативное решение, учитывающие несколько критериев. Но при этом, не учитываются случайные факторы, влияющие на оценку альтернативных решений.

Для устранения данной проблемы в источнике [1] предлагается представить задачу в виде 3D-модели матрицы полезностей для принятия решения в условиях многокритериальности, а также с учетом случайных факторов. Рассмотрим применение данной модели для задачи принятия решения об открытии фирмы или расширения ее сферы деятельности на примере организации, предоставляющей интернет-услуги населению.

Целью работы является исследование возможности применения 3D-модели матрицы полезностей для решения многокритериальных задач с учетом случайных факторов, влияющих на прибыль предприятия.

Формализация задачи

Так как для данной задачи отсутствует вся необходимая информация и вероятности наступления событий неизвестны, данную задачу отнесем к задаче принятия решений в условиях неопределенности [2].

Для формализации данной задачи в условиях неопределенности необходимо выполнить следующие действия:

- Определить множество $\{Q_1, Q_2,\ldots,Q_n\}$ всех возможных внешних ситуаций, которые влияют на экономические результаты решения.

- Составить перечень $\{X_1, X_2, \ldots, X_m\}$ всех альтернативных решений, которые требуется анализировать, и для которых экономический

результат будет зависеть от реализованной «внешней» ситуации.

- Определить множество критериев, используемых для оценки альтернативных решений;
- Определить модель представления задачи;
- Определить оценки принятия X_i решения (из множества указанных выше анализируемых альтернатив) при внешней, не зависящей от ЛПР ситуации, которая соответствует событию Q_j (обозначим как a_{ij}).
- Сформировать 3D-модель матрицы полезностей.
- Выполнить сечение матрицы по различным осям для получения оптимальных альтернатив.

Для представленной таким образом задачи, из рассматриваемого множества альтернативных решений $\{X_i, i = \overline{1,m}\}$ выбрать одну альтернативу (наилучшую для ЛПР) [3].

Определение множества {Q1, Q2,…,Qn} всех возможных внешних ситуаций, которые влияют на экономические результаты решения.

На выбор того или иного альтернативного решения оказывают влияние объективные условия (случайные ситуации). Объективные условия представляются в задаче принятия решений в виде множества $Q=\{q_1,\dots,q_n\}$, одно из которых обязательно будет иметь место в действительности в момент реализации выбранного действия.

В [4] определено множество случайных факторов для данной задачи (табл. 1). Указанный набор ситуаций $\{Q_j, j=\overline{1,n}\}$ должен представлять собой полную группу событий [5]. Это означает, что одновременное наступление любых двух событий такой полной группы – невозможно и одно из событий полной группы наступит обязательно.

Таблица 1 - Множество случайных факторов, влияющих на результаты решения

Случайная ситуация	Описание
Погодные условия	Природные факторы, которые могут приводить к поломкам оборудования. Можно выделить три погодных условия, влияющих на работу интернет-провайдера: - гроза, при которой могут выйти из строя коммутаторы и кабель; - при сильном ветре, урагане повышается вероятность падения деревьев, в результате чего может быть повреждено оборудование; - благоприятные погодные условия, при которых оборудования не выходит из строя.
Сбой оборудования клиента	К оборудованию клиента относятся: роутер, компьютер, сетевой адаптер. При этом можно выделить несколько ситуаций: - сбой оборудования произошел; - сбой оборудования не произошел.
Сбой работы оборудования провайдера	К оборудованию провайдера относят коммутаторы, кабель, сервер. При выходе из строя данного оборудования, предприятие может понести большие потери. При этом можно выделить несколько ситуаций: - сбой произошел; - сбой не произошел.
Количество абонентов	От количества абонентов напрямую зависит прибыль предприятия. Выделим следующие состояния: - количество абонентов – мало; - количество абонентов – среднее количество; - количество абонентов – много.

Согласно правилу комбинаторики, при формализации соответствующей полной группы событий необходимо учесть $n_1 * n_2 * n_3 * n_4$ различных сценариев/вариантов развития событий, где n_i – количество различных вариантов изменения соответствующего i-го фактора. Так как рассмотрение случайных ситуаций во всех возможных комбинациях является очень громоздкой задачей (3 * 2 * 2 * 3 = 24 случайные ситуации), для примера рассмотрим только некоторые, наиболее вероятные из них (табл. 2).

Таблица 2 - Рассматриваемые ситуации

Ситуация	Погодные условия	Сбой оборудования клиента	Сбой работы оборудования провайдера	Кол-во абонентов
Q_1	благоприятные	не произошел	не произошел	много
Q_2	благоприятные	не произошел	не произошел	среднее
Q_3	благоприятные	не произошел	не произошел	мало
Q_4	благоприятные	произошел	не произошел	много
Q_5	благоприятные	не произошел	произошел	много

Составление перечня {X1, X2, …, Xm} альтернативных решений

Принятие решения имеет смысл, если существуют варианты альтернативных действий, число которых должно быть не менее двух. То есть необходимо определить совокупность альтернативных действий $X = \{x_1,\ldots,x_m\}$, которые может выбрать лицо, принимающее решение для достижения поставленной цели.

В данной задаче, альтернативное решение представляет собой набор услуг, который может предоставить фирма. В связи с этим в [4] определен перечень услуг, предоставляемых интернет-провайдером:

- предоставление интернет услуг (подключение интернета с различными характеристиками физическим и юридическим лицам);
- переподключение абонентов от других интернет-провайдеров;
- диагностика оборудования;
- вызов монтажной бригады;
- предоставление дополнительных услуг (выделение статического IP-адреса, регистрация дополнительного MAC-адреса и т.д.);
- предоставление цифрового телевидения.

На основе данного перечня определим возможные альтернативные решения, то есть составим комбинации услуг (по аналогии с определением случайных ситуаций). Так как возможных комбинаций достаточно много, рассмотрим основные из них:

X_1 – Предоставление интернет-услуг, переподключение абонентов от других интернет-провайдеров, диагностика/настройка оборудования;

X_2 – Предоставление интернет-услуг, переподключение абонентов от других интернет-провайдеров, диагностика/настройка оборудования, вызов монтажной бригады;

X_3 – Предоставление интернет-услуг, переподключение абонентов от других интернет-провайдеров, диагностика/настройка оборудования, вызов монтажной бригады, предоставление дополнительных услуг;

X_4 – Предоставление интернет-услуг, переподключение абонентов от других интернет-провайдеров, диагностика/настройка оборудования, вызов монтажной бригады, предоставление дополнительных услуг, предоставление цифрового телевидения

Определение множества рассматриваемых критериев

В большинстве практических задач принятия решения исходы оцениваются не по одному, а по нескольким критериям [6]. Для данной задачи определим следующее множество критериев $\{C_k, k=\overline{1,p}\}$:

C_1 – максимизация прибыли;
C_2 – минимизация расходов;
C_3 – минимизация трудозатрат.

Определение модели представления задачи

Матрица полезностей строится путем определения ожидаемых доходов (расходов) a_{ij} для случаев, когда будет принято решение X_i (из множества анализируемых альтернатив $\{X_i, i=\overline{1,m}\}$, а внешняя, не зависящая от ЛПР ситуация сложится такая, которая соответствует событию Q_j (из множества событий полной группы $\{Q_j, j=\overline{1,n}\}$, влияющей на экономический результат). В результате получаем матрицу $A=(a_{ij})$, которую в теории называют матрицей полезностей (потерь) [7]. Представим данную матрицу полезностей графически (рис. 1), где по оси X будут располагаться события из множества событий полной группы, а по оси Y – альтернативы из множества анализируемых альтернатив.

		Случайные ситуации				
		Q_1	Q_2	Q_3	…	Q_m
Альтернативы	X_1	A_{11}	A_{12}	A_{13}	…	A_{1n}
	X_2	A_{21}	A_{22}	A_{23}	…	A_{2n}
	X_3	A_{31}	A_{32}	A_{33}	…	A_{3n}
	…	…	…	…	…	…
	X_n	A_{m1}	A_{m2}	A_{m3}	…	A_{mn}

Рисунок 1 - Графическое отображение матрицы полезностей (потерь)

Элемент a_{ij} стоит на пересечении i-той строки, которая соотносится с решением X_i, и j-того столбца, который соотносится с внешней случайной ситуацией Q_j.

Представленная матрица построена для одного критерия. Чтобы построить матрицу полезностей (потерь) для нескольких критериев одновременно, добавим ось Z, к которой прикрепим множество всех необходимых критериев $C = \{Ck, k=\overline{1,p}\}$. Таким образом получаем трёхмерную матрицу полезностей (потерь) $A = \{a_{ijk}\}$, графическое отображение которой представлено на рис. 2.

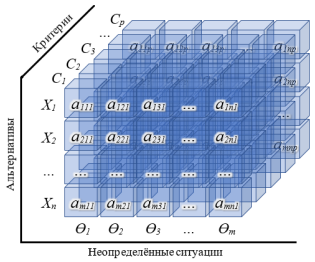

Рисунок 2 - Графическое представление трехмерной матрицы полезностей (потерь)

Формирование матриц полезностей (потерь) для выделенных критериев

Ожидаемые значения a_{ij} матрицы полезностей для критерия C_1 (получение максимальной прибыли) представлены в табл. 3.

Таблица 3 - Матрица полезностей для критерия C_1

Альтернативы	Случайные ситуации				
	Q_1	Q_2	Q_3	Q_4	Q_5
X_1	13	9	8	10	10
X_2	15	13	11	12	13
X_3	18	16	13	15	17
X_4	20	18	15	18	19

Ожидаемые значения a_{ij} матрицы потерь для критерия C_2 (минимизация расходов) представлены в табл. 4.

Таблица 4 - Матрица потерь для критерия C_2.

Альтернативы	Случайные ситуации				
	Q_1	Q_2	Q_3	Q_4	Q_5
X_1	12	10	8	13	13
X_2	14	12	11	15	15
X_3	15	13	10	19	19
X_4	16	14	12	20	20

Ожидаемые значения a_{ij} матрицы потерь для критерия C_3 (минимизация трудозатрат) представлены в табл. 5.

Таблица 5 - Матрица полезностей для критерия C_3.

Альтернативы	Случайные ситуации				
	Q_1	Q_2	Q_3	Q_4	Q_5
X_1	4000	3520	3040	5440	5600
X_2	4800	4000	3360	5440	5600
X_3	4960	4160	3520	5600	5760
X_4	5120	4320	3680	5760	5920

После заполнения матриц полезностей(потерь) можно разбить куб на срезы, перпендикулярно каждой оси. Выбор оси зависит от результата, который необходимо получить лицу, принимающему решение.

Разбиение матрицы на срезы для выбора наилучшей альтернативы по выбранному критерию

Разбиение матрицы на срезы, перпендикулярно оси критериев позволяет узнать, при выборе какой из альтернатив выбранный критерий наиболее точно выполнится вне зависимости от определенных ситуаций. Вид данного сечения представлен на рис. 3.

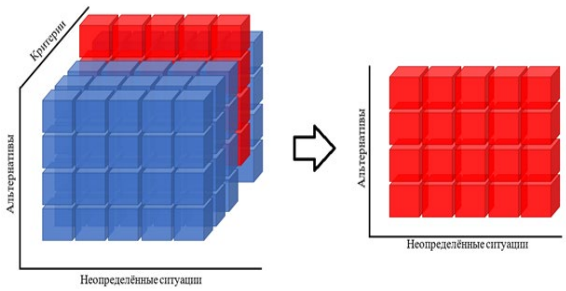

Рисунок 3 - Сечение матрицы перпендикулярно оси критериев

В результате получаем три двумерных матрицы (по количеству критериев):

Для критерия C_1:

	Q_1	Q_2	Q_3	Q_4	Q_5
X_1	13	9	8	10	10
X_2	15	13	11	12	13
X_3	18	16	13	15	17
X_4	20	18	15	18	19

Для критерия C_2:

	Q_1	Q_2	Q_3	Q_4	Q_5
X_1	12	10	8	13	13
X_2	14	12	11	15	15
X_3	15	13	10	19	19
X_4	16	14	12	20	20

Для критерия C_3:

	Q_1	Q_2	Q_3	Q_4	Q_5
X_1	25	22	19	34	35
X_2	30	25	21	34	35
X_3	31	26	22	35	36
X_4	32	27	23	36	37

Для получения наилучшей альтернативы по каждому критерию, воспользуемся критерием Гурвица (критерием пессимизма - оптимизма) [7].

Для критерия C_1 (максимизация прибыли):

Поскольку критерий стремится к максимуму, за оптимальную принимается та альтернатива, для которой выполняется соотношение: $max(S_i)$, где S_i вычисляется по формуле 1:

$$S_i=y*min(a_{ij})+(1-y)*max(a_{ij}) \quad (1)$$

В качестве значения y возьмем 0.5.

Рассчитаем S_i:

	Q_1	Q_2	Q_3	Q_4	Q_5	min	max	S_i
X_1	13	9	8	10	10	8	13	10.5
X_2	15	13	11	12	13	11	15	13
X_3	18	16	13	15	17	13	18	15.5
X_4	20	18	15	18	19	15	20	17.5

Выбираем из (10.5; 13; 15.5; 17.5) максимальный элемент 17.5, соответствующий альтернативе X_4.

Для критерия C_2 (минимизация расходов):

Поскольку необходимо минимизировать затраты, за оптимальную принимается та альтернатива, для которой выполняется соотношение: $min(S_i)$, где S_i вычисляется по формуле 1.

Рассчитаем S_i:

	Q_1	Q_2	Q_3	Q_4	Q_5	min	max	S_i
X_1	12	10	8	13	13	8	13	10.5
X_2	14	12	11	15	15	11	15	13
X_3	15	13	10	19	19	10	19	14.5
X_4	16	14	12	20	20	12	20	16

Выбираем из (10.5; 13; 14.5; 16) минимальный элемент 10.5, соответствующий альтернативе X_1.

Для критерия C_3 (минимизация количества сотрудников):

Поскольку необходимо минимизировать затраты, за оптимальную принимается та альтернатива, для которой выполняется соотношение: $min(S_i)$, где S_i вычисляется по формуле 1.

Рассчитаем S_i:

	Q_1	Q_2	Q_3	Q_4	Q_5	min	max	S_i
X_1	25	22	19	34	35	19	35	27
X_2	30	25	21	34	35	21	35	28
X_3	31	26	22	35	36	22	36	29
X_4	32	27	23	36	37	23	37	30

Выбираем из (27; 28; 29; 30) минимальный элемент 27, соответствующий альтернативе X1.

Разбиение матрицы на срезы для выбора наилучшей альтернативы для каждого случайного события

Разбиение матрицы на срезы, перпендикулярно оси неопределенных ситуаций позволяет узнать, какое из неопределенных событий наиболее благоприятно для всех критериев вне зависимости от выбора альтернатив. Вид данного сечения представлен на рис. 4.

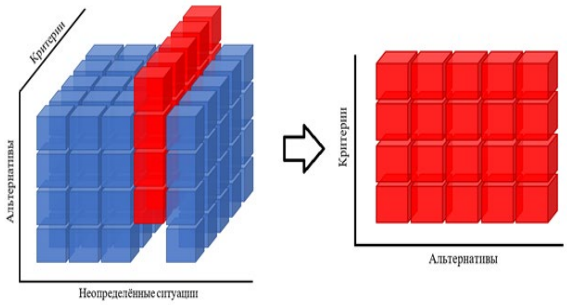

Рисунок 4 - Сечение матрицы перпендикулярно оси случайных событий.

В результате получаем 5 двумерных матриц (по количеству случайных событий):

Для события Q_1:

	C_1	C_2	C_3
X_1	13	12	4000
X_2	15	14	4800
X_3	18	15	4960
X_4	20	16	5120

Для события Q_2:

	C_1	C_2	C_3
X_1	9	10	3520
X_2	13	12	4000
X_3	16	13	4160
X_4	18	14	4320

Для события Q_3:

	C_1	C_2	C_3
X_1	8	8	3040
X_2	11	11	3360
X_3	13	10	3520
X_4	15	12	3680

Для события Q_4:

	C_1	C_2	C_3
X_1	10	13	5440
X_2	12	15	5440
X_3	15	19	5600
X_4	18	20	5760

Для события Q_5:

	C_1	C_2	C_3
X_1	10	13	5600
X_2	13	15	5600
X_3	17	19	5760
X_4	19	20	5920

В данном случае оценки по критериям даны в разных шкалах измерений (стоимость, количество сотрудников), следовательно, они несоизмеримы между собой. В связи с этим, перед началом решения задачи необходимо провести нормирование оценок [9]. Под нормированием понимается приведение всех критериев к единому масштабу и безразмерному виду. Выполним нормализацию критериев по формулам 2,3:

для критериев, которые стремятся к max:

$$Ci'(X)=\frac{Ci(X)}{Ci\ max} \quad (2)$$

для критериев, которые стремятся к min:

$$Ci'(X)=\frac{Cimin}{Ci(X)} \quad (3)$$

В результате получаем следующие матрицы:

Для события Q_1:

	C_1	C_2	C_3
X_1	0,65	1,00	1,00
X_2	0,75	0,86	0,83
X_3	0,90	0,80	0,81
X_4	1,00	0,75	0,78

Для события Q_2:

	C_1	C_2	C_3
X_1	0,50	1,00	1,00
X_2	0,72	0,83	0,88
X_3	0,89	0,77	0,85
X_4	1,00	0,71	0,81

Для события Q_3:

	C_1	C_2	C_3
X_1	0,53	1,00	1,00
X_2	0,73	0,73	0,90
X_3	0,87	0,80	0,86
X_4	1,00	0,67	0,83

Для события Q_4:

	C_1	C_2	C_3
X_1	0,56	1,00	1,00
X_2	0,67	0,87	1,00
X_3	0,83	0,68	0,97
X_4	1,00	0,65	0,94

Для события Q_5:

	C_1	C_2	C_3
X_1	0,53	1,00	1,00
X_2	0,68	0,87	1,00
X_3	0,89	0,68	0,97
X_4	1,00	0,65	0,95

Для получения наилучшей альтернативы по каждому событию, воспользуемся методом целевого программирования [10]. Название этой группы методов связано с тем, что ЛПР задает определенные цели $\bar{f}_1, \bar{f}_2, \ldots, \bar{f}_k$ для каждого критерия. Задача многокритериальной оптимизации преобразуется в задачу минимизации суммы отклонений с некоторым показателем p и значение вычисляется по формуле 4:

$$z=\left|\sum_{k=1}^{k} w_k\left|f_k(x)-\bar{f}_k\right|^p\right|^{1/p} \rightarrow \min \quad (4)$$

где w_k - некоторые весовые коэффициенты, характеризующие важность того или иного критерия.

Задачу можно конкретизировать в зависимости от значений параметра p и заданных целей. В частности, при $p=2$ и $w_k=1$ получим задачу минимизации суммы квадратов отклонений (формула 5):

$$z=\sqrt{\sum_{k=1}^{k}\left|f_k(x)-\bar{f}_k^*\right|^2} \rightarrow \min, \quad (5)$$

в которой минимизируется евклидово расстояние от множества достижимости f до «абсолютного максимума» $f^*=(f_1^*,f_2^*,\ldots,f_l^*)$ в пространстве критериев. Здесь $f_k^*=\max\limits_{x\in D} f_k(x)$. Для нормированных матриц $f_k^*=1$.

Найдем наилучшие альтернативы для каждого случайного события:

Для события Q_1:

$Z_1=\sqrt{|0.65-1|^2+|1-1|^2+|1-1|^2}=0.35$

$Z_2=\sqrt{|0.75-1|^2+|0.86-1|^2+|0.83-1|^2}=0.33$

$Z_3=\sqrt{|0.9-1|^2+|0.8-1|^2+|0.81-1|^2}=0.3$

$Z_4=\sqrt{|1-1|^2+|0.75-1|^2+|0.78-1|^2}=0.33$

Минимальное значение Z = 0.3, соответствующее альтернативе X_3.

Для события Q_2:

$Z_1=\sqrt{|0.5-1|^2+|1-1|^2+|1-1|^2}=0.5$

$Z_2=\sqrt{|0.72-1|^2+|0.83-1|^2+|0.88-1|^2}=0.35$

$Z_3=\sqrt{|0.89-1|^2+|0.77-1|^2+|0.85-1|^2}=0.3$

$Z_4=\sqrt{|1-1|^2+|0.71-1|^2+|0.81-1|^2}=0.34$

Минимальное значение Z = 0.3, соответствующее альтернативе X_3.

Для события Q_3:

$Z_1=\sqrt{|0.53-1|^2+|1-1|^2+|1-1|^2}=0.47$

$Z_2=\sqrt{|0.73-1|^2+|0.73-1|^2+|0.9-1|^2}=0.39$

$Z_3=\sqrt{|0.87-1|^2+|0.8-1|^2+|0.86-1|^2}=0.28$

$Z_4=\sqrt{|1-1|^2+|0.67-1|^2+|0.83-1|^2}=0.38$

Минимальное значение Z = 0.28, соответствующее альтернативе X_3.

Для события Q_4:

$Z_1=\sqrt{|0.56-1|^2+|1-1|^2+|1-1|^2}=0.44$

$Z_2=\sqrt{|0.67-1|^2+|0.87-1|^2+|1-1|^2}=0.36$

$Z_3=\sqrt{|0.83-1|^2+|0.68-1|^2+|0.97-1|^2}=0.36$

$Z_4=\sqrt{|1-1|^2+|0.65-1|^2+|0.94-1|^2}=0.35$

Минимальное значение Z = 0.35, соответствующее альтернативе X_4.

Для события Q_5:

$Z_1=\sqrt{|0.53-1|^2+|1-1|^2+|1-1|^2}=0.47$

$Z_2=\sqrt{|0.68-1|^2 + |0.87-1|^2 + |1-1|^2} = 0.34$
$Z_3=\sqrt{|0.89-1|^2 + |0.68-1|^2 + |0.97-1|^2} = 0.33$
$Z_4=\sqrt{|1-1|^2 + |0.65-1|^2 + |0.95-1|^2} = 0.35$

Минимальное значение Z = 0.33, соответствующее альтернативе X_3.

Анализ полученных результатов решения

В результате сечения матрицы по двум осям получаем следующие результаты решения, представленные в таблицах 6-7:

Таблица 6 - Лучшие альтернативы для каждого случайного события

Случайное событие		Лучшая альтернатива
Q_1		X_3
Q_2		X_3
Q_3		X_3
Q_4		X_4
Q_5		X_3

Таблица 7 - Лучшие альтернативы для каждого критерия

Критерий	Лучшая альтернатива
C_1	X_4
C_2	X_1
C_3	X_1

Для выбора наилучшей альтернативы предлагается использовать следующие подходы:

- В качестве наилучшего решения выбрать ту альтернативу, которая встречается наибольшее количество раз. Преимуществом данного подхода является простота выбора наилучшей альтернативы. Недостатком – при получении абсолютно разных результатов по различным сечениям невозможно выбрать одну альтернативу.

- В случае, если для каждого сечения получаем абсолютно разные результаты, необходимо построить еще одну задачу принятия решения на основании полученных данных. В результате решения новой задачи, получим наилучшее альтернативное решение. Преимуществом данного подхода является получение наилучшей альтернативы при любых обстоятельствах. Недостатком - сложность формирования новой задачи принятия решения, а именно заполнение новой матрицы полезностей/потерь.

В статье не рассмотрено практическое применение сечения по альтернативам, которое помогает выбрать наилучшую альтернативу по выбранным критериям вне зависимости от того, какое случайное событие наступит. Данное сечение можно применить для заполнения пустых ячеек в новой матрице полезностей/потерь при использовании подхода 2. Это предположение будет рассмотрено в будущих исследованиях.

Выводы

В работе представлена задача принятия решения об открытии фирмы или расширения ее сферы деятельности в виде трехмерной модели матрицы полезностей для принятия решения в условиях многокритериальности, а также с учетом случайных факторов, влияющих на результаты решения. Рассмотрено практическое применение алгоритма решения многокритериальной задачи в условиях неопределенности на основе данной модели для организации, предоставляющей интернет-услуги.

На основании предложенных данных построена трехмерная модель матрицы. Используя алгоритм решения задачи на основе 3D-модели, исследованы разбиения матрицы на срезы, перпендикулярные оси критериев и случайных ситуаций. В результате исследования получены альтернативные решения как для каждого случайного события, так и для каждого рассматриваемого критерия.

Применив данную методику, можно выделить следующие преимущества:

- возможность посмотреть на задачу одновременно с нескольких сторон;

- возможность учитывать несколько целей, которые ставит ЛПР при нахождении наилучшего решения.

Недостатком данной методики является громоздкость вычислений для получения оптимального решения. Однако, данный недостаток не является существенным по сравнению с преимуществами методики.

Вопросом дальнейшего исследования является формирования новой задачи принятия решения на основании полученных данных (множества альтернатив по критериям ислучайным событиям) для получения одного наилучшего решения.

Литература

1. Савкова, Е. О. 3D-модель матрицы полезностей для принятия решения в условиях многокритериальности / Е. О. Савкова, И. В. Матях, А. С. Милая // Сборник научных трудов Донецкого института железнодорожного транспорта, 2018. - № 51. - С. 22-29.

2. Теория принятия решений в 2 т. Том 1 : учебник и практикум для вузов / В. Г. Халин [и др.] ; под редакцией В. Г. Халина. — Москва : Издательство Юрайт, 2022. — 250 с. — (Высшее образование). — ISBN 978-5-534-03486-8.

3. Оразбаев, Б. Б. Теория и методы системного анализа: учебное пособие / Б. Б. Оразбаев, Л. Т. Курмангазиева, Ш. К. Коданова. – М.:

Издательский дом Академии Естествознания, 2017. – 248 с.

4. Матях, И. В. Исследование проблемы выбора минимального набора услуг для расширения деятельности предприятия на примере интернет-провайдера / Е. О. Савкова, И. В. Матях, О. В. Ченгарь // Информатика и кибернетика, 2017. – № 3(9). - С. 94-99.

5. Милая, А. С., Савкова Е. О. Формализация задачи принятия решения об оптимизации работы по обслуживанию клиентов в условиях неопределённости // Электронные информационные системы, 2017. - №1(12). - С. 603-614.

6. Подиновский, В. В. Идеи и методы теории важности критериев в многокритериальных задачах принятия решений / В. В. Подиновский. – М. : Наука, 2019. – 103 с. – ISBN 978-5-02-040241-6

7. Бродецкий, Г. Л. Системный анализ в логистике: выбор в условиях неопределенности: учебник для вузов / Г. Л. Бродецкий. - М.: ИЦ «Академия», 2010. - 336 с.

8. Черноруцкий, И. Г. Методы принятия решений: учебное пособие / И. Г. Черноруцкий. – СПб.: БХВ-Петербург, 2005. — 416 с.

9. Афоничкин, А. И., Михаленко Д. Г. Управленческие решения в экономических системах: учебник для вузов. – СПб.: Питер, 2009. – 480 с.

10. Системы поддержки принятия решений: учебник и практикум для бакалавриата и магистратуры / под ред. Халина В. Г., Черновой Г. В. - М.: Издательство Юрайт, 2015. - 494 с. - Серия: Бакалавр и магистр. Академический курс.

Бишаров Д.А., Матях И.В., Савкова Е.О. Исследование 3D-модели матрицы полезностей для принятия решения о целесообразности деятельности интернет-провайдера. *В работе предлагается представить задачу принятия решения об открытии фирмы или расширения ее сферы деятельности в виде трехмерной модели матрицы полезностей в условиях многокритериальности, а также с учетом случайных факторов, влияющих на прибыль предприятия. Рассмотрено практическое применение алгоритма решения многокритериальной задачи в условиях неопределенности на основе данной модели для организации, предоставляющей интернет-услуги населению. На основании предложенных данных построена трехмерная модель матрицы полезностей. Используя алгоритм решения задачи на основе 3D-модели, исследованы разбиения матрицы на срезы, параллельные оси случайных факторов и оси критериев. В результате каждого среза получены двумерные матрицы полезностей.*

Ключевые слова: *многокритериальная задача, случайные факторы, альтернативные решения, 3D-матрица, сечение матрицы.*

Bisharov D.A., Matyakh I.V., Savkova E.O. Investigation of the 3D model of the utility matrix for making a decision on the expediency of an Internet provider. *The paper proposes to present the problem of making a decision on opening a company or expanding its field of activity in the form of a three-dimensional model of the utility matrix in a multi-criteria environment, as well as taking into account random factors affecting the company's profits. The practical application of an algorithm for solving a multi-criteria problem in conditions of uncertainty based on this model for an organization providing Internet services to the public is considered. Based on the proposed data, a three-dimensional model of the utility matrix is constructed. Using an algorithm for solving the problem based on a 3D model, the partitions of the matrix into slices parallel to the axes of random factors and the axes of criteria are investigated. As a result of each slice, two-dimensional utility matrices are obtained.*

Keywords: *multi-criteria problem, random factors, alternative solutions, 3D mater*

Статья поступила в редакцию 21.02.2025
Рекомендована к публикации профессором Скобцовым Ю. А.

УДК № 004.42

Чистый код как концепция развития программного обеспечения: теория и применение в компонентно-ориентированном программировании

М. Ю. Павлов, А. В. Боднар
Донецкий национальный технический университет, г. Донецк
кафедра программной инженерии
E-mail: vigototheroad@mail.ru

Аннотация:
Статья рассматривает понятие чистого кода и его роль в разработке ПО. Анализируются отличия чистого и «грязного» кода и их влияние на проектирование. Особое внимание уделено компонентно-ориентированному программированию как способу эффективного применения принципов чистого кода. Подход к чистому коду как к философии развития программного продукта позволяет достичь оптимального баланса между качеством архитектуры и практическими требованиями разработки.

Введение

Что делает код «чистым»? В программировании часто обсуждают принципы хорошего стиля, архитектурной чистоты и читаемости кода. Но где проходит грань между чистым и грязным кодом? Можно ли считать плохо структурированный, но работающий код действительно «грязным»?

В своей книге «Чистый код» Роберт Мартин утверждает: «Умение писать чистый код – тяжёлый труд. Оно не ограничивается знанием паттернов и принципов. Над кодом необходимо попотеть. Необходимо пытаться и терпеть неудачи» [1].

Так что же такое «чистый» код и каким критериям он должен соответствовать? Код, который не соответствует эти критериям, автоматически становится плохим? Важно понимать, что в реальных условиях программирования иногда приходится отклоняться от полного соответствия стандартам в угоду скорости разработки, требованиям бизнеса или ограничениям ресурсов.

В этой статье рассматривается, чем отличается чистый код от «грязного», в каких ситуациях «грязный» код может быть оправдан и почему фанатичное следование правилам может привести к обратному эффекту.

Чистый Код

Стоит уточнить, что само выражение «чистый код» имеет множество смыслов и трактовок, но в рамках данной работы будет выведено общее определение, которое наилучшим образом отражает его идею. В книге Роберта Мартина нет чёткого определения – вместо этого автор описывает отношение ведущих разработчиков к чистому коду и то, как, по их мнению, он должен выглядеть.

Ниже приведены некоторые цитаты из данной книги [1]:

«Вы работаете с чистым кодом, если каждая функция делает примерно то, что вы ожидали. Код можно назвать красивым, если у вас также создается впечатление, что язык был создан специально для этой задачи», – Уорд Каннингем.

«Я мог бы перечислить все признаки, присущие чистому коду, но существует один важнейший признак, из которого следуют все остальные. Чистый код всегда выглядит так, словно его автор над ним тщательно потрудился», – Майкл Физерс [1].

«Сокращение дублирования, высокая выразительность и раннее построение простых абстракций. Все это составляет чистый код в моем понимании», – Рон Джеффрис [1].

Самая лаконичная и при том информативная цитата принадлежит Бьёрну Страуструпу:

«Чистый код – это код, который легко читать и улучшать» [1].

Про способы написания чистого кода, изложенные в книге, Роберт Мартин говорит следующее:

«Мы излагаем свои мнения в виде беспрекословных истин и не извиняемся за свою категоричность. Для нас, на данном моменте наших карьер, они являются беспрекословными истинами» [1].

Хотя Роберт Мартин не даёт прямого определения чистому коду, сама книга описывает критерии, которым следуют как сам Мартин, так и разработчики, которых он приводит в пример.

Тем самым он описывает следующие характеристики чистого кода:

- понятность;
- лаконичность;
- логичность;
- работа на одном уровне абстракции;
- модульность;
- легкость в поддерживании;
- тестируемость.

Эти принципы прослеживаются в конце книги, в списке эвристических правил и «запахов» кода. Под «запахами» подразумеваются опыт и интуиция программиста, говорящие ему о том, что с кодом что-то не так. Поскольку эвристические правила зависят от конкретного программиста и его видения кода, то их существует огромное множество.

Важным является то, что вышеописанные идеи нашли отражение во многих подходах к разработке программного обеспечения. К примеру, в объектно-ориентированном программировании (ООП) принципы инкапсуляции и полиморфизма помогают писать расширяемый код, при этом не разрушая изначальную структуру и создавая управляемую конструкцию. В компонентно-ориентированном программировании (КОП) акцент делается на гибкость модулей. В нём реализация выходит за рамки конкретного компонента, позволяя взаимодействовать с любыми объектами, имеющими определенный модуль, через «системный» компонент, что так же поддерживает ранее описываемую идею.

Все вышеописанное позволяет вывести следующее определение чистого кода:

Чистый код – это не просто стиль написания, а разумный подход к разработке. Это код, который выполняет свою задачу и не мешает улучшению продукта.

Грязный код

Грязный код является противоположностью чистого. Если идти от обратного, то у такого кода должна отсутствовать хотя бы одна из характеристик чистого кода. Возникает вопрос: достаточно ли отсутствия одного критерия, чтобы код считался грязным, или он должен не соответствовать всем критериям одновременно? Как упоминалось в книге, отсутствие определенных качеств в коде может привести к проблемам. Например, неправильное именование переменной вызывает вопросы о её предназначении. Это и есть первопричина появления «не чистого» кода.

Дать прямое определение «грязному» коду довольно сложно. Можно ли считать весь код, который не соответствует критериям чистого, грязным? Сложности с определением связаны с отсутствием чистых критериев. Однако, здесь возможны два подхода.

Первый подход заключается в отрицании чистого кода: любой код, не соответствующий его критериям, автоматически считается грязным.

Второй подход предполагает описание собственного понимания грязного кода и определение границы, за которой «грязь» начинает вредить.

Грязный код как отрицание чистого кода

Со знанием критериев чистого кода становится возможным сформулировать то, чем является «грязный» код:

- неочевидность (использование непонятных имен переменных и функций, слишком длинные или вложенные конструкции, затрудняющие восприятие, применение «магических чисел» вместо осмысленных констант);
- избыточность (дублирование кода, написание избыточных проверок, которые уже предусмотрены языком, использование ненужных переменных и классов, усложняющих структуру);
- хаотичность (отсутствие четкой структуры, использование разных стилей кодирования в одном файле, изменение переменных в разных частях кода, что затрудняет отслеживание их состояния);
- смешение абстракций (одновременная работа с абстрактными и конкретными деталями);
- монолитность (весь код сосредоточен в одном огромном классе без разделения на модули, один метод выполняет несколько задач вместо их делегирования, отсутствие четких границ ответственности между частями кода);
- хрупкость (любое изменение ломает другие части системы, код зависит от множества несвязанных модулей, функции и классы разрастаются, содержат десятки параметров);
- плохая тестируемость (многие функции имеют неочевидные входные и выходные данные или выполняют дополнительные действия, влияющие не только на результат).

Если хотя бы один из этих пунктов применим к коду, его можно считать «грязным».

Однако, имеется нюанс: код не всегда бывает идеальным. Даже с учетом приведенного определения сложно представить код, который бы соответствовал всем требованиям одновременно.

Например, оптимизированный алгоритм может выглядеть запутанно, но он необходим для повышения производительности. Или в коде могут быть реализованы сложные взаимодействия, но при этом входные и выходные данные сами по себе просты.

Таким образом, в рамках этого подхода практически любой код можно считать «грязным».

Когда грязный код действительно вреден

Когда код можно считать «грязным»? Логично предположить, что решать проблему стоит тогда, когда она действительно начинает негативно влиять на разработку.

Когда код является приемлемым?

- он локализован и не разрастается по всей системе;
- он выполняет простую логику без неоднозначных выводов;
- его легко адаптировать;
- он не создает лишние зависимости;
- он оправдан скоростью работы или разработки;
- он временный и будет исправлен позже.

Когда грязный код начинает вредить?

- его сложно понять и реализовать;
- он провоцирует цепную реакцию багов при изменении;
- он требует изменение базовой логики при дальнейшей разработке;
- его операции ведут к неявным входным и выходным данным.

Таким образом, граница между неидеальным, но рабочим кодом и по-настоящему вредным кодом проходит там, где он начинает мешать развитию и поддержке продукта. Если код замедляет разработку, вносит хаос и увеличивает вероятность ошибок – это уже не компромисс, а проблема.

Второй подход кажется более практичным, особенно в условиях реальной разработки. Он несет в себе простую идею: грязный код становится проблемой только тогда, когда начинает негативно влиять на проект. Конечно, в некоторых случаях частные правила важнее глобальных, и попытка внедрить идеальные стандарты там, где они не нужны, может только навредить, запутывая логику и усложняя ранее выполненную работу.

Использование чистого кода в КОП

Понимание принципов чистого кода позволяет эффективно применять их в компонентно-ориентированном подходе. В данном разделе рассматривается, как принципы чистого кода используются в этой парадигме, каким образом их соблюдение упрощает разработку, а также какие базовые особенности написания кода в рамках КОП необходимо учитывать. Слепое заимствование практик чистого кода оказывается недостаточным – их требуется адаптировать к специфике КОП и особенностям разработки для обеспечения эффективности и гибкости создаваемых систем.

КОП активно опирается на принцип «композиция вместо наследования». Вместо построения сложных иерархий классов, где поведение объектов определяется через наследование, КОП формирует объекты путём комбинирования независимых компонентов, каждый из которых отвечает за отдельную функциональность. Это повышает гибкость системы, позволяя без значительных затрат добавлять, удалять или изменять поведение объектов. Наследование, напротив, часто приводит к жёстким связям между классами, что затрудняет модификацию системы. Например, в RPG-системе, если персонаж наследует базовый класс Character, а затем создаются подклассы Warrior, Mage и другие, изменение базового класса способно нарушить корректную работу всех производных классов. Композиция позволяет динамически добавлять компоненты по мере необходимости, что значительно упрощает настройку объектов.

Ещё одной важной характеристикой КОП является возможность добавления, удаления или изменения компонентов во время выполнения программы. Это обеспечивает адаптивность системы к изменяющимся условиям без необходимости модификации исходного кода или перекомпиляции проекта. Например, в RPG-проекте персонаж может получить новую способность в процессе выполнения квеста, что повышает интерактивность игрового мира.

КОП часто применяет системы, координирующие взаимодействие групп компонентов. Вместо прямого взаимодействия между компонентами данные обрабатываются системами, которые обновляют их состояние. Такой подход характерен для архитектуры Entity-Component-System (ECS), одной из популярных реализаций КОП. В ECS компоненты представляют собой структуры данных, а системы реализуют прикладную логику, что снижает связанность и способствует повышению производительности.

КОП также поощряет декларативный подход, при котором поведение и свойства компонентов определяются через данные, а не через жёстко закодированную логику. Это реализуется посредством сериализации данных, например через форматы JSON, XML или настройки редакторов игровых движков, таких как Unity. Такой подход упрощает процесс настройки компонентов дизайнерами без необходимости внесения изменений в код. Несмотря на возможное противоречие с принципом самодокументируемости кода, в Unity использование атрибутов, таких как [SerializeField], позволяет создавать наглядные интерфейсы редактора и задавать связи между компонентами без нарушения читаемости.

Компоненты в КОП должны быть максимально автономными, то есть их поведение не должно зависеть от состояния других компонентов, за исключением явно

определённых взаимодействий, например, через события или системы. Автономность обеспечивает переиспользуемость компонентов в различных контекстах без необходимости их модификации. Например, компонент HealthComponent может использоваться как для игроков, так и для NPC или разрушаемых объектов.

Принцип читаемости ярко выражен в КОП, поскольку каждый компонент отвечает за строго определённую функциональность. Названия компонентов и их методов должны быть интуитивно понятными. Например, вместо абстрактного Component1 предпочтительнее использовать наименования вроде PlayerMovementComponent или InventoryComponent, что упрощает восприятие кода. Лаконичность достигается за счёт того, что каждый компонент описывает только необходимые характеристики, используемые системами для реализации функциональности, минимизируя избыточный код и повышая управляемость компонентов.

Принцип единой ответственности в КОП реализуется элегантно: каждый компонент выполняет только одну функцию. Например, в RPG-системе HealthComponent управляет здоровьем персонажа, а InventoryComponent – его предметами. Это предотвращает создание так называемых «божественных объектов», объединяющих множество функций, что увеличивает сложность системы и снижает её гибкость. Зависимости между компонентами минимизируются посредством обмена данными через события, сообщения или интерфейсы, а не через прямые ссылки. Это снижает связанность и повышает модульность системы. Например, компонент атаки может генерировать событие OnAttack, обрабатываемое компонентом здоровья без знания о его внутреннем устройстве.

Для устранения дублирования кода в КОП применяются базовые компоненты или утилитные классы. Например, если несколько компонентов используют событийную модель, целесообразно создать общий EventManager, который централизованно управляет событиями, устраняя необходимость повторного написания однотипного кода.

Чистый код в КОП: синтез авторских подходов

Понимание чистого кода в компонентно-ориентированном программировании не может ограничиваться только практическими приёмами. Для полноценного применения этой концепции необходимо учитывать теоретические и методологические подходы, выработанные ведущими исследователями в области программной инженерии. В данной главе рассмотрены взгляды авторов, оказавших значительное влияние на формирование современной культуры разработки, а также проведена интерпретация их идей в контексте КОП.

В книге «Совершенный код» С. Макконнелл подчёркивает, что читаемость, локализация изменений и снижение когнитивной нагрузки – ключевые признаки качественного кода [2]. Эти принципы непосредственно применимы к КОП, где каждый компонент реализует узкоспециализированную функцию, а значит, должен быть предельно понятным и изолированным.

М. Фаулер в книге «Рефакторинг» рассматривает архитектурную чистоту как результат регулярной реорганизации кода без изменения поведения [3]. Для КОП это особенно важно, так как позволяет модифицировать поведение системы через изменение отдельных компонентов без затрагивания остальной архитектуры.

Д. Кнут рассматривает программирование как форму выражения мысли [4]. В КОП, где компоненты должны быть переиспользуемыми и инкапсулированными, ясность их интерфейсов и поведения становится критическим фактором. Концепция чистого кода как выразительного средства полностью применима к описанию поведения компонентов и систем.

Авторы «Паттернов проектирования» подчеркивают важность слабой связанности и высоко уровня абстракции [5]. Эти характеристики органично вписываются в КОП, где вместо иерархий классов применяются независимые модули, взаимодействующие через общие интерфейсы или события.

К. Бек в рамках экстремального программирования настаивает на простоте реализации и приоритете тестируемости [6]. Для КОП, где поведение системы определяется взаимодействием множества компонентов, возможность изолированного тестирования каждого из них – необходимое условие надёжности.

В «Программисте-прагматике» Томас и Хант подчёркивают значение повторного использования, отсутствия дублирования и локализации логики [7]. Эти требования совпадают с архитектурной философией КОП, в рамках которой компоненты должны быть автономными и легко заменяемыми.

М. Физерс в книге «Работа с унаследованным кодом» описывает стратегии постепенного улучшения сложных систем без полной их переработки [8]. Это напрямую применимо к КОП, так как замена или модификация конкретных компонентов позволяет эволюционно адаптировать систему к новым требованиям.

Анализ подходов признанных экспертов позволяет сделать вывод, что многие принципы чистого кода – читаемость, модульность, слабая связанность, тестируемость и минимализм – естественным образом интегрируются в компонентно-ориентированную архитектуру. Эти авторские идеи не только дополняют теоретическую базу КОП, но и обосновывают его эффективность с точки зрения инженерной практики. Таким образом, чистый код и КОП не просто совместимы – они взаимно усиливают друг друга в процессе построения гибких и сопровождаемых программных систем.

Когда используется Чистый Код

Здесь необходимо обратиться к парадигмам программирования и их предназначению. Если кратко, парадигмы программирования – это концепции, определяющие стиль написания кода в соответствии с определёнными принципами. Они помогают достичь оптимальных результатов в конкретных условиях. Многие парадигмы включают принципы и методологии, которые способствуют повышению эффективности разработки. Одним из ключевых преимуществ ООП является возможность структурировать программу с помощью классов, каждый из которых отвечает за свою область. Однако это может стать и недостатком: чрезмерное усложнение архитектуры ведёт к увеличению связей и затруднению сопровождения кода.

Используя данный подход, важно учитывать последствия архитектурных решений. Следует задуматься о глубине иерархии: не приведёт ли она к чрезмерному усложнению структуры кода? Не сделает ли избыточная иерархия базовый класс хрупким? И, конечно, не приведёт ли слепое следование парадигме к ненужному усложнению кода?

Важно понимать, что, когда разработчик пишет код, он пишет его не для компилятора, а для других людей. Когда возникает потребность в написании чистого кода? В первую очередь, каждый программист должен следить за тем, что он пишет. Когда он замечает проблему, он должен её решить. Роберт Мартин называет это «запахами кода» – ситуациями, когда разработчик интуитивно понимает, где может возникнуть проблема [1].

И здесь кроется главная особенность книги: «чистый код» – это не просто свод правил, а концепция развития кода. Это инструмент, который может как улучшить, так и ухудшить ситуацию, или философия, которая влияет на разработку, позволяя программисту использовать её не как цель, а как возможность улучшить состояние проекта.

Если относиться к чистому коду не как к набору строгих правил, а как к идее о том, что код стоит чистить и улучшать, это позволит будущим разработчикам не тратить время на попытки разобраться в том, что делает этот код и зачем он нужен. Это, в свою очередь, облегчит развитие и поддержку кода. Именно поэтому в начале книги делается большой акцент на опыте других разработчиков. Большинство правил, приведённых в книге, используются на подсознательном уровне, когда программист понимает, с какими проблемами он может столкнуться, возвращаясь к коду, или где части кода могут быть слишком запутанными, как неопытное описание автора. Но опыт и стремление сделать свой код лучше – это лучшее лекарство от таких проблем. Анализ того, почему определённый кусок кода кажется непонятным или запутанным, позволяет взглянуть на код с другой стороны, как на искусство, порождающее понимание как в себе, так и в других людях. Именно к этому отсылает фраза из начала книги:

«А как мне написать чистый код?» Бесполезно пытаться написать чистый код, если вы не знаете, что это такое! – Боб Мартин [1].

И самый большой плюс такого подхода – это простота разработки. Когда код начинает получаться простым, с понятной и проработанной структурой, не возникает вопросов о том, как получить доступ к переменной или где выделить функции для реализации определённой фичи так, чтобы это было органично. А когда приходится дополнять такой код, ранее разработанная структура начинает работать на разработчика, а не разработчик пытается втиснуть новый код в уже перегруженную систему.

Поэтому многие находят отражение программирования в разных сферах: кто-то – в кулинарии, как Кнут, кто-то – в искусстве, как Мартин [4]. И действительно, чем больше разработчик рассматривает другие области, тем больше решений он может найти для своих задач, что побуждает к развитию и изучению нового.

Ни архитектура, ни чистый код не требуют от нас совершенства – просто будьте честны и делайте все, что можете – Боб Мартин [1].

Выводы

В данной статье проведён комплексный анализ понятия чистого кода, его характеристик и противоположности – грязного кода. Показано, что понятие чистого кода выходит за рамки формальных правил и является важным аспектом культуры программирования, ориентированным на повышение читаемости, сопровождаемости и адаптивности программных решений.

Особое внимание уделено интеграции принципов чистого кода в компонентно-ориентированное программирование. Продемонстрировано, что КОП благодаря свойственным ему свойствам композиции, слабой связности и автономности компонентов,

позволяет наиболее естественным образом применять идеи чистого кода в архитектуре программных систем.

Отмечено, что эффективное использование принципов чистого кода требует осознанной адаптации к контексту конкретной задачи, а не механического следования установленным рекомендациям. Подход к чистому коду как к философии развития программного продукта позволяет достичь оптимального баланса между качеством архитектуры и практическими требованиями разработки.

Перспективы дальнейших исследований связаны с изучением применения принципов чистого кода в других парадигмах программирования, а также с разработкой методик автоматизированного анализа качества кода в рамках КОП.

Литература

1. Мартин, Р. Чистый код: создание, анализ и рефакторинг. Библиотека программиста / Р. Мартин. - СПб.: Питер. 2019. – 452 с.

2. Макконнелл, С. Совершенный код: мастер-класс по разработке программного обеспечения / С. Макконнелл. – СПб.: Питер, 2017. – 896 с.

3.Фаулер, М. Рефакторинг. Улучшение проекта существующего кода/ М. Фаулер. Пер. с англ. - СПб: Символ-Плюс, 2003. - 432 с.

4. Кнут, Д. Искусство программирования. Том 1: Основные алгоритмы / Д. Кнут. — М.: Вильямс, 2011. — 784 с.

5. Гамма, Э. Приёмы объектно-ориентированного проектирования. Паттерны проектирования / Э. Гамма, Р. Хелм, Р. Джонсон, Влиссидес. – СПб.: Питер, 2016. – 368 с.

6. Бек, К. Экстремальное программирование: разработка через ценности / К. Бек. — СПб.: Питер, 2017. — 240 с.

7. Томас, Д. Программист-прагматик: путь от подмастерья к мастеру / Д. Томас, Э. Хант. — СПб. Диалектика, 2020. — 352 с.

8. Физерс М. Работа с унаследованным кодом: как улучшить старый код / М. Физерс. — М.:Вильямс, 2009. — 400 с.

9. Heinman, G. T. Component-based software engineering : putting the pieces together / G. T. Heinman, W. T. Concill. - Boston : Addison-Wesley, 2001. – 818 с.

10. Тепляков, С. Паттерны проектирования на платформе .NET / С. Тепляков. – СПБ: Питер, 2015. - 320 с.

11. Эванс, Э. Предметно-ориентированное проектирование (DDD): структуризация сложных программных систем / Э. Эванс. - Москва : 000 "И.Д. Вильямс", 2011. - 448 с.

12. Медведев, В. И. NET компонентно – ориентированное программирование / В. И. Медведев. - 2-е издание – Казань: Республиканский центр мониторинга качества образования, 2013. – 248 с.

13. Кулямин, В. В. Компонентный подход в программировании/ В. В. Кулямин. - 2-е издание - М.: НОУ "Интуит" 2016. – 590 с.

Павлов М. Ю., Боднар А. В. Чистый код как концепция развития программного обеспечения: теория и применение в компонентно-ориентированном программировании. *Статья рассматривает понятие чистого кода и его роль в разработке ПО. Анализируются отличия чистого и "грязного" кода и их влияние на проектирование. Особое внимание уделено компонентно-ориентированному программированию как способу эффективного применения принципов чистого кода. Подход к чистому коду как к философии развития программного продукта позволяет достичь оптимального баланса между качеством архитектуры и практическими требованиями разработки.*

Ключевые слова*: чистый код, грязный код, компонентно-ориентированное программирование, композиция, архитектура ПО, сопровождаемость.*

Pavlov M., Bodnar A. Clean Code as a Software Development Concept: Theory and Practice in Component-Oriented Programming. *The article examines the concept of clean code and its role in software development. It analyzes the differences between clean and "dirty" code and their impact on design. Special attention is given to component-oriented programming as an effective way to apply clean code principles. The approach to clean code as a software product development philosophy allows achieving an optimal balance between the quality of architecture and the practical requirements of development.*

Keywords*: clean code, dirty code, component-oriented programming, composition, software architecture, maintainability.*

Статья поступила в редакцию 25.02.2025
Рекомендована к публикации профессором Зори С. А.

УДК 004.93

Свёрточные нейронные сети в системах обнаружения и распознавания лиц

В. В. Кочетуров, О. И. Федяев
Донецкий национальный технический университет
кафедра программной инженерии им. Л.П. Фельдмана
E-mail: v.v_kocheturov@mail.ru

Аннотация:
В статье рассматривается применение свёрточных нейронных сетей на различных этапах систем распознавания лиц: обнаружение, выравнивание и распознавание. Обозреваются ключевые архитектуры и методы, применяемые на каждом из этапов. Описываются многозадачные CNN, выполняющие несколько функций одновременно. Выделяются современные тенденции развития в этой области. Описываются многозадачные CNN, выполняющие несколько функций одновременно. Выделяются современные тенденции развития: применение механизмов внимания, трансформеров и обучение на малых данных.

Введение

Системы распознавания лиц стали неотъемлемой частью многих современных технологий, от биометрической аутентификации [1] до анализа видеопотоков в системах безопасности. В основе успеха этих систем лежат свёрточные нейронные сети (CNN), которые продемонстрировали выдающуюся способность к извлечению информативных признаков из изображений.

Процесс распознавания лиц является многоэтапным, и для каждого этапа существуют специализированные или адаптированные архитектуры CNN. Понимание особенностей этих архитектур и методов их применения критически важно для разработки эффективных и робастных систем. В данной статье рассматриваются современные CNN, используемые на этапах обнаружения, выравнивания, извлечения признаков и принятия решения, а также архитектуры, объединяющие некоторые из этих этапов.

Обнаружение лиц с использованием CNN

Обнаружение лиц – первый и один из важнейших этапов, заключающийся в локализации всех лиц на изображении или в видеопотоке. Детекторы лиц должны быть устойчивы к изменениям масштаба, ракурса, освещения, частичным окклюзиям и выражениям лиц. Высокая скорость работы также является критическим требованием для многих приложений.

Современные CNN-детекторы можно условно разделить на два основных типа (рис. 1).

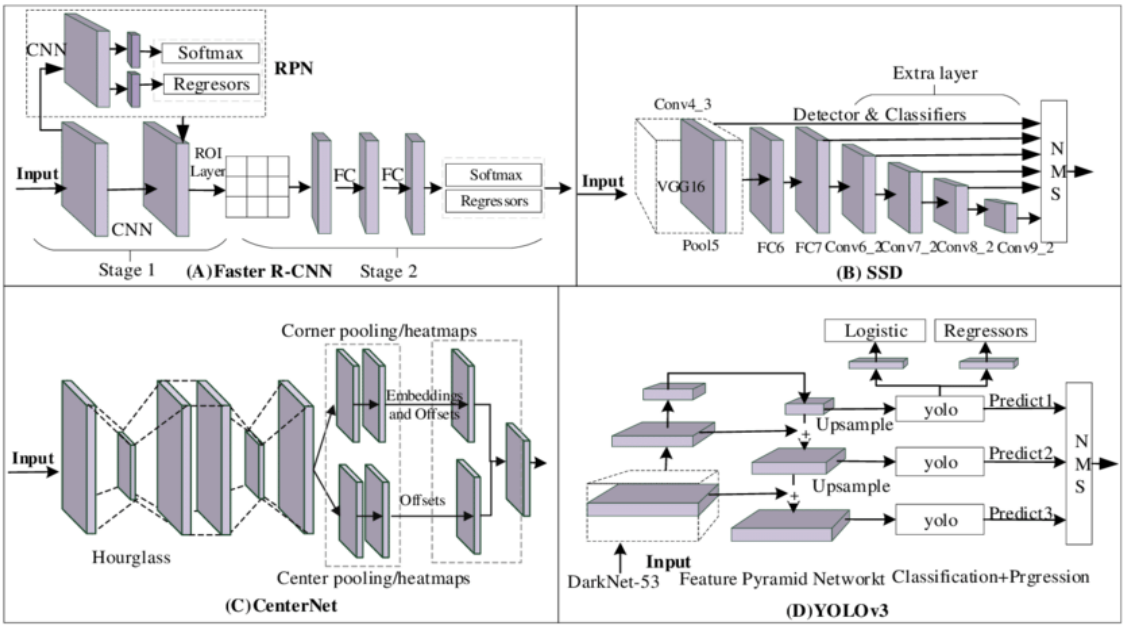

Рисунок 1 – Архитектуры CNN-детекторов (Faster R-CNN, SSD, CenterNet, YOLOv3)

Двухэтапные, такие как Faster R-CNN, сначала генерируют регионы-кандидаты, а затем классифицируют их [2]. Они обычно более точные, но медленнее. Одноэтапные, например, YOLO и SSD, предсказывают ограничивающие рамки и классы объектов за один проход по сети [3]. Они, как правило, быстрее, что делает их предпочтительными для задач реального времени. К одноэтапным также относится CenterNet, который использует подход "объекты как точки": вместо рамок он определяет центры объектов на тепловой карте и регрессирует их размеры. Это позволяет достичь хорошего баланса между точностью и скоростью, особенно в условиях отсутствия якорей.

Ключевыми архитектурами CNN для обнаружения являются:

– SSD (Single Shot MultiBox Detector) и YOLO (You Only Look Once): хотя это детекторы общего назначения, они успешно адаптируются для обнаружения лиц путем обучения на соответствующих датасетах. Они используют предсказания на картах признаков разного масштаба для детекции объектов разного размера [3].

– MTCNN (Multi-task Cascaded Convolutional Networks): Каскад из трёх сетей (P-Net, R-Net, O-Net), где первые сети (P-Net, R-Net) выполняют грубую детекцию и отсев кандидатов [4]. Подробнее будет рассмотрен в разделе интегрированных архитектур.

– RetinaFace: Одноэтапный детектор, часто использующий ResNet или MobileNet в качестве базовой сети (backbone) и Feature Pyramid Network (FPN) для эффективной работы с лицами разных масштабов [5].

Для увеличения поля обзора без увеличения числа параметров и сохранения разрешения карт признаков в детекторах лиц активно применяются расширенные свёртки (atrous/dilated convolutions). Это позволяет сетям захватывать более широкий контекст, что особенно важно для различения лиц на сложном фоне или для анализа связей между частями лица. Например, в RetinaFace контекстные модули, построенные на расширенных свёртках, помогают улучшить качество детекции.

В задачах детекции обычно используются комбинации функций потерь: для классификации (лицо/не лицо) – кросс-энтропия или Focal Loss (эффективна при дисбалансе классов), для регрессии координат рамок – Smooth L1 loss.

Выравнивание лиц с помощью CNN

После обнаружения лицо необходимо выровнять – привести к каноническому виду путем детектирования и нормализации по ключевым точкам. Эти точки, такие как углы глаз, кончик носа, углы рта, контуры бровей и подбородка, служат ориентирами для геометрической трансформации. Цель выравнивания – не просто найти точки, а использовать их для геометрической трансформации (например, аффинной или подобия) исходного изображения лица. Это приводит лицо к стандартному масштабу, ориентации и положению, где, например, глаза находятся на определенной горизонтальной линии, а расстояние между ними фиксировано. Такой канонический вид минимизирует вариации, не связанные с идентичностью, такие как поза и масштаб.

Выравнивание значительно снижает вариативность входных данных для последующей сети извлечения признаков, что напрямую повышает точность и робастность распознавания. Без выравнивания даже небольшие изменения в ракурсе или масштабе лица могут привести к значительным изменениям в векторе признаков, затрудняя сравнение.

Количество детектируемых ключевых точек варьируется в зависимости от задачи и необходимой точности:

– 5 точек: обычно это центры глаз, кончик носа и углы рта. Этого достаточно для базового выравнивания (например, в MTCNN или RetinaFace для последующей грубой нормализации).

– 68 точек: Стандарт, часто используемый в академических исследованиях (например, датасет iBUG 300-W), позволяет более точно смоделировать контур лица, форму глаз, бровей и губ.

– 98 точек и более: обеспечивают еще более детальное представление, включая контуры зрачков, век, более плотные точки на губах и контуре лица.

Большинство современных методов выравнивания основаны на регрессии координат ключевых точек с помощью CNN. Существуют два основных подхода [6]:

1. Прямая регрессия координат: Сеть напрямую предсказывает (x, y) координаты для каждой из N ключевых точек. В качестве функции потерь здесь обычно используются L1-норма (Mean Absolute Error) или L2-норма (Mean Squared Error), либо их комбинации, такие как Smooth L1 loss, которая менее чувствительна к выбросам, чем L2, и имеет более стабильные градиенты для малых ошибок, чем L1.

2.Регрессия тепловых карт: Сеть предсказывает N тепловых карт, по одной для каждой ключевой точки. Каждая тепловая карта представляет собой 2D распределение вероятностей, где пик яркости (высокое значение) указывает на наиболее вероятное положение соответствующей ключевой точки. Координаты точки затем извлекаются как

положение максимума на тепловой карте, часто с субпиксельной точностью (например, путем интерполяции или подгонки функции Гаусса). Этот подход часто более робастен и точен, особенно для сложных случаев, так как он неявно кодирует пространственную информацию и структуру вокруг точки и позволяет сети обучаться с более гладкими градиентами. Функции потерь для тепловых карт обычно основаны на MSE между предсказанной и эталонной (сгенерированной, например, 2D функцией Гаусса вокруг истинной координаты) тепловой картой. Продвинутые функции потерь, такие как Wing Loss и Adaptive Wing Loss, были предложены для улучшения точности, особенно для точек с большими отклонениями, уделяя больше внимания ошибкам на границах объектов и уменьшая чувствительность к малым ошибкам вблизи центра.

Архитектурно это могут быть как отдельные CNN, так и модули внутри более крупных сетей. Например, O-Net в MTCNN или специальная ветка в RetinaFace одновременно с детекцией предсказывают 5 ключевых точек. Это эффективно, так как признаки, извлеченные для детекции, частично полезны и для локализации точек. Для предсказания большого числа точек (68+) часто используются более глубокие и сложные архитектуры. Популярны архитектуры на основе модификаций ResNet или Hourglass Networks. Hourglass Networks, благодаря своей многомасштабной архитектуре с последовательными этапами понижения и повышения разрешения и пропусками соединений, эффективно агрегируют как глобальные контекстные, так и локальные высокодетализированные признаки, что критично для точной локализации всех точек. Другие архитектуры, такие как High-Resolution Networks (HRNet), стремятся поддерживать представление с высоким разрешением на протяжении всей сети, объединяя параллельные сверточные потоки разного разрешения, что также способствует высокой точности локализации.

В некоторых продвинутых системах выравнивание может также включать оценку 3D-положения головы (углы Эйлера: рыскание, тангаж, крен) по 2D-изображению с помощью CNN. Эти параметры затем могут использоваться для выполнения более сложной 3D-нормализации лица, проецируя его на фронтальный вид или используя их как дополнительные признаки.

Распознавание лиц: извлечение признаков, идентификация и верификация

Это ключевой этап, где выровненное изображение лица преобразуется в компактный и дискриминативный вектор признаков, который затем используется для идентификации или верификации личности.

Основная цель извлечения признаков – получить вектор, который будет дискриминативным (векторы признаков разных людей должны быть хорошо разделимы в пространстве признаков) и инвариантным (векторы признаков одного и того же человека должны быть близки, несмотря на изменения освещения, выражения лица, возраста и других неидентификационных факторов).

Для извлечения признаков часто используются следующие базовые архитектуры CNN:

– VGGNet: использовалась в ранних системах (например, VGG-16 (см. рис. 2)), но сейчас уступила место более глубоким архитектурам.

– ResNet (Residual Networks): является де-факто стандартом благодаря возможности обучать очень глубокие сети (ResNet-50 (см. рис. 3), ResNet-100 и глубже).

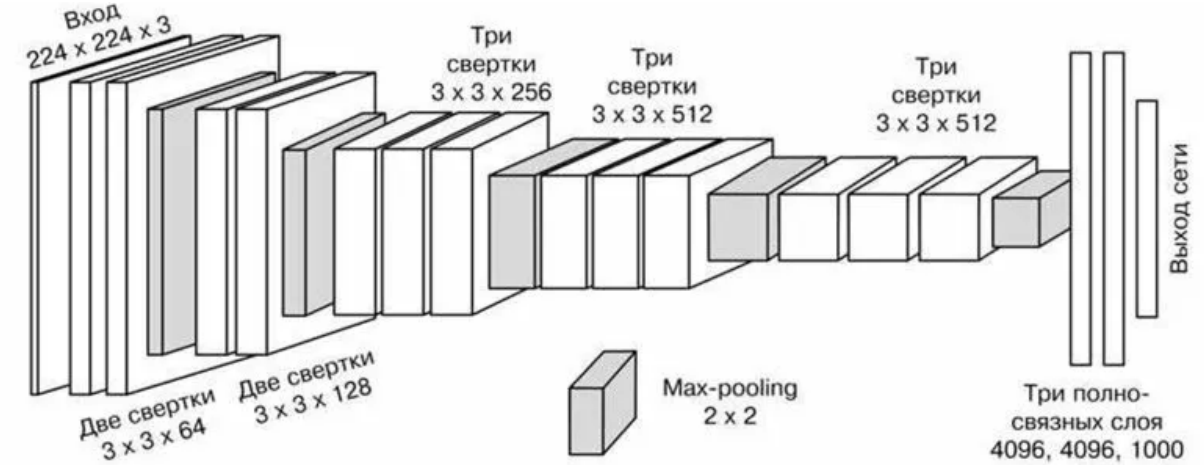

Рисунок 2 – Структура VGG-16

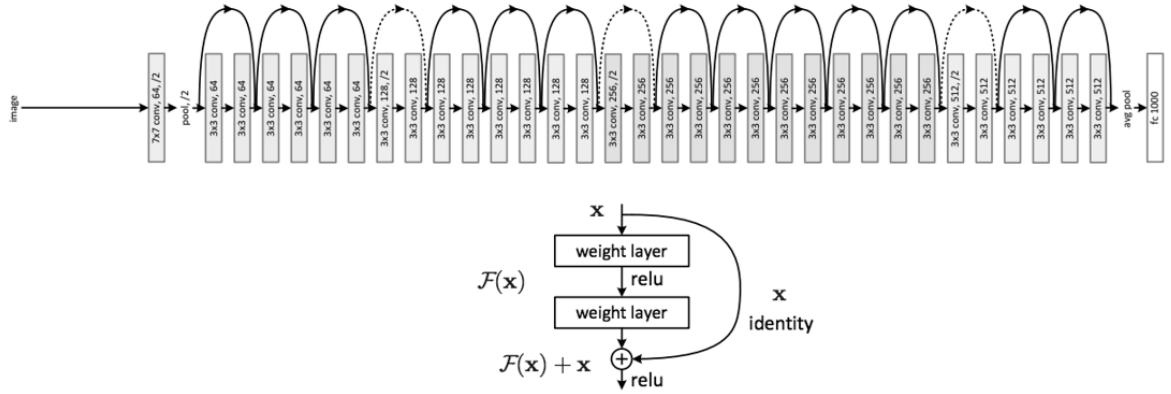

Рисунок 3 – Архитектура свёрточной нейронной сети ResNet50

– Inception: Архитектуры с инцепшн-модулями (например, в GoogLeNet (см. рис. 4)) эффективны благодаря параллельной обработке признаков на разных масштабах [7].

– Легковесные архитектуры: MobileNets, EfficientNets, ShuffleNets. Используют методы вроде глубинно-разделимых свёрток для снижения вычислительной сложности, что критично для мобильных устройств (например, MobileNet v2, представленная на рисунке 5) [8].

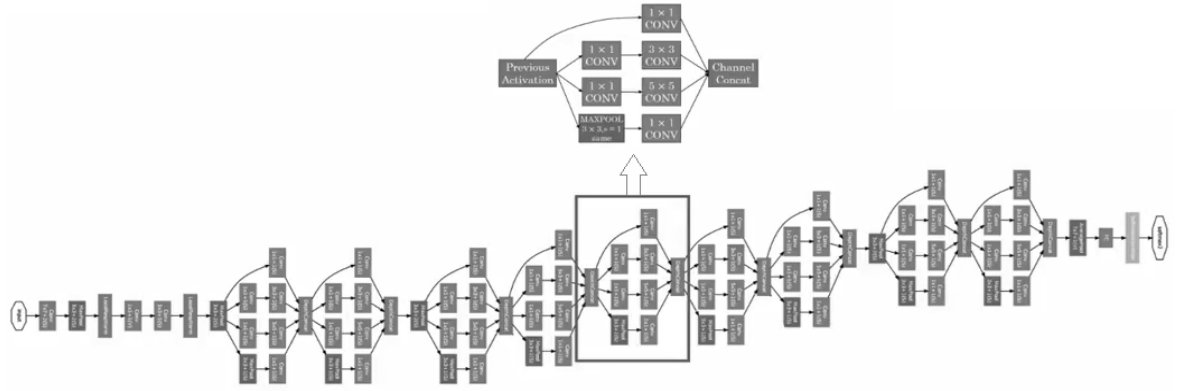

Рисунок 4 – Архитектура свёрточной нейронной сети Inception (GoogLeNet)

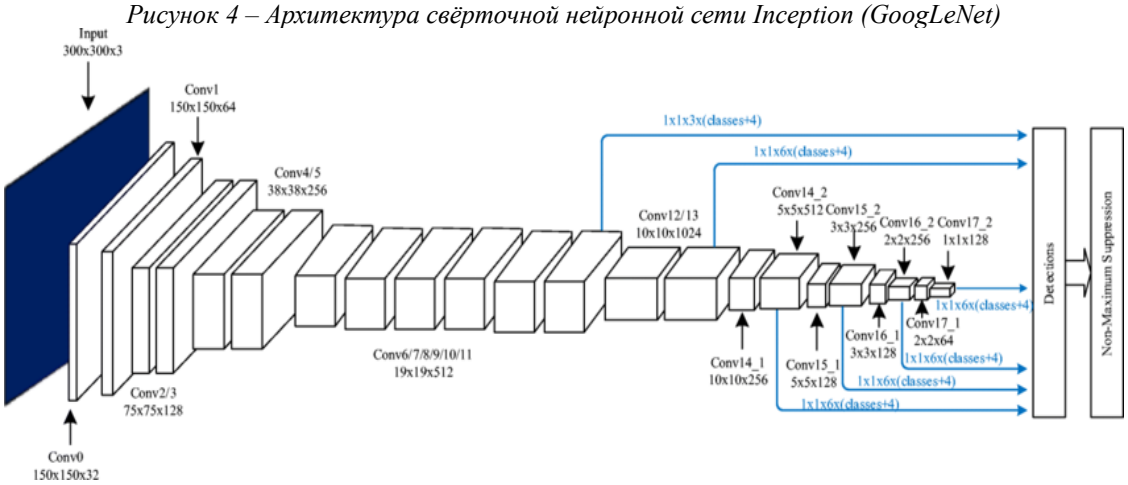

Рисунок 5 – Структура сети Mobilenet v2 + SSD

Для обучения дискриминативных признаков разработаны специальные функции потерь, работающие поверх выходов backbone-сети. К ним относятся Contrastive Loss и Triplet Loss, направленные на минимизацию расстояния между признаками одного класса и максимизацию расстояния между признаками разных классов. Также популярны модификации стандартной Softmax-потери с введением отступов (margins) для усиления внутриклассовой компактности и межклассовой разделимости, такие как SphereFace (A-Softmax), CosFace и ArcFace (Additive Angular Margin Loss), последняя является одной из наиболее популярных на сегодняшний день.

После получения векторов признаков следует этап принятия решения:

– Верификация: Задача сравнения "один к одному" (1:1). Вектор признаков предъявленного лица сравнивается с эталонным вектором признаков заявленной личности (например, хранящимся в базе данных). Если расстояние между векторами (например, евклидово или косинусное сходство) соответствует заданному порогу, личность подтверждается.

– Идентификация: Задача сравнения "один ко многим" (1:N). Вектор признаков предъявленного лица сравнивается со всеми векторами признаков в базе данных. Личность определяется как та, чей эталонный вектор наиболее близок к предъявленному (например, имеет наименьшее расстояние или наибольшее косинусное сходство), при условии, что это сходство превышает определенный порог.

Сами по себе эти этапы сравнения обычно не требуют специализированных CNN, а полагаются на метрики расстояния и пороговые значения. Однако качество и дискриминативность признаков, извлеченных CNN, напрямую определяют точность верификации и идентификации.

Для повышения робастности к окклюзиям, маскам и другим искажениям на этапе извлечения признаков или предобработки применяются различные подходы [9].

Аугментация данных включает применение различных преобразований к обучающим изображениям. Частичные свёртки могут использоваться в сетях для задач восстановления окклюдированных частей лица перед извлечением признаков или для обучения сетей быть более устойчивыми к отсутствующим данным; свёртка применяется только к валидным пикселям, а результат нормализуется.

Стробированные свёртки применяются в генеративных моделях и задачах реконструкции; обучаемый механизм стробирования может помочь сети адаптивно фокусироваться на значимых участках лица, игнорируя или восстанавливая искаженные, что косвенно улучшает качество извлекаемых признаков для распознавания.

Интегрированные и многозадачные CNN-архитектуры

Вместо последовательного применения отдельных моделей для каждого этапа, современные системы часто используют интегрированные архитектуры, выполняющие несколько задач одновременно.

Классическим примером интегрированной системы является MTCNN (Multi-task Cascaded Convolutional Networks) [4] (см. рис. 6).

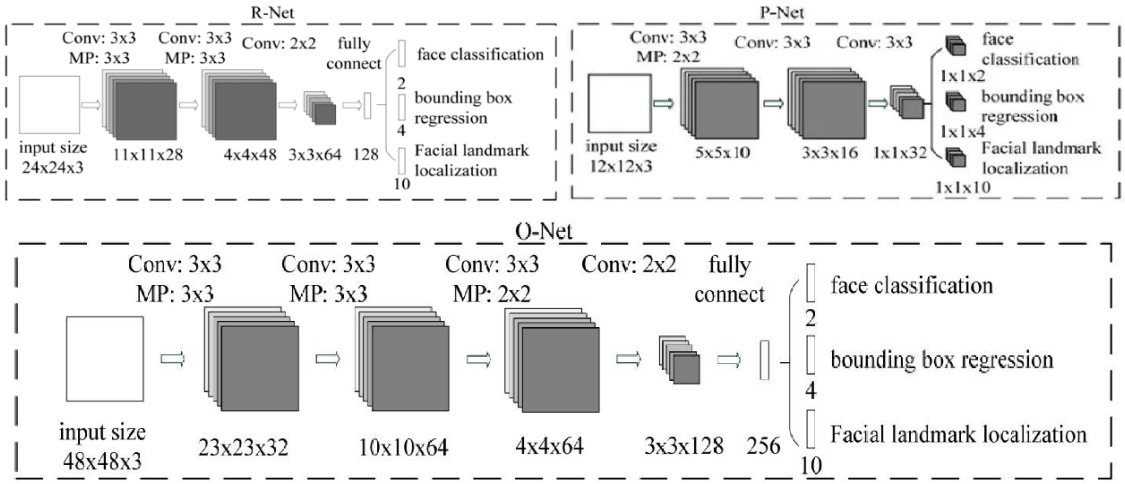

Рисунок 6 – Система MTCNN

Это каскад из трёх CNN: P-Net (Proposal Network) генерирует кандидатов окон с лицами; R-Net (Refine Network) фильтрует кандидатов и уточняет рамки; O-Net (Output Network) финально уточняет рамки и предсказывает координаты 5 ключевых точек лица для выравнивания. Каждая сеть решает несколько задач: классификация (лицо/не-лицо), регрессия ограничивающей рамки, регрессия ключевых точек (для O-Net).

Другим примером является RetinaFace (см. рис. 7) – современный одноэтапный детектор, который также является многозадачным [5]. На основе общего основной сети (например, ResNet+FPN) RetinaFace одновременно классифицирует области на наличие лица, регрессирует координаты ограничивающей рамки и координаты 5 ключевых точек лица. Иногда дополнительно предсказывает 3D-параметры лица или плотность пикселей лица. Преимуществами интегрированных подходов являются потенциально более высокая скорость (один проход через общую часть сети), лучшее использование признаков (признаки, полезные для одной задачи, могут быть полезны и для другой) и меньший размер общей модели.

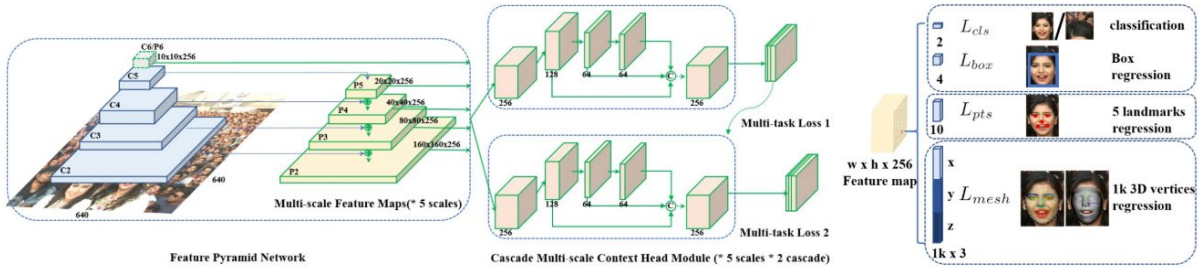

Рисунок 7 – Структура сети RetinaFace

Однако такие подходы сопряжены с вызовами, такими как сложность обучения и балансировки различных функций потерь, а также потенциальная интерференция между задачами. Хотя обнаружение и выравнивание часто интегрируются, этап извлечения признаков для распознавания (и последующей идентификации/верификации) обычно выполняется отдельной, более глубокой и специализированной CNN (например, ResNet с ArcFace) после того, как лицо было обнаружено и выровнено интегрированной системой.

Современные тенденции

Развитие CNN для анализа лиц продолжается, фокусируясь на следующих направлениях:

– Механизмы внимания: интеграция модулей, таких как Squeeze-and-Excitation (SE) блоки или Convolutional Block Attention Module (CBAM), позволяет сетям динамически перераспределять вычислительные ресурсы, фокусируясь на наиболее информативных признаках изображения и подавляя шум. Трансформерные модули самовнимания также начинают применяться для захвата глобальных зависимостей между частями лица, улучшая понимание контекста. Это особенно важно для распознавания лиц в сложных условиях (окклюзии, неоптимальные ракурсы).

– Трансформеры в зрении (Vision Transformers, ViT): показывают конкурентоспособные результаты, появляются гибридные модели (CNN+Transformer) типа ConvNeXt.

– 3D-распознавание и защита: Переход от 2D к 3D-анализу лиц позволяет повысить точность распознавания, особенно при вариациях позы, и является ключевым для борьбы со спуфинг-атаками (предъявление фотографии, видео или 3D-маски вместо реального лица). CNN обучаются извлекать признаки из 3D-данных (например, карт глубины, облаков точек) или оценивать 3D-форму лица по 2D-изображению. Для защиты от подделки разрабатываются CNN, анализирующие текстурные паттерны, микродвижения, отражательные свойства кожи и другие признаки "живости".

– Обучение на малых данных (Few-Shot Learning и One-Shot Learning) [10] представляет собой направление, ориентированное на разработку моделей, способных распознавать лица новых людей, имея лишь ограниченное число примеров. Суть подхода заключается не в запоминании конкретных классов, а в обучении самой способности быстро адаптироваться к новым задачам. Это достигается за счёт так называемого эпизодического обучения: модель во время тренировки решает множество небольших задач, имитирующих будущие тестовые условия, например, когда требуется различить пять лиц по одному примеру каждого. В качестве архитектурных решений часто применяются модели, основанные на сравнении эмбеддингов изображений. Например, в сиамских сетях модель учится определять, принадлежат ли два изображения одному и тому же человеку. В прототипических сетях каждое лицо представляется как вектор в признаковом пространстве, а классификация новых примеров выполняется по расстоянию до "центра" соответствующего класса. Также применяются методы метаобучения, такие как MAML (Model-Agnostic Meta-Learning), где модель обучается таким образом, чтобы её параметры можно было быстро адаптировать к новой задаче с минимальным числом градиентных шагов.

– Генеративные модели (GANs) находят применение для синтеза фотореалистичных изображений лиц, что полезно для аугментации данных, особенно для редких ракурсов, выражений или демографических групп. Они также используются для задач нормализации изображений (например, изменение ракурса лица к фронтальному, удаление очков, омоложение/состаривание), что может улучшить каноничность входных данных для основной сети распознавания.

Выводы

Свёрточные нейронные сети играют центральную роль на каждом этапе систем распознавания лиц. Для обнаружения и выравнивания часто используются

специализированные или многозадачные CNN, такие как MTCNN и RetinaFace, применяющие в том числе расширенные свёртки для лучшего анализа контекста. Для извлечения дискриминативных признаков, используемых в дальнейшем для идентификации и верификации, применяются глубокие архитектуры типа ResNet в сочетании с продвинутыми функциями потерь (ArcFace, CosFace). Облегчённые модели (MobileNet) обеспечивают работу на мобильных устройствах. Использование частичных и стробированных свёрток помогает в обработке неидеальных данных и генерации. Будущее, вероятно, за дальнейшей интеграцией, гибридными моделями и решением проблем робастности и скорости работы.

Литература

1. Федяев, О. И. Автоматическая регистрация присутствия студентов на учебном занятии с помощью компьютерного зрения / О. И. Федяев, И. А. Коломойцева // XXI Национальная конференция по искусственному интеллекту с международным участием КИИ-2023 (Смоленск, 16-20 октября 2023 г.). Труды конференции. В 2-х томах. Т.1. – Смоленск: Принт-Экспресс, 2023. – С. 294-303.

2. Du L., Zhang R., Wang X. Overview of two-stage object detection algorithms //Journal of Physics: Conference Series. – IOP Publishing, 2020. – Т. 1544. – №. 1. – С. 012033.

3. Zhang Y. et al. A comprehensive review of one-stage networks for object detection //2021 IEEE International Conference on Signal Processing, Communications and Computing (ICSPCC). – IEEE, 2021. – С. 1-6.

4. Zhang N., Luo J., Gao W. Research on face detection technology based on MTCNN //2020 international conference on computer network, electronic and automation (ICCNEA). – IEEE, 2020. – С. 154-158.

5. Deng J. et al. Retinaface: Single-shot multi-level face localisation in the wild //Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. – 2020. – С. 5203-5212.

6. Wang X., Bo L., Fuxin L. Adaptive wing loss for robust face alignment via heatmap regression //Proceedings of the IEEE/CVF international conference on computer vision. – 2019. – С. 6971-6981.

7. Sam S. M. et al. Offline signature verification using deep learning convolutional neural network (CNN) architectures GoogLeNet inception-v1 and inception-v3 //Procedia Computer Science. – 2019. – Т. 161. – С. 475-483.

8. Zhang X. et al. Shufflenet: An extremely efficient convolutional neural network for mobile devices //Proceedings of the IEEE conference on computer vision and pattern recognition. – 2018. – С. 6848-6856.

9. Alzubaidi L. et al. Review of deep learning: concepts, CNN architectures, challenges, applications, future directions //Journal of big Data. – 2021. – Т. 8. – С. 1-74.

10. Holkar A., Walambe R., Kotecha K. Few-shot learning for face recognition in the presence of image discrepancies for limited multi-class datasets //Image and Vision Computing. – 2022. – Т. 120. – С. 104420.

Кочетуров В. В., Федяев О. И. Свёрточные нейронные сети в системах обнаружения и распознавания лиц. *В статье рассматривается применение свёрточных нейронных сетей на различных этапах систем распознавания лиц: обнаружение, выравнивание и распознавание. Обозреваются ключевые архитектуры и методы, применяемые на каждом из этапов. Описываются многозадачные CNN, выполняющие несколько функций одновременно. Выделяются современные тенденции развития: применение механизмов внимания, трансформеров и обучение на малых данных.*

Ключевые слова: *свёрточные нейронные сети, распознавание лиц, обнаружение лиц, выравнивание лиц, архитектуры сетей, компьютерное зрение.*

Kocheturov V.V., Fedyaev O.I. Convolutional neural networks in face detection and recognition systems. *The article discusses the application of convolutional neural networks at different stages of face recognition systems: detection, alignment and recognition. The key architectures and methods used in each stage are reviewed. Multitasking CNNs that perform multiple functions simultaneously are described. Current development trends are highlighted: the application of attention mechanisms, transformers, and learning from small data.*

Keywords: *convolutional neural networks, face recognition, face detection, face alignment, network architectures, computer vision.*

Статья поступила в редакцию 28.02.2025
Рекомендована к публикации профессором Зори С. А.

УДК 004.021+004.853

Разработка комплексной методики для аннотирования изображений биопсии

Хрюкин Е.А.[1], **Мартыненко Т.В.**[1], **Васяева Т.А.**[1], **Шворoб Д.С.**[2]
[1]ФГБОУ ВО «Донецкий Национальный Технический Университет»
[2]Донецкий государственный медицинский университет им. М. Горького
E-mail: wadealer@yandex.ru

Аннотация:
*В статье предложена комплексная методика аннотирования медицинских изображений биопсии на основе языка **Python** и библиотеки **large-image**, включающая конвертацию изображений в формат **JPEG**, автоматическое выделение участков биопсии, их разбиение на фрагменты и использование инструмента **Makesense.AI** для аннотирования, позволяющая сэкономить вычислительные ресурсы и упростить работу специалистов, обеспечивающая эффективную подготовку данных для обучения нейросетей.*

Введение

Обработка и аннотирование изображений высокого разрешения представляет собой актуальную задачу в сфере анализа медицинских данных, особенно в контексте компьютерной диагностики и подготовки обучающих выборок для нейросетевых моделей.

Такие изображения, как правило, характеризуются высокой детализацией и большим объёмом, что требует использования специализированных алгоритмов и инструментов для эффективной работы с ними [1].

Особую сложность представляет аннотирование гистологических срезов, полученных в результате биопсии, из-за необходимости точного выделения морфологических структур на масштабных данных. Традиционные методы аннотирования оказываются трудоёмкими и малоэффективными при работе с подобными изображениями, особенно в условиях ограниченных вычислительных ресурсов [2].

В настоящей работе сделан акцент на оптимизации этапа подготовки данных за счёт автоматизации предварительной обработки изображений, алгоритмической фильтрации нерелевантных участков и адаптации существующих **open source**-инструментов для нужд конкретной задачи. Анализ современных решений, таких как **SlideRunner**, **QuPath** и специализированных **Python**-библиотек, позволил сформировать обоснованные требования к проектируемому подходу.

Целью статьи является формализация и реализация методики, позволяющей повысить эффективность аннотирования медицинских изображений большого формата при сохранении качества и полноты данных, необходимых для последующего обучения моделей компьютерного зрения.

Под эффективностью аннотирования понимается предоставление патологоанатому инструментария, позволяющего проаннотировать изображение максимально точно, без ошибок. Точность аннотирования в конечном счете может быть определена только экспертно.

Особенности медицинских изображений большого размера

Медицинские изображения большого размера, такие как снимки биопсии, представляют собой сложные данные, которые требуют особого подхода к обработке и анализу. Один снимок биопсии может занимать гигабайты памяти, особенно если он состоит из множества слоёв или сделан с использованием трёхмерной визуализации. Это создаёт сложности при передаче, хранении и обработке данных, а также увеличивает время, необходимое для аннотирования.

Изображения биопсии обычно создаются с использованием мощных микроскопов, что обеспечивает детализированное отображение тканей. Это необходимо для точного анализа микроскопических структур, таких как клетки, сосуды и патологические изменения. Однако высокое разрешение приводит к огромному объёму данных, что усложняет их хранение, обработку и аннотирование [3].

Формат **TIFF** (**Tagged Image File Format**) является одним из наиболее популярных форматов для хранения медицинских изображений благодаря своей гибкости и способности сохранять данные высокого разрешения. Основные преимущества формата **TIFF**: поддержка высокого разрешения, многостраничные файлы**,** поддержка сжатия. Для работы с **TIFF**-изображениями в программных приложениях используются различные библиотеки, каждая из которых имеет свои особенности и ограничения (таблица 1).

Таблица 1 – Библиотеки для работы с форматом **TIFF**

Наименование библиотеки	**Описание**
Libtiff	Библиотека с открытым исходным кодом, предоставляющая базовые функции для чтения, записи и манипуляции **TIFF**-файлами.
OpenSlide	Специализированная библиотека для работы с цифровыми слайдами, включая медицинские изображения в формате **TIFF.**
Pillow (Python Imaging Library)	Библиотека для работы с изображениями в **Python**, поддерживающая чтение и запись **TIFF**-файлов.
Bio-Formats	Библиотека, предназначенная для работы с биомедицинскими изображениями, включая **TIFF**.
Tifffile	**Python**-библиотека, специально разработанная для работы с **TIFF**-файлами, включая многослойные изображения и сжатие.
GDAL	Библиотека, изначально разработанная для геопространственных данных, поддерживает работу с **TIFF**.

Математическая постановка задачи

Рассматривается задача аннотирования больших медицинских изображений высокого разрешения, где требуется эффективно выделять и объединять области интереса.

Пусть I – изображение, заданное как отображение:

$$I: \Omega \subset \mathbb{R}^2 \rightarrow \mathbb{R}^c, \quad (1)$$

где Ω – ограниченная область в двумерном евклидовом пространстве, $\mathbb{R}$ – множество вещественных чисел, а c – количество цветовых каналов.

Требуется найти множество ограничивающих прямоугольников Ri, минимизирующее суммарную ошибку разметки и штраф за перекрытие областей. Формально, задача формулируется как задача оптимизации:

$$\min_{\mathrm{R_i}} \left(\sum_i L_{\text{разметки}}(R_i) + \lambda L_{\text{перекрытия}}(R_i) \right), \quad (2)$$

где $\mathrm{L_{разметки}(R_i)}$ – функция ошибки разметки для прямоугольника Ri;

$\mathrm{L_{перекрытия}(R_i)}$ – функция штрафа за перекрытие прямоугольников;

$\lambda > 0$ – регуляризационный коэффициент, который регулирует баланс между точностью разметки и минимизацией перекрытия.

Таким образом, задача состоит в построении эффективной разметки изображения, учитывающей точность выделения областей интереса. Функция ошибки разметки вычисляется по формуле:

$$L_{\text{разметки}}(R_i) = 1 - IoU(R_i, G_i), \quad (3)$$

где G_i – соответствующий эталонный (истинный) прямоугольник;

IoU (Intersection over Union) – пересечение по объединению.

Разработка комплексной методики для аннотирования изображений биопсии

Современные инструменты для аннотирования медицинских изображений предоставляют широкий функционал для работы с данными, однако они имеют ряд ограничений, особенно при работе с изображениями большого размера. Наиболее популярные решения и их недостатки представлены в таблице 2.

Для преодоления описанных выше ограничений, упрощения процесса аннотирования изображений биопсии и решения поставленной задачи была разработана комплексная методика, основанная на использовании языка **Python** и специализированных библиотек.

Методика включает применение следующих инструментов и технологий.

Библиотека **large-image** [4] **–** позволяет эффективно загружать и обрабатывать изображения по частям, что значительно снижает требования к оперативной памяти. Предоставляет унифицированный интерфейс для работы с низкоуровневыми библиотеками, описанными в предыдущем разделе. В качестве бэкэнда выбрана библиотека **tifffile,** которая в ходе экспериментов показала наилучшую производительность.

1. Конвертация изображения в формат **JPEG** с уменьшением разрешения. При этом сохраняется необходимое соотношение между размером файла и качеством изображения, чтобы обеспечить точность аннотации.

2. Для постобработки сконвертированного изображения используется библиотека **OpenCV** [5] – отраслевой стандарт для работы с изображениями.

Обобщенный алгоритм обработки изображения представлен на рис. 1 и включает следующие этапы: бинаризация и поиск контуров; разделение изображения на фрагменты; сохранение фрагментов.

Таблица 2 – Инструменты для аннотирования медицинских изображений.

Название	Описание	Недостатки
QuPath [6]	Один из наиболее распространённых инструментов для аннотирования биомедицинских изображений. Предоставляет мощные функции для работы с изображениями высокого разрешения.	Ограниченная производительность при обработке сверхбольших изображений, высокая нагрузка на аппаратное обеспечение, сложности в интеграции с пользовательскими скриптами или внешними системами.
ImageJ/Fiji [7, 8]	Широко используемый инструмент для анализа медицинских изображений, предоставляет модульность и поддержку множества плагинов.	Интерфейс, требующий значительного обучения, отсутствие специализированных инструментов для аннотирования биопсийных изображений, сложность обработки больших объёмов данных.
Aperio ImageScope	Ориентирован на просмотр и аннотирование цифровых слайдов.	Ограниченная поддержка форматов изображений, высокая стоимость лицензии, отсутствие гибкости для кастомизации рабочих процессов.
DeepLabCut и аналогичные инструменты	Предназначены для использования в исследованиях, связанных с машинным обучением и компьютерным зрением.	Сложность настройки и использования без технической подготовки, отсутствие инструментов для точной медицинской аннотации.
SlideRunner [9]	Инструмент с открытым исходным кодом, предназначенный для аннотирования цифровых слайдов. Он предоставляет интерфейс для ручной разметки, а также интеграцию с библиотеками машинного обучения.	Отсутствие удобного интерфейса для работы с изображениями сверхбольшого размера, недостаточная оптимизация для аннотирования сложных структур, ограниченные возможности автоматизации процесса разметки.
Облачные инструменты: AWS SageMaker, Google Cloud AI	Предлагают интеграцию с облачными вычислениями и поддерживают работу с большими данными.	Высокая стоимость использования, зависимость от интернет-соединения, отсутствие специализированных функций для работы с медицинскими изображениями.

Описание алгоритма бинаризации и выделения контуров

Визуализация результатов каждого этапа обработки изображений представлены на рис. 2.

Процесс бинаризации цветного изображения (рис. 2, а) выполняется с использованием алгоритмов обработки изображений, предоставляемых библиотекой **OpenCV**, и включает несколько этапов

Цветное изображение в формате **OpenCV** имеет три цветных канала в формате **BGR**. В результате проведенных экспериментов было установлено, что бинаризацию наиболее эффективно проводить по красному каналу (**channel** = 2).

Для выделения объектов на изображении используется метод пороговой бинаризации. Результат бинаризации — двоичное изображение **binary_image**, где объекты выделены белым цветом на чёрном фоне. Для улучшения качества бинарного изображения применяется морфологическая обработка, которая помогает устранить шумы и выделить структуры на изображении. Изображение, полученное в результате бинаризации, представлено на рис 2, б.

Алгоритм выделения контуров состоит из нескольких шагов:

- выделение начальных прямоугольников (рис. 2, в);
- объединение пересекающихся прямоугольников;
- объединение близко расположенных прямоугольников;
- фильтрация прямоугольников;
- повторное объединение пересекающихся прямоугольников.

Этот алгоритм используется для выделения областей интереса на изображении. Он позволяет уменьшить количество ложных срабатываний, объединить пересекающиеся и близкие области, а также сохранить только значимые регионы (рис. 2, г).

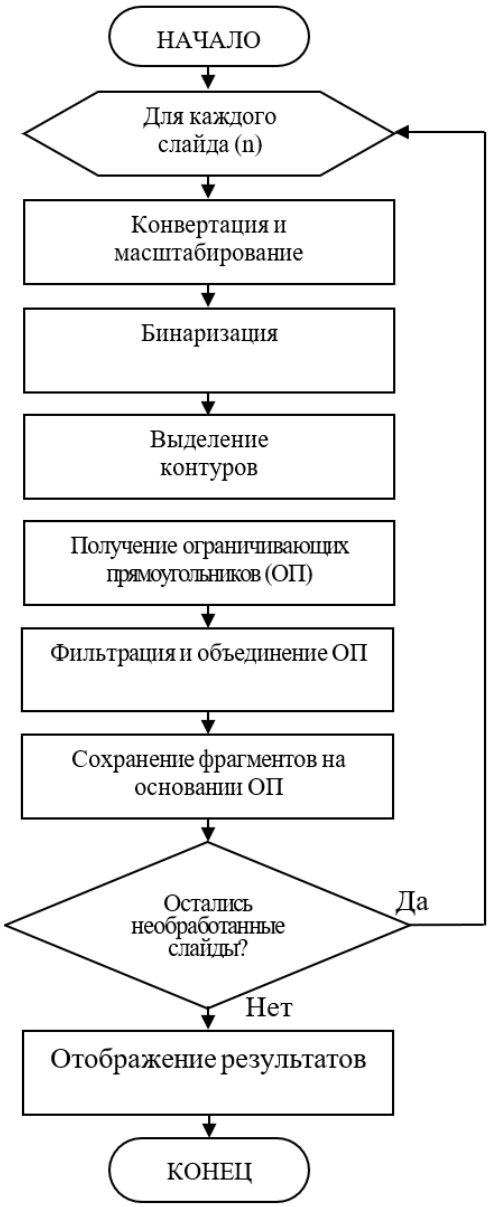

Рисунок 1 – Обобщенный алгоритм обработки изображения

Подготовленные фрагменты (рис. 2, д) сохраняются в отдельные файлы, в именах которых закодирована информация, необходимая для последующего переноса аннотаций на оригинальное изображение высокого разрешения. Эта информация включает имя исходного файла, координаты фрагмента относительно исходного изображения, ширину и высоту фрагмента до масштабирования.

Для разметки подготовленных фрагментов используется **Makesense.AI** [10] – веб-инструмент, который отличается простым и интуитивно понятным интерфейсом, поддержкой различных форматов аннотации (например, **JSON, XML**), а также гибкостью в настройке классов и типов аннотаций, что важно для патологоанатомов, работающих с уникальными структурами биопсийных материалов.

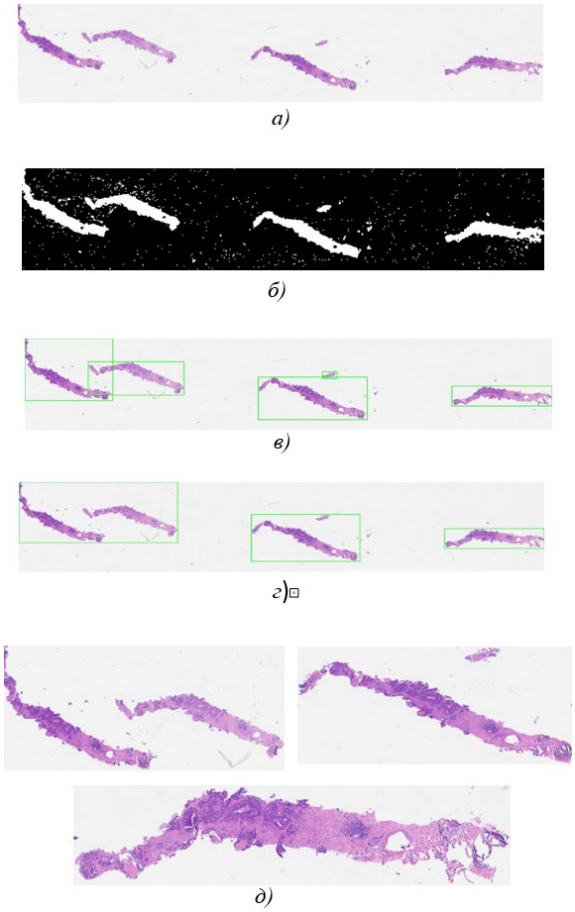

*Рисунок 2 – Этапы подготовки изображения: а) изображение сконвертировано в формат **JPEG** выбранного разрешения; б) произведен процесс бинаризации; в) выделены первичные ограничивающие прямоугольники для найденных объектов; г) близкие и пересекающиеся ограничивающие прямоугольники объединены; д) сохранены отдельные фрагменты изображения*

Заключение

В данной статье были рассмотрены основные сложности аннотирования медицинских изображений большого размера, таких как снимки биопсии, и предложено комплексное решение для их преодоления.

Предложенная методика использует библиотеку **large-image** с **tifffile** в качестве бэкэнда для обработки **TIFF**-изображений. Конвертация исходных данных в формат **JPEG** с необходимым разрешением позволяет значительно уменьшить размер файлов при сохранении качества, необходимого для аннотирования. Методы бинаризации и поиска контуров автоматизируют выделение областей интереса, а разбиение изображения на фрагменты упрощает процесс разметки. Сохранение метаданных, таких как координаты вырезанных областей, в именах файлов

позволяет отслеживать связь между исходными и аннотированными данными.

Для аннотирования полученных фрагментов был выбран инструмент **Makesense.AI** благодаря его простоте, гибкости и доступности для специалистов без технической подготовки. Такой подход обеспечивает удобство работы для патологоанатомов и минимизирует временные затраты на разметку.

Литература

1. Litjens, G., Kooi, T., Bejnordi, B. E., et al. A survey on deep learning in medical image analysis. *Medical Image Analysis*, 2017, 42: 60–88. DOI: 10.1016/j.media.2017.07.005

2. Ronneberger, O., Fischer, P., Brox, T. U-Net: Convolutional networks for biomedical image segmentation. *Medical Image Computing and Computer-Assisted Intervention (MICCAI)*, 2015, 9351: 234–241. DOI: 10.1007/978-3-319-24574-4_28

3. Aresta, G., Araújo, T., Kwok, S., et al. BACH: Grand challenge on breast cancer histology images. Medical Image Analysis, 2019, 56: 122–139. DOI: https://doi.org/10.1016/j.media.2019.05.010.

4. Large-Image Documentation. Kitware, Inc. https://doi.org/10.5281/zenodo.14624225 Интернет-ресурс. - Режим доступа : www/ URL: https://girder.github.io/large_image/

5. OpenCV Documentation. Open Source Computer Vision Library. Интернет-ресурс. - Режим доступа : www/ URL: https://docs.opencv.org/

6. Bankhead, P., Loughrey, M. B., Fernández, J. A., et al. QuPath: Open source software for digital pathology image analysis. Scientific Reports, 2017,7(1): 16878. DOI: 10.1038/s41598-017-17204-5.

7. Rueden, C. T., Schindelin, J., Hiner, M. C., et al. ImageJ2: ImageJ for the next generation of scientific image data. BMC Bioinformatics, 2017,18(1): 529. DOI: 10.1186/s12859-017-1934-z.

8. Schindelin, J., Arganda-Carreras, I., Frise, E., et al. Fiji: An open-source platform for biological-image analysis. *Nature Methods*, 2012, 9(7): 676–682. DOI: 10.1038/nmeth.2019

9. M. Aubreville, C. Bertram, R. Klopfleisch and A. Maier (2018) SlideRunner - A Tool for Massive Cell Annotations in Whole Slide Images. In: Bildverarbeitung für die Medizin 2018. Springer Vieweg, Berlin, Heidelberg, 2018. pp. 309-314. https://doi.org/10.1007/978-3-662-56537-7_81 Режим доступа : www/ URL: https://github.com/maubreville/SlideRunner

10. Makesense.AI. Онлайн-инструмент для аннотирования изображений. Интернет-ресурс. - Режим доступа : www/ URL: https://www.makesense.ai/

Хрюкин Е.А., Мартыненко Т.В., Васяева Т.А., Шворoб Д.С. Разработка комплексной методики для аннотирования изображений биопсии. *В статье предложена комплексная методика аннотирования медицинских изображений биопсии на основе языка* ***Python*** *и библиотеки* ***large-image****, включающая конвертацию изображений в формат* ***JPEG****, автоматическое выделение участков биопсии, их разбиение на фрагменты и использование инструмента* ***Makesense.AI*** *для аннотирования, позволяющая сэкономить вычислительные ресурсы и упростить работу специалистов, обеспечивающая эффективную подготовку данных для обучения нейросетей.*

Ключевые слова: *пороговая бинаризация, выделение контуров, large-image, tifffile, Python.*

Yevgeniy A. Khriukin, Tatyana V. Martynenko, Tatyana A. Vasyaeva, Danil S. Shvorob. Development of a comprehensive method for biopsy image annotation. *This paper presents a comprehensive approach to annotating medical biopsy images using* ***Python*** *and the* ***large-image*** *library. The approach includes converting images to* ***JPEG*** *format, automatically detecting biopsy regions, dividing them into fragments, and employing the* ***Makesense.AI*** *tool for annotation. This method helps save computational resources and simplifies the work of specialists, making it an effective tool for preparing data for neural network training.*

Keywords: *threshold binarization, contour detection, large-image, tifffile, Python.*

Статья поступила в редакцию 02.03.2025
Рекомендована к публикации профессором Скобцовым Ю. А.

УДК 004.4'2

Применение обучающих игр жанра «квест» в качестве метода интерактивного обучения

А.В. Боднар, С.А. Айдин
Донецкий национальный технический университет
кафедра программной инженерии
Email: linabykova13@ya.ru, aidin.serg@gmail.com

Аннотация
В статье рассматривается такой жанр игр как квест, проводится анализ необходимости применения интерактивных средств обучения в образовательной программе, выделяются ключевые элементы для игр жанра квест, формируются требования к играм данного жанра, рассматриваются существующие программные продукты для создания игр этого жанра. Также в статье выделяются требования для ПО для создания и проигрывания компьютерных обучающих игр, сформированных на основе структуры и требований к играм жанра квест.

Введение

Интерактивное обучение – форма образовательной деятельности, при которой обучающиеся становятся активными участниками процесса, посредством взаимодействия друг с другом, с преподавателей или с учебным материалом. Интерактивное обучение подразумевает заинтересованность и вовлеченность обучающихся в процесс, в отличие от стандартных методов обучения, до сих пор популярных в школах. Особое внимание уделяется к развитию умений анализировать, искать и обрабатывать информацию самостоятельно, в группе с учителем или коллегами.

Цель интерактивного обучения – помощь в создании необходимых условий для эффективного получения и обработки знаний обучающимися, изучение практической стороны применения знаний для выполнения некоторых реальных или почти реальных задач.

Квест – игровой жанр, который подразумевает последовательное выполнение игроками заданий для продвижения по сюжету игры.

Обучающие игры жанра «квест» – вид интерактивных средств обучения, представляющий собой игру в жанре «квест», в которой обучающиеся выступают игроками, которые должны решить основные задачи игры, которыми могут быть закрепляющие задания по изученной теме. В то же время материал изучаемой темы может служить подсказками в игре и тем самым мотивировать обучающихся к исследованию квеста, вместо прямолинейного следования от одного задания к другому.

Информационные технологии – совокупность методов и средств для обработки, передачи и хранения информации, активное применение которых, а также правильное их использование позволяет достичь наиболее эффективных результатов обучения, однако стоит оценивать данные методы не как дополнение, а как полноценное развитие в системе образования, методах организации и т.д [1].

Постановка проблемы

В настоящее время необходимость применения интерактивных методов обучения становится наиболее важной. Старые методы постепенно теряют собственную эффективность ввиду бурного роста информационных технологий, в том числе в области искусственного интеллекта. При этом, заинтересованность обучающихся в получении знаний становится всё меньше, что приводит к появлению методов обхода важных процессов образования. Из-за этого преподавателю не удаётся правильно оценить способности ребёнка, а ребёнку нет мотивации следовать более сложному пути, каким является ответственное выполнение работы.

Все эти проблемы создают необходимость в поднятии заинтересованности у обучающихся в учебном процессе, а также привлечении их к собственному размышлению, исследованию. Идеальным решением этой проблемы будет рассмотрение некоторых методик интерактивного обучения и выявление универсальной, доступной и эффективной из них для дальнейшего продвижения в образовательном процессе.

Анализ исследований и публикаций

В своей научной работе Строгонова Н. А. затрагивает тему интерактивных игр с подробным примером проведения живой игры-квеста в оренбургском училище [2]. В статье Панова Е. И. рассматривается обширный спектр

интерактивных игр, в том числе веб-квесты и квесты в формате презентаций [3]. Мясникова О. В. в своей работе подробно описывает понятие обучающих квестов, а также предоставляет собственный пример игры-квеста живого действия [4]. Сафонова Е. В. в своей научной работе рассмотрел теоретические сведения о жанр обучающий квест, выделив требования и классификации представленного жанра [5].

После подробного изучения вышеописанных статей можно сделать вывод, что все статьи рассматривают квест преимущественно как игру живого действия, что требует хорошей подготовки у преподавателя, и при этом трудно реализуемо за время стандартного урока.

Также становится ясно, что идея создания и проведения квестов с помощью информационных технологий рассмотрена недостаточно.

Рассмотренная проблема и существующие публикации позволяют установить ***цель исследования.*** Необходимо рассмотреть такой метод интерактивного обучения как обучающий квест, а также определить его актуальность применения в настоящее время. Для этого нужно вывести точное определение жанра квест, рассмотреть его основные особенности, провести опрос специалистов в сфере образования, составить требования к играм жанра обучающий квест. На основе этих требований можно составить требования к разрабатываемому ПО, функциями которого является создание и проигрывание компьютерных обучающих квестов.

Основные результаты исследования

Прежде всего, необходимо рассмотреть, что из себя представляет жанр "квест", чтобы понять его преимущества и недостатки, а также составить границы для разрабатываемого проекта.

На данный момент под "квестами" подразумевается обширный круг обучающих игр и мероприятий, где участникам даётся ряд задач, которые они должны решить. Этими задачами могут быть в зависимости от предмета обучения: загадки, ребусы, головоломки, математические выражения, неравенства, иностранные термины и т.д.

Поскольку на данный момент не существует унифицированных типов интерактивных обучающих игр, каждый трактует понятие "квест" по-своему: иногда это может быть простые тесты соревновательного характера, а иногда полноценные ролевые игры.

Далее будет рассмотрены квесты как организованное мероприятие исследовательского характера, где обучающиеся поодиночке или группами проводят поиск подсказок для решения ключевых задач, позволяющих им успешно пройти игру или же продвинуться на новый, так называемый, уровень.

Были составлены основные элементы, которые могут использоваться при создании обучающих игр жанра квест.

1. Сюжет. Каждый квест прежде всего состоит из истории, которая оправдывает задачи, очерчивает локации, может привести к причине появления подсказок, а также мотивирует участников следовать по логическому пути квеста.

2. Задача. В каждом квесте во главе стоит задача, решение которой позволяет перейти к последующей задаче или успешно завершить квест. Решение задачи должно быть однозначным.

3. Локация. Квест должен содержать обязательно хотя бы одну локацию, в которой могут находиться участники. Данные структуры позволяют ограничить круг поиска, тем самым игроки имеют меньше шансов потеряться в поисках решения задач.

4. Подсказки. Информация различного вида (визуальная или звуковая), возможно скрытая, а возможно наглядная для играющих. Она должна давать полезную информацию для решения задачи или поиска других подсказок.

5. Декорации. Желательно добавлять различные декорации (визуальные, звуковые, тактильные и т.д.), которые позволяют создать необходимый антураж игры, чтобы больше заинтересовать игроков в участии.

6. Наставники. Данную роль обычно играет преподаватель и, возможно, несколько других людей. Они не принимают участие в игре-квесте, однако могут помогать участникам, если последние столкнулись с какой-то проблемой. Также они могут самостоятельно давать какие-то подсказки и также могут поддерживать заинтересованность игроков посредством активного общения.

Все эти элементы можно использовать в дальнейшем для проектирования структуры компьютерных обучающих квестов.

Квесты являются сложными в организации с точки зрения времени, возможности обучающихся и способности преподавателя к проектированию загадок. Частичным решением может послужить применение информационных технологий для того, чтобы переместить загадки и их решение в виртуальное пространство компьютера. Это приводит к появлению требований к квестам, как к обучающим играм.

Для расширения требований к обучающим квестам был проведён неформальный опрос группы преподавателей из общего среднего общеобразовательного учреждения, а также нескольких преподавателей из политехнического института. Затрагивались такие вопросы, как:

удовлетворённость в текущем формате обучения, пожелания в изменении, трудности обучения и применения новых методов. По мнению опрошенных были составлены такие выводы:

- на данный момент до сих пор актуальна проблема «задних парт» - когда обучающиеся на последних рядах уделяют меньшее внимание к обучению и пассивно принимают участие в практической деятельности;
- стандартный формат домашних заданий перестал показывать настоящие знания обучающихся, потому что интернет позволяет найти ответы почти на все задания, а с развитием нейронных сетей даже модифицированные преподающим задания можно решить без понимания теории;
- перед обучающими ставится необходимость добавления интерактивных средств обучения в образовательных процесс, однако чаще всего это не реализуется в должном порядке из-за сжатых сроков на рассмотрение тем занятий;
- обучающие заинтересованы в привлечении обучающихся к учебной деятельности, однако помехой выступает либо мотивированность студентов, либо желание преподавателя уделять больше минимального необходимого времени на подготовку занятий;
- текущие методы интерактивного обучения необходимо развивать и делать более доступными для повсеместного применения;
- применение информационных технологий в образовании неизбежно, поэтому следует разработать технологии для того чтобы правильным образом их применять для обучения.

На основе вышеописанных результатов можно выявить актуальность разработки программных средств для интерактивного обучения. Прежде всего в них нуждается среднее общее образования, однако в дальнейшем сфера применения может расшириться за счёт заведений профессионального и высшего образования.

Особое внимание стоит выделить тому, что первое внедрение технологии наиболее уместно для замены стандартных домашних заданий в школах. Общее недовольство касательно домашних заданий выражают как ученики, так и учителя, потому что они дают всё меньший эффект для закрепления материала – это стало более очевидно во время дистанционного обучения, вынуждающего обучающихся к самостоятельной работе.

Дистанционное обучение требует от обучающегося большей ответственности, вызывает большую перегрузку из-за необходимости самостоятельно изучать темы и затем прорабатывать их, что приводит к появлению желания упростить себе работу посредством списывания. При очном обучении ещё можно заметить это учителю, но при дистанционной такой возможности нет [6].

Также возможно такая ситуация, когда обучающийся не просто просит помощи у своих родителей, но и перекладывает выполнение домашних заданий на них. Объясняется это тем, что родители в виду своих знаний, полученных при очном обучении, могут решить учебные задачи быстрее, чем ученик. Но от такого сам обучающийся не получает ничего кроме неоправданных оценок по предмету.

При этом, домашние задания как форма обучения остаются необходимыми. Мнение педагогов и родителей обучающихся остаются таковыми – нельзя изучить учебную тему только по занятиям, требуется закрепление. К тому же, считается что домашняя работы повышает самостоятельность и ответственность у ученика, что ещё раз подтверждает необходимость не исключения домашней работы, а её преобразования [7].

Требуется преобразовать домашнюю работу так, чтобы она вызывала у обучающегося интерес, поэтому важным вопросом становится мотивация. Её можно получить если придать процессу обучения характер ролевой игры, опираясь на возрастную категорию обучающихся.

Основываясь на вышеуказанной информации и результатах опросов, можно составить требования к обучающим играм жанра квест (далее – обучающим квестам):

1. Обучающий квест должен иметь основной целью обучение игроков.

1.1. Обучающий квест должен иметь некоторую однозначно определенную область знаний или навыков, который обучающиеся развивают при прохождении игры.

1.2. Образовательные цели и темы должны быть интегрированы в весь игровой процесс и применяться для решения загадок.

1.3. Необходимо подбирать баланс между игровой составляющей квеста и обучающей частью, чтобы не превратить обучающий квест в простую игру или в простой урок.

1.4. Обучающий квест должен быть адаптирован под возрастную категорию играющих, а также учитывать их умственные способности. Слишком сложный квест может вызвать лишь скуку у обучающихся, а слишком лёгкий квест не сможет их обучить чему-то.

2. Обучающий квест должен содержать в себе задачи и требовать их решения.

2.1. Обучающая игра в жанре квест должна иметь задачи, представляющие из себя загадки, для решения которых требуется логическое мышление и анализ информации из подсказок.

2.2. Обучающий квест должен мотивировать игроков к исследованию и анализу полученной информации, а не просто давать ответ на заданный вопрос.

2.3. Обучающий квест должен отражать возможные реальные ситуации, чтобы развивать у игроков практическое применение полученных знаний.

3. Обучающий квест должен быть интерактивным.

3.1. Участники обучающего квеста должны принимать активное участие в игре.

3.2. Роль преподавателя в обучающем квесте должна быть сведена к минимуму.

3.3. Преподаватель может обсуждать с игроками их рассуждения, давать небольшие подсказки, но не должен вмешиваться в мыслительный процесс игроков, даже если он неверный (за редкими исключениями, когда идеи игроков в корне неверные или нарушают порядок игры).

3.4. Обучающий квест должен всяким образом указывать обучающимся, если их решение неправильное. Недопустимо продолжение квеста, если задача была решена неверно.

4. Обучающий квест должен мотивировать участников.

4.1. Обучающий квест должен включать в себя игровые механики (уровни, очки, баллы, награды и т.д.).

4.2. Если обучающий квест состоит из нескольких главных задач, идущих последовательно, то требуется, чтобы их сложность повышалась по мере продвижения участников квеста.

5. Обучающий квест должен быть коллаборативным (опционально)

5.1. Обучающий квест должен иметь возможность совместного прохождения группой участников.

5.2. Обучающий квест должен позволять коммуникацию между игроками, обсуждение задания и имеющейся информации.

6. Обучающий квест должен быть связан с обучающей программой и соответствовать образовательным стандартам.

6.1. Содержание обучающего квеста должно соответствовать образовательной программе учебных заведений, что требуется для применения данного жанра игр в образовательном процессе на повсеместном уровне.

6.2. Обучающий квест должен позволять получить оценку способностей участников или групп участников в зависимости от некоторых факторов оценивания (время прохождения, количество попыток, количество необходимых подсказок и др.).

Для того чтобы позволить проведение обучающих квестов в качестве домашней работы обучающихся, необходимо разработать программный продукт, который способен воспроизводить созданные сюжеты квестов пользователю, отслеживать его прогресс и демонстрировать результаты выполнения заданий при завершении квеста. Из этого требования следует следующее: необходимо разработать программное обеспечения для создания и проведения обучающих игр жанра квест.

Применение информационных технологий для проведения квестов как интерактивных средств обучения не является новым. Существует большое количество сайтов и мобильных приложений, созданных для добавления игровой составляющей в образовательный процесс. Далее будет рассмотрены некоторые примеры, на основе которых будут составлены требования к разрабатываемому ПО.

Genially – веб-платформа для создания различных интерактивных средств, включая игры жанра квест, презентации, а также обычные виды тестов. Инструмент предлагает обширный набор интерактивных элементов, а создание квеста похоже на всем привычную презентацию в Microsoft PowerPoint. Авторы программного продукта ориентируют свой проект на улучшение качества электронного обучения обучающихся посредством геймификации, при этом стараясь упростить работу преподавателя по созданию квестов [8].

На рисунке 1 представлен пример квеста и интерфейса веб-платформы Genially. В таблице 1 представлены преимущества и недостатки Genially.

Таблица 1 – Преимущества и недостатки Genially

Преимущества	**Недостатки**
Поддержка мультимедийных элементов (изображений, видео, аудио информации)	Бесплатное использование накладывает ряд ограничений на интерактивные элементы
Наличие готовых шаблонов и элементов для простого создания небольших квестов.	Необходимо подключение к сети Интернет
Простота использования	Отсутствие русской локализации
Возможность создания нелинейного повествования	

Рисунок 1 – Genially

Платформа отлично подходит для создания визуального квеста, наполненного большим количеством различного медиа-контента. Однако для получения наиболее эффективного продукта требуется покупка подписки, что делает повсеместное применение данным ресурсом менее вероятным. Также отсутствие русской локализации усложняет работу преподавателей с платформой.

Twine – инструмент для создания текстовых интерактивных игр жанра квест. Twine поддерживает нелинейное повествование истории, что позволяет создать симуляцию перемещения по локациям. Данный продукт, за некоторыми недостатками, является хорошим кандидатом для набора инструментов для создания простых обучающих игр жанра квест в виде книги с нелинейным повествованием.

Важной деталью разработчики проекта отмечают, что готовые квесты можно экспортировать в файле HTML, что позволяет имплементировать данный квест на любом другом сайте. К тому же, инструмент является полностью бесплатным, даже для коммерческого использования созданных с его помощью квестов [9].

На рисунке 2 представлен интерфейс веб-версии приложения и формат создаваемых квестов. В таблице 2 представлены преимущества и недостатки Twine.

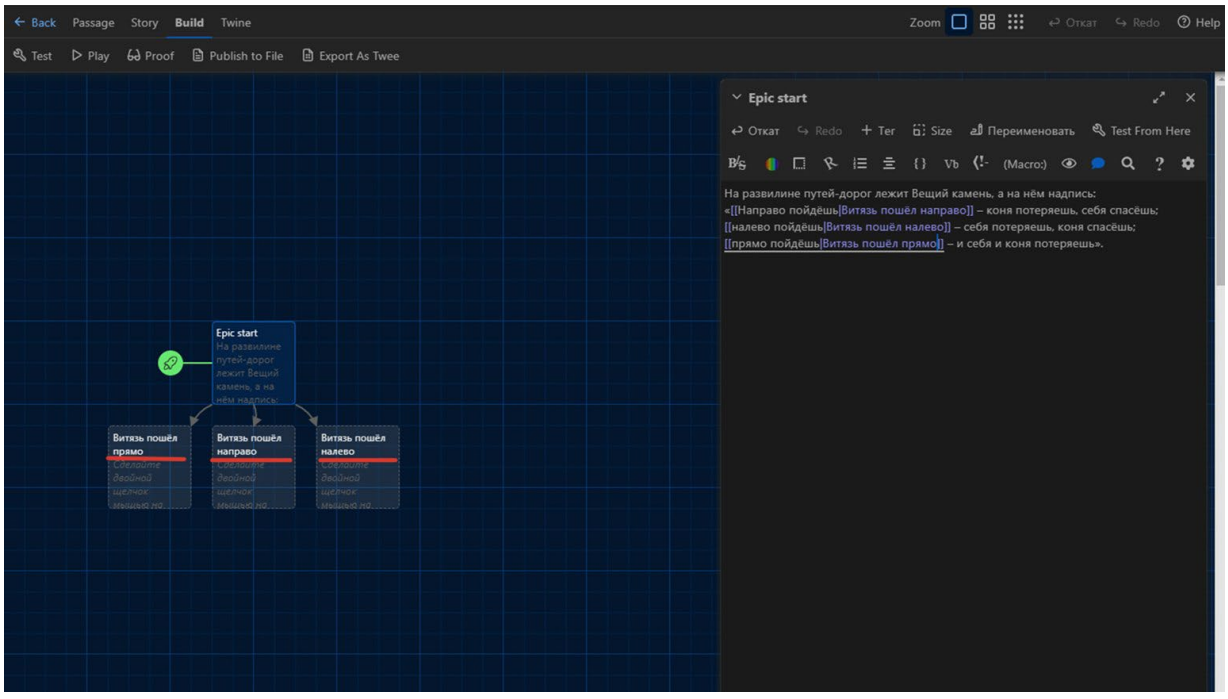

Рисунок 2 – Twine (веб-версия)

Таблица 2 – Преимущества и недостатки Twine

Преимущества	Недостатки
Бесплатное программное обеспечение с открытым исходным кодом	Требуется базовое знание программирования для работы с сложными функциями.
Простота создания разветвленного сюжета для интерактивного повествования	Для использования мультимедийных файлов требуется модификация кода вне приложения.
Поддержка встроенных сценариев, что позволят создавать и использовать более сложные игровые механики в квестах	Интерфейс не является интуитивно понятным, требуется время для изучения программы.
Экспорт квестов в HTML формате	Отсутствие локализации на русский язык
Онлайн и оффлайн версия программы	

Инструменты Twine отлично подходит для создания текстовых квестов с нелинейной историей, которые акцентируют внимание на логике и решении проблем в различном порядке. Однако стоит отметить, что визуально инструмент не является ни комфортным, ни привлекательным, что может негативно сказаться на мотивации учеников проходить эти квесты, а у учителя – мотивации создавать данные квесты, особенно без наличия русского языка в интерфейсе продукта.

Google Forms – веб-платформа, позволяющая создавать тестовые формы и просматривать информацию по результатам тестирования обучающихся.

Google Sites – инструмент для создания собственного веб-сайта без знаний в программировании. Подходит для создания личных веб-сайтов или учебных веб-страниц.

Совместно эти два инструмента можно использовать для создания веб-квеста. Google Forms собирает информацию от пользователей, а Google Sites представляет визуальную составляющую квеста. В сети Интернет присутствует большое количество текстовых рекомендаций по созданию веб-квестов с пошаговыми инструкциями к построению сайта.

На рисунке 3 представлен пример веб-квеста с помощью Google Sites на тему выбора своей будущей профессии, а на рисунке 4 представлен фрагмент веб-квеста с использованием встроенного инструмента Google Forms для сбора и оценки ответов. На рисунках видно, что созданный веб-квест не совсем соответствует основным требованиям для обучающих игр жанра квест, а задания могут быть ограничены возможностями средств утилиты Google Forms, которая позволяет создавать тесты с вариантами, задания с вводом числа или строки и с соотношением элементов двух колонок.

В таблице 3 представлены преимущества и недостатки использования Google Forms и Google Sites совместно для проектирования веб-квестов.

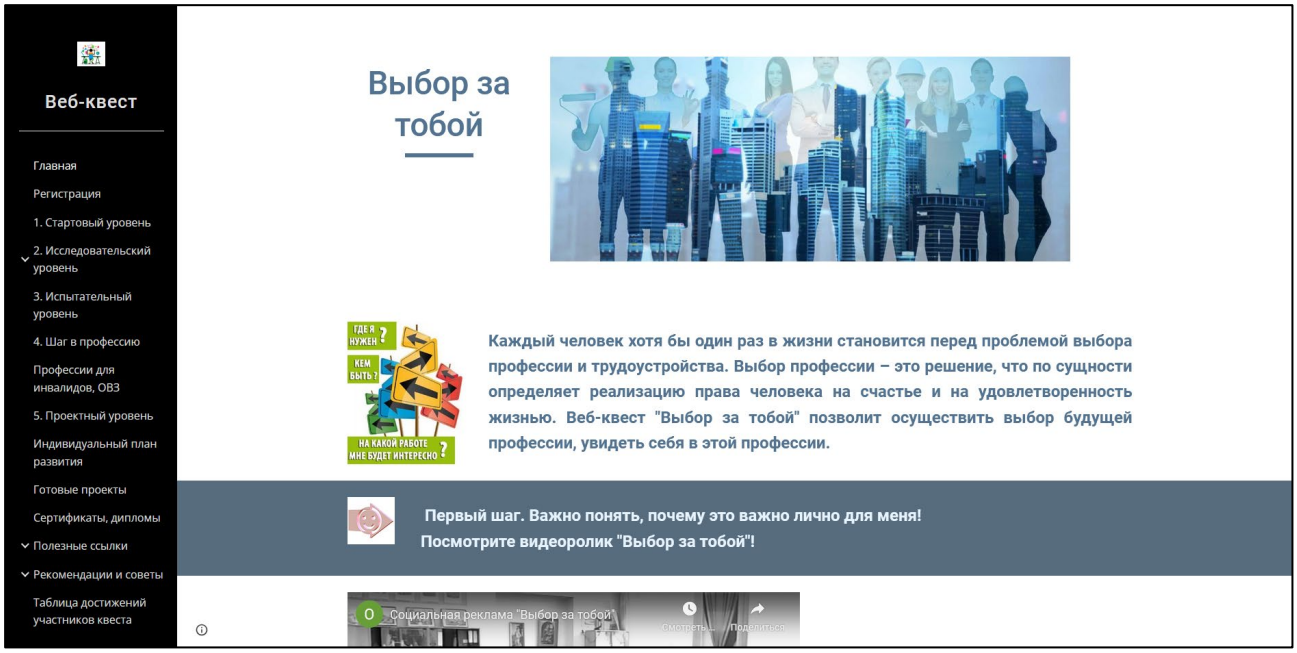

Рисунок 3 – Интерфейс веб-квеста с Google Sites

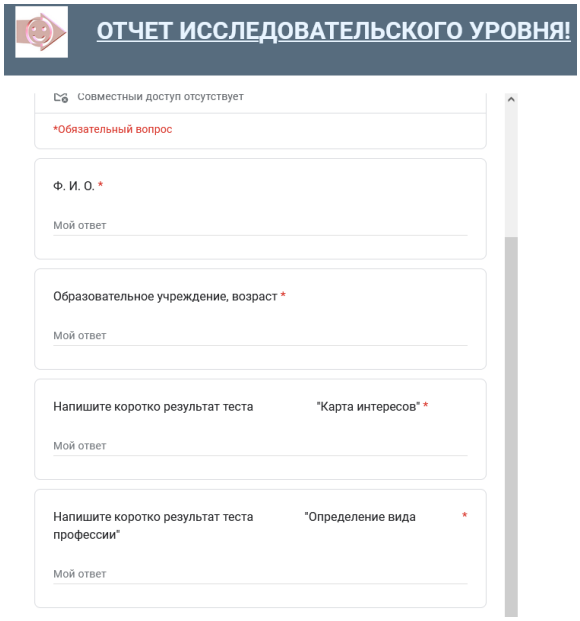

Рисунок 4 – Пример использования Google Forms в Google Site для сбора ответов

Таблица 3 – Преимущества и недостатки Google Forms + Google Sites

Преимущества	**Недостатки**
Бесплатный и широкий доступ	Невозможность создания сложных интерактивных механик
Интеграция аудио, видео и ссылочных файлов.	Требуется знания в создании сайтов и форм
Автоматизация процесса оценивания и сбора данных	Не подходит для квестов, в основе которых нет тестов.
	Требуется подключение к интернету для прохождения и создания квестов

В качестве вывода можно сделать следующее: Google Forms и Google Sites является эрзац-решением, если необходимо составить небольшой квест на основе теста. Google Forms уже сейчас используются как практика в обучении, однако для того чтобы сделать из тестов квест – требуются значительные усилия со стороны обучающего, что будет помехой в повсеместном применении. Требуется обладать творческим подходом чтобы выйти за границы простого теста с помощью данных утилит.

Room Escape Maker (REM) – веб-платформа для создания графических квестов по принципу «побег из комнаты». Игроку необходимо решить ряд логических задач для того, чтобы отпереть дверь, ведущую из комнаты. Создатели позиционируют свой продукт как эффективный и быстроразвивающийся проект, позволяющий создавать квесты как для развлекательный целей, так и для дистанционного обучения посредством геймификации [10].

На рисунке 5 представлен интерфейс программного продукта и простой пример созданного квеста. В таблице 4 представлены преимущества и недостатки Room Escape Maker.

Таблица 4 – Преимущества и недостатки Room Escape Maker

Преимущества	**Недостатки**
Простота использования	Не подходит для создания нелинейных и разветвленных квестов
Встроенные элементы геймификации (подсказки, система баллов и таймеры)	Бесплатная версия накладывает ограничения на все этапы проектирования и проведения квеста
Возможность использования мультимедийных элементов	Дополнительные объекты требуется покупать за реальные средства.
	Требует постоянного подключения к интернету
	Нет локализации на русский язык

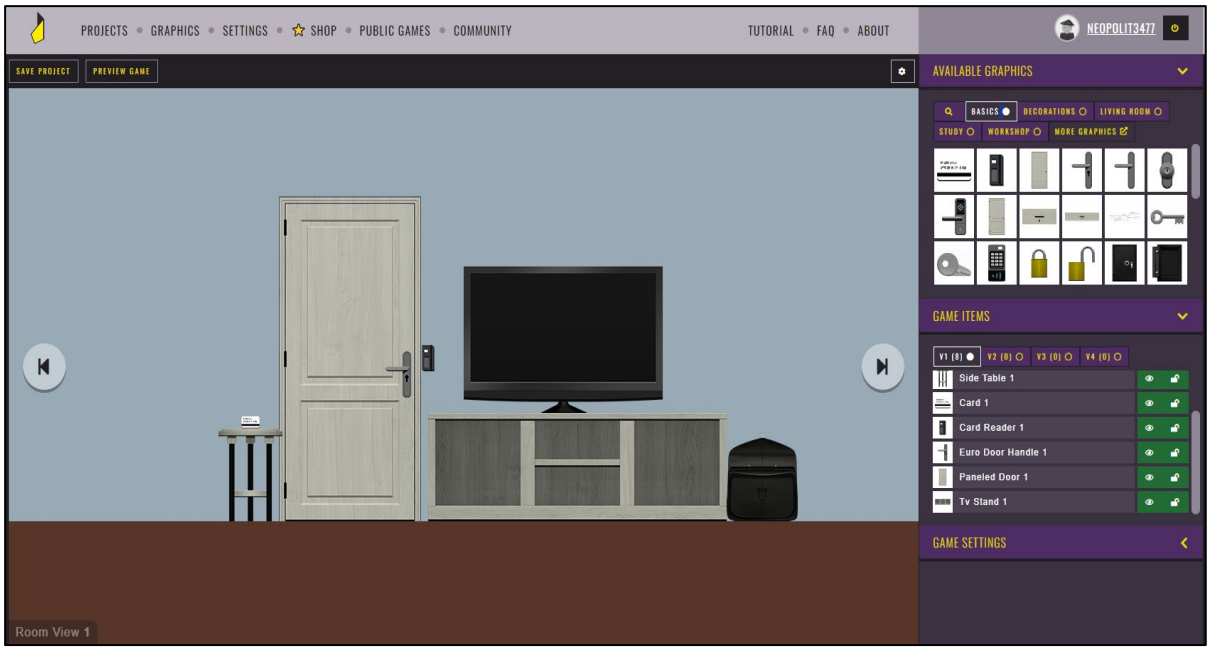

Рисунок 5 – Room Escape Maker

Веб-платформа Escape Room Maker подходит для создания простых квестов на логику, которые могут подойти для обучающихся младших и средних классов. Из-за невозможности создания нелинейного повествования и крайне ограниченной бесплатной версии, применение данного инструмента в образовательных целях повсеместно невозможно.

Из рассмотренных программных продуктов можно сделать вывод, что ни один из них не является достаточно качественным чтобы использоваться повсеместно в процессе обучения. Все инструменты содержат в себе некоторые положительные стороны, но также имеют серьёзные недостатки: нехватка локализации на русский язык, сложный интерфейс, коммерческие ограничения. Также стоит отметить, что все инструменты представляют разные виды квестов: в Genially и REM упор делается на графическое содержание и логику, в то время как в Twine и Google Sites упор делается на текстовое содержание квеста.

В зависимости от дисциплины и целевой аудитории игры жанра квест могут понадобиться как текстовое наполнение, как у книги, так и визуальное наполнение, как у презентации. Универсальным решением является квест, в котором присутствует сочетание как текстового, так и графического материала (вложенные изображения, видео или аудио). Требования к разрабатываемому программному продукту исходят из требований к играм жанра квест [11], а также из соображений простоты, комфорта и эффективности использования:

Функциональные требования ПО:

- ПО должно позволять создавать квесты.
- ПО должно поддерживать создание нелинейных квестов.
- ПО должно позволять запускать готовые квесты, созданные с помощью аналогичного приложения
- ПО должно содержать базовые шаблоны квестов разных уровней сложности для упрощения создания квестов на первых этапах.
- ПО должно поддерживать использование мультимедийный элементов в квестах (изображения, видео, аудио)
- ПО должно содержать все необходимые базовые элементы для построения квеста (локации, подсказки, наставники, задания)
- ПО должно поддерживать элементы геймификации (баллы, полоски прогресса) и средства для их реализации на этапе создания квеста
- ПО должно проводить оценку прохождения квеста (время, количество перемещений, ответы на задания, баллы и шкала прогресса)

Требования к интерфейсу ПО:

- должен быть интуитивно понятен даже для молодого пользователя.
- должен иметь локализации на русском и английском языках.
- должен поддерживать возможность добавления локализации в будущем.
- должен реализовать создание квестов графическим способом с помощью drag-and-drop элементов.
- должен содержать инструкцию по использованию как для создания квестов, так и для прохождения.
- должен поддерживать все основные виды формирования текста (шрифт, размер,

стили, цвет, положение и др.)

- должен предупреждать пользователя об ошибках, возникающих при создании квеста и предлагать их решение

Требования к экспорту и импорту данных:

- должен позволять экспортировать результаты прохождения квеста в файл как демонстрация результата работу игрока.
- все необходимые для квеста объекты мультимедиа должны быть собраны в единый файл для простоты импорта и экспорта.
- экспортировать квест в ограниченном (для игрока) или полном виде (для редактора).

Требования к доступности ПО:

- ПО должно распространяться на бесплатной основе, все функции приложения должны быть доступны на некоммерческой основе.

Вывод

В рамках данного исследования выявлены проблемы распространенности интерактивных средств обучения, затронуты проблемные места в образовательной деятельности и представлены методы решения посредством расширения интерактивности обучения.

Проведён обзор игр жанра квест и составлены требования к данному жанру для того, чтобы повсеместно использоваться в образовательном процессе.

Рассмотрены существующие инструменты для создания интерактивных игр жанра квест с целью геймификации образовательного процесса, выявлены их положительных и отрицательные стороны, на основе которых составлены требования к новому ПО для создания и проигрывания обучающих игр жанра квест.

Дальнейшее исследование по этой теме позволит развить требования к разрабатываемому ПО и открыть новые возможности и интерактивные методы обучения.

Литература

1. Кошкина, В. А. Интерактивные средства обучения: классификация и потенциал / В. А. Кошкина, Е. А. Пазенко // Мир науки. Педагогика и психология. – 2021. – Т. 9. - № 3. – EDN RNQAJU.

2. Строгонова, Н. А. Образовательный квест как средство активизации познавательной деятельности обучающихся / Н. А. Строгонова // World science: problems and innovations : Сборник статей XIII Международной научно-практической конференции. В 2-х частях, Пенза, 30 сентября 2017 года. Том Часть 2. – Пенза: "Наука и Просвещение" (ИП Гуляев Г.Ю.), 2017. – С. 223-225. – EDN ZGUUIF.

3. Панов, Е. И. Использование интерактивных обучающих игр и квестов как элемент инновации в образовательном процессе / Е. И. Панов // Преемственность в образовании. – 2021. – № 30(12). – С. 152-155. – EDN CIBPXV.

4. Мясникова, О. В. Опыт использования квест-технологии в обучении (на примере проведения квест-игры для старшеклассников «Учитель будущего») / О. В. Мясникова, Н. Н. Пержан // Опыт и перспективы обучения иностранным языкам в евразийском образовательном пространстве. – 2024. – № 9. – С. 69-75. – EDN ZAZRAK.

5. Сафонова, Е. В. Образовательный квест: смысл, содержание, технологические приёмы / Е. В. Сафонова // Народное образование. – 2018. – № 1-2(1466). – С. 83-87. – EDN YSTFWX.

6. Лебедева, Е. И. Проблемы реализации домашнего задания в школе в условиях дистанционного обучения / Е. И. Лебедева // Управление цифровой трансформацией общего и профессионального образования : Сборник трудов всероссийской научно-практической конференции с международным участием, Павлово, 03 марта 2021 года. – Павлово: Павловский филиал федерального государственного автономного образовательного учреждения высшего образования "Национальный исследовательский Нижегородский государственный университет им. Н.И. Лобачевского", 2021. – С. 137-141. – EDN TMIGJY.

7. Ускова, И. В. Анализ практики организации самостоятельной учебной деятельности обучающихся: сравнительные результаты исследований / И. В. Ускова // Формирование единого образовательного пространства: задачи, решения, перспективы : Сборник научных трудов Юбилейного форума с международным участием, Москва, 16 ноября 2023 года. – Москва: Институт стратегии развития образования, 2023. – С. 296-308. – EDN MUYJPY.

8. Genially: Программа для создания интерактивного контента [Электронный ресурс]. – URL: https://genially.com (дата обращения: 21.10.2024). – Текст : электронный.

9. Twine: Программа для создания интерактивных текстовых историй [Электронный ресурс]. – URL: https://twinery.org (дата обращения: 21.10.2024). – Текст : электронный.

10. Room Escape Maker: Платформа для создания онлайн-квестов [Электронный ресурс]. – URL: https://roomescapemaker.com (дата обращения: 22.10.2024). – Текст : электронный.

11. Айдин, С. А. Применение информационных технологий в сфере настольных ролевых игр / С. А. Айдин, А. В. Боднар // Информатика и кибернетика. – Донецк: ДонНТУ, 2024. – № 1(35). – С. 24-32. – EDN OBHIFI.

Айдин С.А., Боднар А.В. Применение обучающих игр жанра «квест» в качестве метода интерактивного обучения. *В статье рассматривается такой жанр игр как квест, проводится анализ необходимости применения интерактивных средств обучения в образовательной программе, выделяются ключевые элементы для игр жанра квест, формируются требования к играм данного жанра, рассматриваются существующие программные продукты для создания игр этого жанра. Также в статье выделяются требования для ПО для создания и проигрывания компьютерных обучающих игр, сформированных на основе структуры и требований к играм жанра квест.*

Ключевые слова: *образование, интерактивное обучение, компьютерное обучение, квест, ролевая игра, мотивация, программное обеспечение.*

Aidin S.A., Bodnar A.V. The use of educational «quest» games as a method of interactive learning. *The article examines the genre of «quest» games, analyzes the necessity of implementing interactive learning tools in educational programs, and identifies the key elements of quest games. The article also formulates requirements for games of this genre and reviews existing software tools for creating such games. In addition, the article highlights the requirements for software designed for creating and playing computer-based educational games, which are developed based on the structure and criteria of traditional quest games.*

Keywords: *education, interactive learning, computer teaching, quest, role-playing game, motivation, software.*

Статья поступила в редакцию 15.03.2025
Рекомендована к публикации профессором Мальчевой Р. В.

УДК 004.89

Программная реализация алгоритма генеративных состязательных сетей для задач синтеза изображений

М.О. Лукащук[*1], С.А. Зори[*2]

[*1] магистрант, Донецкий национальный технический университет,
mikhail.lukashchuk@mail.ru

[*2] д.т.н., доцент, Донецкий национальный технический университет,
ik.ivt.rec@mail.ru, OrcID: 0000-0003-4018-234X, SPIN-код: 3565-6330

Аннотация

В статье рассмотрена программная реализация алгоритма генеративных состязательных сетей для задач синтеза изображений. Представлены этапы подготовки данных, проектирования архитектуры генератора и дискриминатора, а также методика обучения модели с использованием комбинированных функций потерь, рассмотрена оценка качества работы генератора. В ходе обучения модели были достигнуты значимые результаты, подтверждённые количественными метриками качества PSNR и SSIM. Перспективами дальнейшего развития являются оптимизация процесса обучения для повышения устойчивости модели, внедрение модификаций архитектуры GAN, а также расширение функциональности системы за счёт поддержки более сложных и разнообразных типов входных данных.

Введение

Современные достижения в области искусственного интеллекта и машинного обучения открывают новые возможности для синтеза и генерации данных [1, 2]. Одним из наиболее революционных направлений в этой области стали генеративные состязательные сети (GAN), которые позволили добиться значительного прогресса в создании фотореалистичных изображений, синтетических видео, текстов, музыки и других видов данных [3].

Принцип работы GAN основан на взаимодействии двух нейронных сетей — генератора и дискриминатора, которые обучаются в процессе состязательной игры. Генератор стремится создавать данные, максимально похожие на реальные, в то время как дискриминатор пытается отличить синтетические данные от настоящих. Благодаря такому подходу обе модели постепенно совершенствуют свои способности, что позволяет добиваться впечатляющих результатов в генеративных задачах.

Реализация алгоритма требует тщательной настройки архитектуры нейросетей, выбора оптимальных методов обучения, обеспечения стабильности сходимости, а также эффективной обработки обучающих данных. Ошибки на любом из этапов могут привести к коллапсу модели или получению низкокачественных результатов.

Целью данной работы является разработка программной реализации базового алгоритма GAN с использованием современных инструментов глубокого обучения и программных моделей алгоритмов GAN.

Теоретические основы алгоритма GAN

Идея алгоритма генеративных состязательных сетей заключается в построении двух нейронных сетей — генератора и дискриминатора, которые обучаются в процессе конкуренции друг с другом. В классическом варианте GAN: Генератор принимает на вход случайный вектор, сгенерированный, например, из нормального распределения, и преобразует его в синтетический пример данных (например, изображение). Дискриминатор принимает на вход либо реальный пример из обучающего набора данных, либо пример, созданный генератором, и должен определить, является ли он подлинным или сгенерированным.

Процесс обучения устроен таким образом, что Генератор стремится "обмануть" дискриминатор, создавая всё более реалистичные данные, а Дискриминатор старается точнее различать реальные и искусственно сгенерированные данные.

Их взаимодействие можно представить как двухстороннюю игру с нулевой суммой, где улучшение одной модели приводит к усложнению задачи для другой.

Проектирование программной реализации алгоритма GAN, выбор инструментария

Программная реализация алгоритма GAN в рамках данного проекта ориентирована на обработку видеофайлов с последующей генерацией новых кадров, имитирующих заданные характеристики. При проектировании

решения особое внимание уделялось модульности кода, обеспечению воспроизводимости экспериментов и удобству последующего масштабирования.

Для реализации программной модели GAN выбран язык программирования Python. В качестве основной платформы для построения и обучения нейронных сетей используется библиотека TensorFlow 2.x [4]. Выбор TensorFlow обусловлен его высокой производительностью, наличием встроенной поддержки работы на GPU и TPU, а также широкими возможностями для масштабирования моделей от локальных экспериментов до промышленного развертывания. Кроме того, TensorFlow обеспечивает стабильную поддержку современных алгоритмов оптимизации и интеграцию с популярными инструментами мониторинга обучения, такими как TensorBoard.

В рамках работы с TensorFlow используется высокоуровневый API Keras [5], который позволяет быстро проектировать и тренировать сложные архитектуры нейронных сетей благодаря удобной модульной структуре и гибкой системе компоновки слоёв. Keras предлагает понятный и лаконичный синтаксис, что значительно ускоряет процесс разработки, минимизирует вероятность ошибок и упрощает модификацию модели в процессе экспериментов.

Для обработки графических данных выбрана библиотека OpenCV [6]. OpenCV предоставляет мощные инструменты для захвата видеопотоков, извлечения и обработки отдельных кадров, изменения размеров изображений, а также их предварительной нормализации. Выбор данной библиотеки обусловлен её высокой производительностью, поддержкой широкого спектра форматов мультимедиа и кроссплатформенностью.

Оценка качества сгенерированных изображений проводилась с использованием метрик PSNR (Peak Signal-to-Noise Ratio) и SSIM (Structural Similarity Index), реализованных в библиотеке scikit-image [7]. Scikit-image представляет собой надёжный и проверенный инструмент для обработки и анализа изображений в Python, предоставляя широкий набор метрик для объективной оценки качества визуальных данных. Применение PSNR и SSIM позволяет количественно оценить степень приближения сгенерированных изображений к реальным, что является главным для оценки эффективности работы генеративной модели.

Архитектура программной системы

Проект состоит из следующих основных модулей:

Извлечение кадров из видео: с помощью OpenCV осуществляется разбивка видеофайла на отдельные изображения.

Предобработка данных: кадры приводятся к фиксированному размеру (256×256 пикселей) и нормализуются в диапазоне [0,1] для оптимальной работы нейросетей.

Построение генератора: последовательная модель с несколькими слоями Conv2DTranspose [8], предназначенная для поэтапного увеличения размерности изображения.

Построение дискриминатора: сверточная нейронная сеть, выполняющая задачу бинарной классификации реальных и сгенерированных изображений.

Определение функций потерь: для дискриминатора используется бинарная кросс-энтропия, для генератора – комбинация контентной потери и состязательной потери.

Цикл обучения: чередующееся обновление генератора и дискриминатора с использованием оптимизаторов Adam [9].

Оценка результатов: вычисление средних значений PSNR и SSIM для оценки качества синтезированных кадров.

Модульная структура кода

Программный код структурирован в логические блоки, обеспечивающие удобство тестирования и масштабирования:

– функция для извлечения кадров;
– функция предобработки кадров;
– построение архитектуры генератора;
– построение архитектуры дискриминатора;
– функции вычисления потерь;
– функция одной итерации обучения;
– функция для оценки качества генерации.

Таким образом, архитектура системы позволяет эффективно реализовать полный цикл работы GAN – от подготовки данных до оценки качества сгенерированных кадров.

Описание алгоритма и реализация

Реализация алгоритма генеративных состязательных сетей требует строгого следования установленной архитектуре взаимодействия между генератором и дискриминатором, правильной организации процесса обучения, а также тщательной предобработки данных. В данной реализации программной системы GAN весь процесс был разделён на несколько последовательных этапов, каждый из которых реализует конкретную функцию.

Извлечение кадров из видео

На первом этапе реализации осуществляется подготовка обучающих данных путём извлечения кадров из видеопотока. Работа с видео предполагает его покадровое считывание

с последующим сохранением каждого отдельного кадра. Это позволяет сформировать статичный набор изображений, которые в дальнейшем будут использоваться для обучения генеративной модели. Выбор этого подхода обусловлен необходимостью работать с реальными данными, а также потребностью в большом объёме разнообразных кадров для достижения высокой генеративной способности сети. Каждый кадр сохраняется в виде отдельного изображения с уникальным именем, обеспечивая корректную индексацию данных. На этом этапе закладывается основа для всех последующих стадий работы алгоритма, поскольку качество исходных данных напрямую влияет на эффективность обучения. Основная функция для этого этапа представлена на рисунке 1.

```
import cv2
import os

def extract_frames(video_path, output_folder):
    # Создание папки для вывода
    os.makedirs(output_folder, exist_ok=True)

    # Открытие файла с видео
    cap = cv2.VideoCapture(video_path)

    frame_count = 0
    while True:
        # Чтение фрейма из видео
        ret, frame = cap.read()

        if not ret:
            break

        # Сохранение фрейма как картинки
        output_path = os.path.join(output_folder, f"frame_{frame_count:04d}.jpg")
        cv2.imwrite(output_path, frame)

        frame_count += 1

    cap.release()
```

Рисунок 1 – Извлечение кадров из видео

Предобработка данных

После извлечения кадров требуется этап их предобработки. Предобработка включает изменение размеров всех кадров до единого стандартизированного формата 256 на 256 пикселей. Это позволяет стандартизировать входные данные и избежать ошибок, связанных с различием размеров изображений. Кроме того, проводится нормализация значений пикселей: значения, изначально находящиеся в диапазоне от 0 до 255, приводятся к диапазону от 0 до 1. Такая нормализация необходима для стабилизации процесса обучения и предотвращения возникновения слишком больших градиентов. Предобработанные изображения сохраняются в отдельную директорию, что обеспечивает возможность многократного и быстрого доступа к ним без необходимости повторной обработки. Этот этап существенно увеличивает скорость обучения моделей и повышает качество итогового синтеза. Основная функция для этого этапа представлена на рисунке 2.

```
def preprocess_frames(input_folder, output_folder, target_size=(256, 256)):
    os.makedirs(output_folder, exist_ok=True)

    for filename in os.listdir(input_folder):
        # Читает фрейм
        frame = cv2.imread(os.path.join(input_folder, filename))

        # Преобразует в нужное разрешение
        frame = cv2.resize(frame, target_size)

        # Нормализует значение пикселей
        frame = frame.astype('float32') / 255.0

        # Сохраняет фрейм
        output_path = os.path.join(output_folder, filename)
        cv2.imwrite(output_path, frame)
```

Рисунок 2 – Предобработка данных

Построение генератора

Следующим шагом является построение архитектуры генератора, который является центральным элементом всей генеративной модели. Генератор предназначен для преобразования случайного шума, полученного из нормального распределения, в реалистичное изображение. Архитектура сети строится таким образом, чтобы постепенно увеличивать пространственное разрешение выходного изображения. На начальном этапе входной вектор шума преобразуется в плотное представление с помощью полносвязного слоя. Далее через последовательность слоёв транспонированной свёртки осуществляется поэтапное увеличение размера изображения. На каждом промежуточном уровне используются функции активации LeakyReLU [10], что позволяет избежать эффекта "затухания" градиентов и способствует более стабильному обучению. Завершающий слой применяет функцию tanh (гиперболический тангенс), обеспечивая нормализацию выходных данных в диапазоне от -1 до 1, что соответствует масштабу нормализованных реальных изображений. Конструкция генератора спроектирована таким образом, чтобы создавать максимально фотореалистичные изображения, которые могли бы "обмануть" дискриминатор. Основная функция для этого этапа представлена на рисунке 3.

```
from tensorflow.keras import layers, models

def build_generator():
    model = models.Sequential()

    # Добавляет несколько слоев
    model.add(layers.Dense(64 * 64 * 256, input_dim=noise_dim))
    model.add(layers.LeakyReLU(alpha=0.2))
    model.add(layers.Reshape((64, 64, 256)))

    model.add(layers.Conv2DTranspose(128, kernel_size=5, strides=2, padding='same'))
    model.add(layers.LeakyReLU(alpha=0.2)

    model.add(layers.Conv2DTranspose(64, kernel_size=5, strides=2, padding='same'))
    model.add(layers.LeakyReLU(alpha=0.2))

    # Выходной слой
    model.add(layers.Conv2DTranspose(3, kernel_size=5, strides=2, padding='same', activation='tanh'))

    return model
```

Рисунок 3 – Построение генератора

Построение дискриминатора

Параллельно с построением генератора проектируется дискриминатор – вторая ключевая составляющая GAN, выполняющая роль "критика" для искусственных данных. Дискриминатор представляет собой сверточную нейронную сеть, которая принимает на вход изображение и должна определить, является ли оно реальным или сгенерированным. Архитектура дискриминатора включает последовательные слои свёртки с последующим применением функций активации LeakyReLU, что способствует улучшению обработки признаков и ускоряет обучение. После нескольких уровней свёртки данные, а затем проходят через полносвязный выходной слой с активацией сигмоиды, который формирует вероятность принадлежности изображения к классу реальных. Основной задачей дискриминатора является обучение максимально точному распознаванию различий между реальными и синтетическими изображениями, что стимулирует генератор к созданию всё более качественных результатов. Процесс построения дискриминатора должен обеспечивать баланс между его способностью различать изображения и сохранением возможностей для генератора получать полезный обучающий сигнал. Основная функция для этого этапа представлена на рисунке 4.

```
def build_discriminator():
    model = models.Sequential()

    model.add(layers.Conv2D(64, kernel_size=5, strides=2, padding='same', input_shape=(256, 256, 3)))
    model.add(layers.LeakyReLU(alpha=0.2))

    model.add(layers.Conv2D(128, kernel_size=5, strides=2, padding='same'))
    model.add(layers.LeakyReLU(alpha=0.2))

    model.add(layers.Flatten())

    model.add(layers.Dense(1, activation='sigmoid'))

    return model
```

Рисунок 4 – Построение дискриминатора

Определение функций потерь

Для успешного обучения генератора и дискриминатора необходима корректная формализация функций потерь (рис. 5). Для дискриминатора в качестве функции потерь используется бинарная кросс-энтропия между предсказанными метками и истинными метками (реальное изображение или поддельное). Это позволяет эффективно наказывать модель за ошибочные классификации и усиливать её способность различать данные. Для генератора формулируется комбинированная функция потерь, состоящая из двух частей. Первая часть — это среднеквадратичная ошибка (MSE) между сгенерированным изображением и желаемым целевым изображением, что позволяет контролировать соответствие содержания. Вторая часть — состязательная потеря, представляющая собой бинарную кросс-энтропию, рассчитываемую на выходе дискриминатора, при которой генератор стремится "обмануть" дискриминатор. Функция представлена на рисунке 6.

```
def discriminator_loss(real_predictions, fake_predictions):
    real_loss = tf.reduce_mean(tf.keras.losses.binary_crossentropy(tf.ones_like(real_predictions), real_predictions))
    fake_loss = tf.reduce_mean(tf.keras.losses.binary_crossentropy(tf.zeros_like(fake_predictions), fake_predictions))
    total_loss = real_loss + fake_loss
    return total_loss
```

Рисунок 5 – Расчет потерь для дискриминатора

```
import tensorflow as tf

# Среднеквадратическое отклонение
def content_loss(desired_frame, generated_frame):
    return tf.reduce_mean(tf.square(desired_frame - generated_frame))

# Бинарная кросс-энтропия
def adversarial_loss(fake_predictions):
    return tf.reduce_mean(tf.keras.losses.binary_crossentropy(tf.ones_like(fake_predictions), fake_predictions))
```

Рисунок 6 – Расчет потерь для генератора

Процесс обучения модели

Обучение генеративных состязательных сетей строится на попеременном обновлении параметров генератора и дискриминатора с целью достижения равновесия между их возможностями. На каждой итерации обучения первым шагом осуществляется обновление дискриминатора. В рамках этого этапа дискриминатор получает на вход реальный кадр из обучающей выборки и сгенерированный кадр, созданный генератором. Для реальных данных дискриминатор должен выдать высокую вероятность подлинности, а для искусственных – низкую. На основании бинарной кросс-энтропии рассчитывается функция потерь дискриминатора, после чего с помощью механизма автоматического дифференцирования вычисляются градиенты, и обновляются веса модели с применением оптимизатора Adam. Вторым шагом в рамках одной итерации является обновление генератора. Генератор создает новый набор изображений на основе случайных векторов шума, передаёт их дискриминатору и получает обратную связь в виде функции потерь (рис. 7). Потери генератора рассчитываются как сумма контентной составляющей (различие между ожидаемым и сгенерированным изображением) и состязательной составляющей (стремление "обмануть" дискриминатор). После расчета потерь также осуществляется вычисление градиентов и обновление параметров генератора с использованием оптимизатора. Такой двухступенчатый процесс позволяет поддерживать динамическое соревнование между двумя сетями, в ходе которого каждая из них стремится улучшить собственную производительность.

Структура тренировочного цикла

Тренировочный цикл организован в виде серии эпох, каждая из которых включает в себя последовательную обработку всех доступных обучающих данных пакетами фиксированного размера. В рамках каждой эпохи для каждого батча проводится полная процедура обучения: обновление дискриминатора и обновление генератора. После завершения обработки всех батчей накапливается статистика потерь, что позволяет контролировать динамику обучения. Для повышения стабильности процесса через определенные интервалы времени осуществляется сохранение весов моделей и промежуточных результатов генерации. Это обеспечивает возможность восстановления обучения в случае прерывания процесса, а также позволяет отслеживать прогресс на различных стадиях тренировки.

Кроме того, предусмотрена система вывода промежуточных метрик потерь, что позволяет оперативно выявлять возможные проблемы, такие как коллапс генератора или чрезмерное усиление дискриминатора. Программная модель цикла представлена на рисунке 8.

```
@tf.function
def train_step(real_frames, desired_frames):
    # Тренируем дискриминатор
    with tf.GradientTape() as disc_tape:
        generated_frames = generator(tf.random.normal([batch_size, noise_dim]))
        real_predictions = discriminator(real_frames)
        fake_predictions = discriminator(generated_frames)
        disc_loss = discriminator_loss(real_predictions, fake_predictions)

    disc_gradients = disc_tape.gradient(disc_loss, discriminator.trainable_variables)
    disc_optimizer.apply_gradients(zip(disc_gradients, discriminator.trainable_variables))

    # Тренируем генератор
    with tf.GradientTape() as gen_tape:
        generated_frames = generator(tf.random.normal([batch_size, noise_dim]))
        fake_predictions = discriminator(generated_frames)
        content_loss = content_loss(desired_frames, generated_frames)
        adv_loss = adversarial_loss(fake_predictions)
        gen_loss = content_loss_weight * content_loss + adv_loss_weight * adv_loss

    gen_gradients = gen_tape.gradient(gen_loss, generator.trainable_variables)
    gen_optimizer.apply_gradients(zip(gen_gradients, generator.trainable_variables))

    return gen_loss, disc_loss
```

Рисунок 7 – Обучение модели

```
for epoch in range(num_epochs):
    for batch_idx in range(len(data) // batch_size):
        # Случайно выбирает батч видео фрейма
        real_frames_batch = sample_real_frames(batch_size)
        desired_frames_batch = sample_desired_frames(batch_size)

        # Одна итерация тренировки
        gen_loss, disc_loss = train_step(real_frames_batch, desired_frames_batch)

    # Лог для мониторинга
    if epoch % 100 == 0:
        print(f"Epoch {epoch}, Generator Loss: {gen_loss}, Discriminator Loss: {disc_loss}")
```

Рисунок 8 – Тренировочный цикл

Оценка качества генерации

Для количественной оценки качества работы генератора используются два основных показателя: PSNR и SSIM. Метрика PSNR измеряет качество восстановления изображения, определяя, насколько сгенерированное изображение близко к эталонному по уровню искажений. Чем выше значение PSNR, тем лучше качество восстановления. Метрика SSIM оценивает структурное сходство между двумя изображениями, принимая во внимание освещенность, контрастность и структуру. Высокие значения SSIM указывают на высокую визуальную идентичность между целевым и сгенерированным изображением.

В процессе оценки на каждом этапе генератор принимает на вход заранее подготовленные контрольные изображения и формирует их сгенерированные версии. Далее для каждой пары изображений рассчитываются значения PSNR и SSIM, после чего выводятся средние значения по всему тестовому набору. Это позволяет объективно оценить прогресс обучения и сравнивать различные версии моделей между собой. Оценка качества генерации представлена на рисунке 9.

```
def evaluate_generator(generator, desired_postures_folder):
    psnr_scores = []
    ssim_scores = []

    for filename in os.listdir(desired_postures_folder):
        # Читает нужный кадр
        desired_frame = cv2.imread(os.path.join(desired_postures_folder, filename))
        desired_frame = cv2.cvtColor(desired_frame, cv2.COLOR_BGR2RGB)

        # Изменение размера и нормализация кадра в соответствии с размером и диапазоном входных данных генератора
        desired_frame = cv2.resize(desired_frame, (256, 256))
        desired_frame = desired_frame.astype('float32') / 255.0

        # Генерирует кадр
        generated_frame = generator(np.random.randn(1, noise_dim)).numpy().squeeze()

        # Вычисляет PSNR и SSIM
        psnr_score = psnr(desired_frame, generated_frame)
        ssim_score = ssim(desired_frame, generated_frame, multichannel=True)

        psnr_scores.append(psnr_score)
        ssim_scores.append(ssim_score)

        # Сохраняет фрейм
        cv2.imwrite(f'generated_postures/{filename}', generated_frame * 255)

    # средние показатели PSNR и SSIM
    mean_psnr = np.mean(psnr_scores)
    mean_ssim = np.mean(ssim_scores)

    return mean_psnr, mean_ssim
```

Рисунок 9 – Оценка качества генерации

Результаты реализации

В ходе реализации проекта была успешно построена и обучена нейросетевая модель, способная синтезировать изображения на основе случайных входных данных. Для количественной оценки качества генерации были применены две объективные метрики: PSNR (Peak Signal-to-Noise Ratio) и SSIM (Structural Similarity Index). Полученные результаты составили:

– PSNR: 18.57;

– SSIM: 0.5620;

Значение PSNR указывает на умеренный уровень визуального шума в сгенерированных изображениях по сравнению с оригиналом, что является типичным результатом для базовой версии GAN без дополнительных методов стабилизации. Показатель SSIM отражает структурное сходство между сгенерированными и реальными изображениями. Значение 0.5620 демонстрирует наличие определённого соответствия между формами, текстурами и пространственными взаимосвязями в изображениях, однако оставляет пространство для дальнейшей оптимизации архитектуры модели и алгоритма обучения. Результатом работа нейросетевой модели является сгенерированное видео на основе обучающих кадров. На рисунке 10 изображен кадр из исходного видео, а на рисунке 11 кадр, полученный на выходе генератора после обучения модели.

Рисунок 10 – Кадр из исходного видео

Рисунок 11 – Кадр из сгенерированного видео

Заключение

В рамках исследования реализован алгоритм генеративных состязательных сетей (GAN) для задач синтеза видео. В статье подробно рассмотрены все этапы построения системы: от подготовки данных и проектирования архитектуры генератора и дискриминатора до определения функций потерь и организации процесса обучения.

Программная реализация выполнена с использованием современных инструментов глубокого обучения, что обеспечило высокую гибкость разработки и стабильность модели на этапах тренировки и применения. В ходе обучения модели достигнуты значимые результаты, подтверждённые количественными метриками качества — PSNR и SSIM.

Перспективами дальнейшего развития проекта являются оптимизация процесса

обучения для повышения устойчивости модели, внедрение модификаций архитектуры GAN, а также расширение функциональности системы за счёт поддержки более сложных и разнообразных типов входных данных.

Литература

1. Мулявин, Д. Е. Расширение возможностей систем генерации изображений путем использования нейронных сетей / Д. Е. Мулявин, Р. В. Мальчева, А. А. Койбаш // Информатика и кибернетика. - Донецк: ДонНТУ, 2024. - № 3 (37). - С. 13-18.

2. Goodfellow, Ian, et al. Generative adversarial nets // Advances in neural information processing systems, 2014.

3. Сметана, В. В. Развитие машинного обучения: от теории к приложениям / В. В. Сметана // Национальная ассоциация ученых. – 2024. – Т. 1. - № 101. – С. 133.

4. TensorFlow. Официальная документация [Электронный ресурс]. – Режим доступа: https://www.tensorflow.org/ – Загл. с экрана.

5. Keras: Deep Learning for Humans [Электронный ресурс]. – Режим доступа: https://keras.io/ – Загл. с экрана.

6. OpenCV: Open Source Computer Vision Library [Электронный ресурс]. – Режим доступа: https://opencv.org/ – Загл. с экрана.

7. scikit-image: Image processing in Python [Электронный ресурс]. – Режим доступа: https://scikit-image.org/ – Загл. с экрана.

8. ConvTranspose2d — PyTorch 2.6 documentation [Электронный ресурс]. – Режим доступа: https://pytorch.org/docs/stable/generated/torch.nn.ConvTranspose2d.html – Загл. с экрана.

9. Adam (Adaptive Moment Estimation) [Электронный ресурс]. – Режим доступа: https://www.geeksforgeeks.org/adam-optimizer-in-tensorflow/ – Загл. с экрана.

10. LeakyReLU — PyTorch 2.6 documentation [Электронный ресурс]. – Режим доступа: https://pytorch.org/docs/stable/generated/torch.nn.LeakyReLU.html – Загл. с экрана.

Лукащук М.О., Зори С.А. Программная реализация алгоритма генеративных состязательных сетей. *В статье рассмотрена программная реализация алгоритма генеративных состязательных сетей для задач синтеза изображений. Представлены этапы подготовки данных, проектирования архитектуры генератора и дискриминатора, а также методика обучения модели с использованием комбинированных функций потерь, рассмотрена оценка качества работы генератора. В ходе обучения модели были достигнуты значимые результаты, подтверждённые количественными метриками качества PSNR и SSIM. Перспективами дальнейшего развития являются оптимизация процесса обучения для повышения устойчивости модели, внедрение модификаций архитектуры GAN, а также расширение функциональности системы за счёт поддержки более сложных и разнообразных типов входных данных.*

Ключевые слова: *синтез видео, GAN, TensorFlow, генератор, дискриминатор, PSNR, SSIM.*

Lukashchuk M.O., Zori S.A. Software Implementation of the Generative Adversarial Network Algorithm. *The article discusses the software implementation of the Generative Adversarial Network (GAN) algorithm for image synthesis tasks. It presents the stages of data preprocessing, the design of the generator and discriminator architectures, and the training methodology using combined loss function, the quality of the generator's output is evaluateds. During the training of the model, significant results were achieved, confirmed by the quantitative quality metrics PSNR and SSIM. The prospects for further development include optimizing the learning process to increase the stability of the model, introducing modifications to the GAN architecture, as well as expanding the functionality of the system by supporting more complex and diverse types of input data.*

Keywords: *video synthesis, GAN, TensorFlow, generator, discriminator, PSNR, SSIM.*

Статья поступила в редакцию 17.03.2025
Рекомендована к публикации профессором Мальчевой Р. В.

УДК 004.032.26

Параметро-эффективное дообучение больших языковых моделей

И. В. Бабич, К. Н. Ефименко
Донецкий национальный технический университет
кафедра «Прикладная математика и искусственный интеллект»
E-mail: babichivanvictorovich@yandex.ru

Аннотация:
В статье приведён обзор теоретической основы parameter-efficient fine-tuning (PEFT) для больших языковых моделей (LLM). Рассмотрены ключевые методы низкоранговых адаптаций (LoRA) и их комбинация с низкобитной квантизацией (QLoRA). Дополнительно приведены рекомендации по выбору ранга адаптации и уровня квантизации на основе спектра Гессиана и эффективной размерности задачи. Успешность использования методов PEFT зависит от правильного выбора параметров, прежде всего ранга адаптации и глубины квантизации.

Введение

Современный этап цифровизации всех сфер жизнедеятельности отличается широчайшим использованием искусственного интеллекта [1-6]. Современные нейронные сети на основе архитектуры трансформеров, известные как большие языковые модели (Large Language Models, LLM), демонстрируют впечатляющие возможности в решении задач обработки естественного языка: генерации текста, автоматическом переводе, ответах на вопросы, извлечении информации и многих других. Высокое качество работы этих моделей достигается путём обучения огромного количества параметров (весов нейронной сети). Например, модель GPT-3 имеет порядка 175 миллиардов параметров.

Общая постановка проблемы

Большой масштаб порождает серьёзную проблему, известную как парадокс масштабирования: для адаптации уже обученной большой языковой модели под конкретную прикладную задачу (например, под медицинские или юридические тексты) необходимо выполнить её дополнительное обучение (дообучение, fine-tuning). При этом, из-за огромного размера LLM (десятки и сотни гигабайт), такое дообучение требует значительных вычислительных ресурсов – сотни графических процессоров (GPU) и большой объём оперативной памяти.

Таким образом, становится актуальной задача поиска параметро-эффективных методов дообучения (Parameter-Efficient Fine-Tuning, PEFT). Под параметро-эффективностью понимается возможность адаптации модели с минимальным изменением её исходных параметров. Формально задача параметро-эффективного дообучения ставится следующим образом:

Пусть имеется большая обученная нейросеть с параметрами (весами) θ_0. Необходимо найти такое минимальное изменение параметров $\Delta\theta$, что:

1. $\|\Delta\theta\| \ll \|\theta_0\|$, то есть изменение весов намного меньше их исходного значения.

2. Новая модель с параметрами $\theta_0+\Delta\theta$ минимизирует целевую функцию ошибки $L(\theta_0+\Delta\theta)$ на специфическом наборе данных.

Целью работы является адаптация модели к новой задаче, минимально затрагивая её исходные параметры и сократив вычислительные затраты на обучение.

Общая концепция и подходы PEFT

Основная идея параметро-эффективного дообучения состоит в том, чтобы ограничить пространство изменений параметров. Вместо того, чтобы изменять все миллиарды параметров исходной модели, мы разрешаем модели изменять лишь небольшую часть или выполнять изменения по определённым правилам. Благодаря этому существенно сокращается объём памяти и вычислительные ресурсы, необходимые для дообучения.

Рассмотрим наиболее распространённые подходы PEFT:

1. Низкоранговые адаптации (Low-Rank Adaptation, LoRA)

LoRA предлагает представлять изменения в весах нейросети в виде специальной матрицы низкого ранга [7]. Что такое низкий ранг? Это значит, что матрица изменений может быть записана как произведение двух матриц меньшего размера. Например, если исходная матрица имеет размер 1000×1000 (1 миллион элементов), то её низкоранговая версия может быть представлена двумя матрицами размером 1000×10 и 10×1000, что в сумме даёт лишь 20 тысяч элементов. Это радикально уменьшает объём параметров для

обучения, сохраняя при этом достаточную гибкость для адаптации.

Формально это записывают как:

$$\Delta W = B \times A,$$

где B и A – небольшие матрицы низкого ранга, которые и будут обучаться вместо исходных весов W.

Таким образом, исходные веса модели остаются нетронутыми, а изменения выражаются через небольшое количество новых параметров.

2. Квантизация весов и адаптация (QLoRA)

Квантизация (англ. quantization) – это способ представления чисел (параметров) с меньшей точностью. Например, вместо 32-битного вещественного числа можно использовать 8-битное или даже 4-битное представление. Это приводит к уменьшению памяти, необходимой для хранения модели. Однако квантизация вносит ошибки округления, которые снижают точность модели. Метод QLoRA совмещает низкоранговые адаптации (LoRA) и квантизацию так, чтобы минимизировать потери точности. Сначала модель сжимается за счёт квантизации (до 4 бит на параметр), а затем поверх неё накладываются небольшие адаптационные матрицы LoRA с более высокой точностью (например, 16 бит), что позволяет компенсировать потери качества, вызванные квантизацией [8].

3. Частичное дообучение (BitFit и IA^3)

Эти методы ещё сильнее ограничивают изменения параметров. Например, в методе BitFit изменяются только отдельные смещения (biases) нейронов, а веса остаются неизменными. В IA^3 изменения ограничены масштабированием активаций, то есть модель изменяет только коэффициенты, на которые умножаются промежуточные значения внутри слоёв сети. Оба подхода дают существенную экономию в ресурсах, хотя и менее гибкие, чем LoRA.

4. Дообучение через префиксы (Prompt/Prefix tuning)

Эти методы вообще не изменяют исходную модель. Вместо этого они добавляют к входу модели небольшие обучаемые последовательности (префиксы), которые подсказывают нейросети, как нужно адаптироваться к новой задаче. Таким образом, параметры самой нейросети вообще не меняются, меняется лишь специальный входной контекст.

Все указанные подходы объединяет одно – они решают задачу адаптации нейронных сетей минимальными средствами. Среди всех методов низкоранговые адаптации LoRA и их вариант с квантизацией QLoRA стали наиболее популярными благодаря оптимальному балансу гибкости и эффективности. В дальнейших разделах статьи мы более глубоко рассмотрим именно их математическое обоснование и теоретические свойства.

Низкоранговая адаптация (LoRA)

Низкоранговая адаптация (Low-Rank Adaptation, LoRA) является одним из наиболее эффективных подходов к параметро-эффективному дообучению больших языковых моделей. Главная идея LoRA заключается в том, что любые изменения в параметрах модели представляются в виде матриц низкого ранга. Рассмотрим, почему это важно.

Весовые матрицы в трансформерах обычно имеют очень большой размер. Например, одна весовая матрица слоя может содержать миллионы или даже сотни миллионов элементов. При стандартном дообучении все эти элементы приходится изменять. LoRA предлагает вместо полного изменения весов добавить к исходной матрице весов небольшую низкоранговую матрицу.

Важное теоретическое преимущество подхода LoRA заключается в его экспрессивности, то есть в способности модели, использующей низкоранговую адаптацию, воспроизводить решения, доступные при полном изменении всех параметров. Если ранг r выбран не меньше, чем так называемый эффективный ранг задачи (определяемый рангом матрицы Гессиана целевой функции ошибки в точке исходного минимума), то низкоранговое решение способно достичь практически того же минимального значения ошибки, что и полное дообучение [7].

Иными словами, LoRA сохраняет «выразительную силу» полной настройки параметров, но значительно экономнее в плане ресурсов, так как существенно ограничивает пространство поиска оптимальных параметров.

LoRA с квантизацией параметров (QLoRA)

Одним из усовершенствований низкоранговой адаптации является подход QLoRA (Quantized Low-Rank Adaptation), который сочетает низкоранговые адаптации и метод квантизации параметров.

Квантизация параметров – это метод представления весов нейронной сети с меньшим количеством бит. Например, вместо стандартного представления весов в формате 32 бита на каждый параметр используются 4-битные веса. Это приводит к восьмикратному сокращению объёма памяти, необходимого для хранения модели [9].

Однако такой метод квантизации вносит неизбежные погрешности округления, называемые квант-шумом. Квант-шум может

существенно ухудшить точность работы нейронной сети, особенно если используется очень низкая точность (например, 4 бита).

Идея QLoRA заключается в следующем:

– исходные веса большой модели сжимаются путём 4-битной квантизации (обычно используется специальный формат NF4, в котором веса представлены в 4-битах).

– поверх этой квантизованной модели накладываются дополнительные низкоранговые адаптеры LoRA в формате с более высокой точностью (обычно 16 бит). Именно эти адаптеры компенсируют потери точности, вызванные квантованием.

Таким образом, QLoRA получает двойную выгоду: сокращает память за счёт квантизации и уменьшает количество обучаемых параметров за счёт низкоранговой структуры. При этом теоретически доказано, что квант-шум остаётся ограниченным (не превышает некоторой небольшой величины), и использование адаптеров позволяет практически полностью восстановить точность исходной модели после квантования.

Теоретические свойства и ограничения PEFT

Методы параметро-эффективного дообучения, такие как LoRA и QLoRA, обладают рядом важных теоретических свойств, которые делают их привлекательными как с практической, так и с теоретической точки зрения (рис. 1).

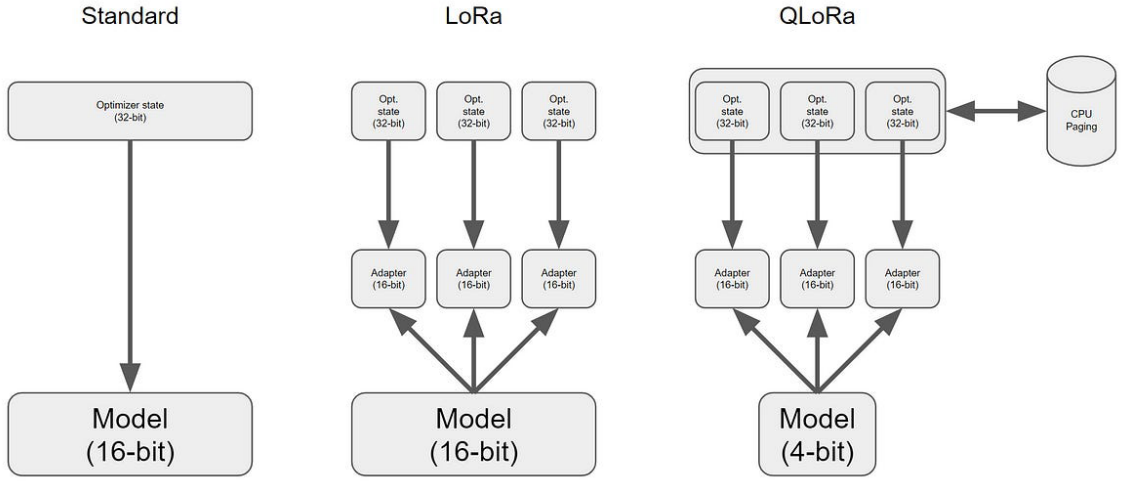

Рисунок 1 – Сравнение схем дообучения LLM: Standard, LoRA и QLoRA

Одним из ключевых преимуществ является то, что использование низкоранговых адаптаций ограничивает сложность модели с точки зрения теории обучения. Концептуально это связано с понятием VC-размера (Vapnik–Chervonenkis dimension), которое характеризует способность модели подстраиваться под любые данные. Чем выше VC-размер, тем больше риск переобучения, то есть модель начинает «запоминать» данные, теряя способность хорошо работать на новых примерах [10].

LoRA и QLoRA снижают VC-размер исходной большой модели за счёт ограничения числа адаптируемых параметров и пространства, в котором происходит обучение. Благодаря этому улучшаются теоретические границы обобщающей способности модели – она становится более устойчивой к переобучению и способна показывать хорошие результаты даже на относительно небольших наборах данных.

Другой важный эффект низкоранговых адаптаций связан с поведением функции ошибки модели. Исследования показывают, что низкоранговые изменения параметров модели чаще всего приводят к относительно «гладкому» изменению функции ошибки. Другими словами, если исходная модель уже находится в локальном минимуме ошибки, то дообучение с использованием небольших низкоранговых адаптеров чаще всего ведёт к минимальному изменению ошибки. Это говорит о том, что модели, адаптируемые через LoRA, находятся в более «плоских» зонах ландшафта ошибок, что также связано с лучшей способностью к обобщению и стабильностью решений.

Однако у подходов PEFT есть и существенные ограничения. Основное ограничение связано с выбором оптимального значения ранга. Слишком маленький ранг не даст модели достаточно гибкости для адаптации, а слишком большой – снизит преимущество метода в плане ресурсоёмкости. Также PEFT-методы не всегда совместимы с более сложными алгоритмами обучения второго порядка

(например, использующими матрицу Гессиана), так как оптимизация в сильно ограниченном пространстве параметров может плохо взаимодействовать с такими методами.

Параметро-эффективные подходы требуют тщательного выбора ограничений и учёта особенностей конкретной задачи, но при правильной реализации способны обеспечить существенные преимущества.

Выбор ранга и уровней квантизации

При практическом использовании низкоранговой адаптации (LoRA) и её комбинации с квантизацией (QLoRA) важнейшим вопросом становится правильный выбор двух основных параметров:

1. Ранга адаптации – насколько большим будет ранг низкоранговой матрицы, отражающей изменения параметров;

2. Уровня квантизации – количества бит, которое используется для хранения каждого параметра модели.

Эти параметры непосредственно влияют на компромисс между качеством и требуемыми ресурсами модели.

При выборе ранга основной идеей является баланс между гибкостью модели (способностью эффективно подстраиваться под задачу) и количеством ресурсов, которые потребуются для её обучения и хранения.

С теоретической точки зрения оптимальный выбор ранга напрямую связан с понятием эффективной размерности задачи. Эффективная размерность определяется тем, насколько разнообразны данные и насколько сложна целевая задача. Формально это связывается с спектром матрицы Гессиана, которая представляет собой вторые производные целевой функции ошибки модели по параметрам. Спектр матрицы Гессиана показывает, в каких направлениях параметров изменения наиболее существенно влияют на ошибку модели, а в каких – почти не влияют [9].

На практике это означает следующее: если спектр Гессиана быстро убывает (то есть имеется небольшое число «сильных» направлений, по которым изменения веса значительно снижают ошибку), то эффективная размерность задачи мала. Тогда для адаптации достаточно выбрать небольшой ранг. Если же спектр Гессиана убывает медленно, и эффективная размерность задачи велика (сложная задача, разнообразные данные), то необходим больший ранг.

Таким образом, на практике рекомендуют начинать адаптацию с относительно небольшого ранга (например, 8 или 16). Если качество модели оказывается неудовлетворительным, постепенно увеличивайте ранг до момента, когда качество начнёт заметно улучшаться. Обычно это значение будет оптимальным компромиссом. Для большинства практических задач средние значения ранга в пределах 32-64 обеспечивают хороший баланс между эффективностью и качеством.

При использовании QLoRA модель дополнительно сжимается за счёт низкобитного представления параметров. Ключевой параметр – число бит, выделяемых на параметр, влияет на компромисс между экономией памяти и точностью работы модели.

32-битное представление – стандартный формат хранения параметров, не вносит дополнительных ошибок, но требует много памяти.

8-битная квантизация – является распространённым и безопасным выбором, поскольку почти не ухудшает качество моделей и экономит в четыре раза больше памяти.

4-битная квантизация (NF4) – обеспечивает ещё большую экономию (до восьми раз по сравнению с 32-битным форматом), однако здесь уже необходимо учитывать наличие заметного квантового шума и компенсировать его увеличением ранга адаптации.

Для типовых приложений, где ресурсы ограничены, но требуется высокое качество, хорошим стартом является 8-битная квантизация. Если стоит задача максимально снизить потребление ресурсов (например, адаптация модели на мобильных устройствах), то переходят к 4-битному формату. При этом важно увеличить ранг адаптации (например, с 16-32 до 64-128), чтобы минимизировать падение точности, вызванное квантованием.

Выводы

Рассмотренные методы параметро-эффективного дообучения больших языковых моделей, в частности низкоранговая адаптация (LoRA) и её комбинация с квантизацией (QLoRA), обеспечивают эффективный компромисс между сложностью и гибкостью адаптации моделей. Их главная особенность заключается в ограничении пространства параметров, что позволяет значительно сократить объём вычислительных ресурсов и памяти, необходимых для дообучения.

С теоретической точки зрения ключевыми преимуществами LoRA и QLoRA являются высокая экспрессивность, благодаря которой модель сохраняет способность достигать оптимального минимума ошибки, и улучшенные свойства обобщения за счёт снижения VC-размера модели. Также важно отметить, что низкоранговые адаптации способствуют гладкости ландшафта ошибки, что улучшает устойчивость модели к изменениям данных и

уменьшает вероятность переобучения.

Тем не менее, успешность использования методов PEFT зависит от правильного выбора параметров, прежде всего ранга адаптации и глубины квантизации. Поэтому дальнейшие теоретические исследования должны быть направлены на более точное понимание связи между рангом адаптации, уровнем квант-шумов и способностью модели сохранять высокие показатели качества при минимальных вычислительных затратах.

Литература

1. Федяев, О. И. Интеллектуальная система принятия решений в отделении медицинского учреждения на основе нейросетевых, продукционных и статистических моделей / О. И. Федяев, В. С. Бакаленко // Статистика и Экономика, 2019. – № 16(3). – С. 70-77. – URL: https://doi.org/10.21686/2500-3925-2019-3-70-77

2. Дворяткина, С. Н. Интеграция фрактальных и нейросетевых технологий в педагогическом контроле и оценке знаний обучаемых / С. Н. Дворяткина // Вестник РУДН. Серия : Психология и педагогика, 2016. - Том. 14. - № 4. – С. 451-465.

3. Федяев, О. И. Прогнозирование остаточных знаний студентов по отдельным дисциплинам с помощью нейронных сетей / О. И. Федяев // Известия ЮФУ. Технические науки. – 2016. – С. 122-136.

4. AI в обучении: на что способны технологии уже сейчас? // EduTech. - 2022. - № 4 [49]. – 60 с.

5. Струнин, Д. А. Искусственный интеллект в сфере образования / Д. А. Струнин. — Текст : непосредственный // Молодой ученый. — 2023. — № 6 (453). — С. 15-16. — URL: https://moluch.ru/archive/453/99921/

6. Федяев, О. И. Автоматическая регистрация присутствия студентов на учебном занятии с помощью компьютерного зрения / О. И. Федяев, И. А. Коломойцева // XXI Национальная конференция по искусственному интеллекту с международным участием КИИ-2023 (Смоленск, 16-20 октября 2023 г.). Труды конференции. В 2-х томах. Т.1. – Смоленск: Принт-Экспресс, 2023. – С. 294-303.

7. Портал huggingface.co [Электронный ресурс] / Интернет-ресурс. – Режим доступа: https://huggingface.co/docs/text-generation-inference/conceptual/lora. – Загл. с экрана.

8. Портал huggingface.co [Электронный ресурс] / Интернет-ресурс. – Режим доступа: https://huggingface.co/blog/4bit-transformers-bitsandbytes. – Загл. с экрана.

9. Портал geeksforgeeks.org [Электронный ресурс] / Интернет-ресурс. – Режим доступа: https://www.geeksforgeeks.org/what-is-qlora-quantized-low-rank-adapter/. – Загл. С экрана.

10. Портал ru.wikipedia.org [Электронный ресурс] / Интернет-ресурс. – Режим доступа: https://ru.wikipedia.org. – Загл. с экрана.

Бабич И.В., Ефименко К.Н. Параметро-эффективное дообучение больших языковых моделей. *В статье приведён обзор теоретической основы parameter-efficient fine-tuning (PEFT) для больших языковых моделей (LLM). Рассмотрены ключевые методы низкоранговых адаптаций (LoRA) и их комбинация с низкобитной квантизацией (QLoRA). Дополнительно приведены рекомендации по выбору ранга адаптации и уровня квантизации на основе спектра Гессиана и эффективной размерности задачи. Успешность использования методов PEFT зависит от правильного выбора параметров, прежде всего ранга адаптации и глубины квантизации.*

Ключевые слова: *параметро-эффективное дообучение, PEFT, LoRA, QLoRA, квантизация, спектр Гессиана.*

Babich I., Efimenko K. Parameter-Efficient Fine-Tuning of Large Language Models. *This paper provides an overview of the theoretical foundations of parameter-efficient fine-tuning (PEFT) for large language models (LLM). It examines the key methods of low-rank adaptation (LoRA) and their combination with low-bit quantization (QLoRA). In addition, it offers recommendations for selecting the adaptation rank and quantization level based on the Hessian spectrum and the intrinsic dimension of the task. The success of using PEFT methods depends on the correct choice of parameters, primarily the adaptation rank and the depth of quantization.*

Keywords: *parameter-efficient fine-tuning, PEFT, LoRA, QLoRA, quantization, Hessian spectrum.*

Статья поступила в редакцию 12.03.2025
Рекомендована к публикации профессором Павлышом В. Н.

Компьютерные науки

УДК 004.421.2

Теорема Хана-Банаха – как одна из основных теорем функционального анализа и ее практическое применение

Ю. Н. Добровольский

Донецкий национальный технический университет, г. Донецк
dyn_don_20.14@mail.ru

Аннотация
Рассмотрена история возникновения теоремы Хана-Банаха. Введены основные линейные нормированные пространства, линейные непрерывные функционалы и их нормы. Введено понятие гиперподпространство, гиперплоскость. Приведены примеры на вычисление нормы функционала и продолжение функционала на все пространство с сохранением нормы. Линейные функционалы и их продолжение на все функциональное пространство с сохранением нормы является мощным математическим аппаратом для развития многих направлений современной математики.

Введение

В начале 20-го века, этап развития классической математики был полностью завершен. Начинался этап развития современной математики.

Необходимо было проанализировать и обобщить накопленные человечеством знания по математике. Перед учеными математиками стояли новые идеи и задачи, посмотреть на изученные вещи иначе, с общей точки зрения. Выяснить, существует ли связь между различными математическими объектами, или другими словами, не вникая в природу математического объекта, изучить его свойства. Это было оправдано тем, что такая связь существует и различные математические объекты имеют общие свойства.

Некоторые положения, которые легли в основу функционального анализа: Обобщение понятия пространства, развитие и обобщение понятия функции, создание понятия функционального пространства.

Также на развитие функционального анализа оказали влияние физические теории, например квантовая механика и квантовая теория поля. Многие понятия функционального анализа широко использовались физиками задолго до того, как им было дано строгое математическое обоснование.

Высокая степень абстракции вводимых понятий делает функциональный анализ довольно сложным. Именно абстрактность позволяет исследовать далекие на первый взгляд друг от друга вопросы и успешно применять его методы в различных других дисциплинах.

В функциональном анализе теорема Хана-Банаха является основным результатом, который позволяет расширить ограниченные линейные функционалы, определенные на векторном подпространстве некоторого векторного пространства, на все пространство.

Теорема также показывает, что существуют непрерывные линейные функционалы, определенные на каждом нормированном векторном пространстве, для изучения двойственного пространства.

Другая версия теоремы Хана-Банаха известна как теорема **Хана-Банаха о разделении** или теорема о разделении гиперплоскостями и имеет множество применений в выпуклой геометрии.

Актуальность данной темы заключается в том, что теорема Хана-Банаха является эффективным теоретическим средством в практическом решении конкретных задач современной теоретической и прикладной математики.

Тема исследования: Линейные непрерывные функционалы, теорема Хана-Банаха.

Объект исследования: линейные функционалы, теорема Хана-Банаха.

Предмет исследования: определение нормы линейного непрерывного функционала, продолжение функционала на все пространство с сохранением нормы.

Цель исследования: дать теоретическое обоснование понятий функционал, продолжение функционала при практическом их применении.

Задачи исследования:

1. Рассмотреть линейные непрерывные функционалы, заданные на линейных нормированных пространствах.
2. Изучить способы определения нормы функционала, дать ему геометрическую интерпретацию.

3. Усвоить практическое применение теоремы Хана-Банаха.

Для решения поставленных задач использовался комплекс методов исследования:

– обобщение изученных ранее данных;

– анализ научно-методической литературы;

– систематизация информации.

Научная новизна: расширение возможностей функционального анализа для решения задач современной науки: некоммутативный функциональный анализ, теория операторных алгебр, вероятностный и стохастический функциональный анализ, функциональный анализ на пространствах фрактальных структур.

Практическое значение: материал, рассмотренный в данной статье, может быть использован для решения задач математического и функционального анализа.

Структура исследования: работа состоит из введения, шести разделов, заключения, списка использованной литературы.

1. История возникновения теоремы Хана-Банаха

Большой вклад в историю развития теоремы Хана-Банаха внесли известные ученые: австрийский математик **Эдуард Хелли** (1884-1943); венгерский и шведский математик **Марсель Рис** (1886-1969).

История теоремы **Хана-Банаха** несколько необычна. **Эдуард Хелли** считается одним из основоположников функционального анализа, но теорема названа в честь австрийского математика **Ганса Хана** (1879-1934) и польского математика **Стефана Банаха** (1892-1945), которые независимо друг от друга доказали ее в конце 1920-х годов.

Тем не менее, теорема возникла в результате работ других учёных:

Эдуард Хелли в 1912 году доказал частный случай теоремы для пространства непрерывных функций на интервале.

Марсель Рис в 1923 году доказал теорему о расширении, из которой можно вывести теорему Хана-Банаха.

Теорема Хана-Банаха утверждает, что любой ограниченный линейный функционал, определённый на подпространстве нормированного пространства, можно продолжить на все пространство с сохранением нормы.

Доказательства теоремы Хана-Банаха используют разные методы, например:

Метод Хелли. Хелли показал, что некоторые линейные функционалы, определённые в подпространстве определённого типа нормированного пространства, имеют расширение той же нормы, используя индукцию.

Метод Хана. В 1927 году Хан определил общие банаховы пространства и применил метод Хелли для доказательства сохраняющей норму версии теоремы Хана-Банаха для банаховых пространств.

Обобщение Банаха. В 1929 году Банах, который не знал о результате Хана, обобщил его, заменив версию, сохраняющую норму, версией с доминирующим расширением, использующей сублинейные функции.

Теорема Хана-Банаха имеет множество приложений в функциональном анализе, например:

Решение бесконечных систем уравнений. Это необходимо для решения таких задач, как задача о моментах, в которой, зная все потенциальные моменты функции, нужно определить, существует ли функция с такими моментами, и если да, то найти ее по этим моментам. Другой подобной задачей является задача о косинусном ряде Фурье, в которой, зная все потенциальные коэффициенты косинуса Фурье, нужно определить, существует ли функция с такими коэффициентами, и если да, то найти ее.

2. Непрерывные линейные функционалы

В данной статье будут рассмотрены следующие линейные нормированные пространства.

1. $C[a,b]$ – пространство всех непрерывных на отрезке $[a,b]$ функций с нормой

$$\|x\| = \max_{t\in[a,b]} |x(t)|.$$

2. l_p – пространство всех числовых последовательностей, таких, что $\left(\sum_{k=1}^{\infty} |x_k|^p\right)^{\frac{1}{p}} < \infty, 1 \le p < \infty$ с нормой

$$\|x\| = \left(\sum_{k=1}^{\infty} |x_k|^p\right)^{\frac{1}{p}}.$$

3. $L_p[a,b]$ – пространство всех функций, определенных на отрезке $[a,b]$, для которых интеграл Лебега $\int\limits_{[a,b]} |x(t)|^p \, dt < \infty$ – конечен, с нормой

$$\|x\| = \left(\int\limits_{[a,b]} |x(t)|^p \, dt\right)^{\frac{1}{p}}.$$

4. $M[a,b]$ – пространство всех ограниченных на $[a,b][$ функций с нормой $\|x\| = \sup\limits_{t\in[a,b]} |x(t)|$.

5. l_∞^n – пространство всех упорядоченных наборов из n действительных чисел.

$x = (x_1, \ldots, x_n), n \geq 1$. Норма определяется по формуле $\|x\| = \max\limits_{1 \leq k \leq n} |x_k|$.

6. $l_p^n, 1 \leq p < \infty$ – пространство всех упорядоченных наборов из *n* действительных чисел $x = (x_1, \ldots, x_n), n \geq 1$. Норма определяется с помощью равенства

$$\|x\| = \left(\sum_{k=1}^{n} |x_k|^p \right)^{\frac{1}{p}}.$$

На этих линейных нормированных пространствах будут заданы линейные непрерывные функционалы.

Определение 1. Пусть *L* – линейное нормированное пространство. Отображение *f*, действующее из *L* в *R*, будем называть **функционалом.**

Определение 2. Функционал *f* называется линейным, если он аддитивен, то есть для всех l_1, l_2 из *L*

$$f(l_1 + l_2) = f(l_1) + f(l_2),$$

и однороден, то есть для всех $l \in L$ и любых вещественных чисел λ

$$f(\lambda l) = \lambda f(l).$$

Множество $\ker f = \{x \in L : f(x) = 0\}$ называется **ядром** функционала *f*.

Определение 3. Функционал *f* называется **непрерывным** в точке $x_0 \in L$, если для любого $\varepsilon > 0$ существует $\delta > 0$, что для всех *x* таких, что $\|x - x_0\|_L < \delta$ выполняется неравенство $|f(x) - f(x_0)| < \varepsilon$.

На практике чаще применяют эквивалентное определение непрерывности:

Определение 4. Функционал *f* называется **непрерывным** в точке $x_0 \in L$, если для любой последовательности $\{x_n\}$, сходящейся к x_0, $f(x_n)$ сходится к $f(x_0)$.

Верно утверждение: Если линейный функционал непрерывен в какой-либо одной точке $x_0 \in L$, то он непрерывен на *L*.

Определение 5. Линейный функционал *f*, заданный на нормированном пространстве *L*, называется **ограниченным,** если существует такая постоянная $c > 0$, что для всех элементов $x \in L$ выполняется равенство

$$|f(x)| \leq c \|x\|_L.$$

Можно показать, что из непрерывности функционала следует его ограниченность, верно и обратное.

3. Геометрический смысл линейного функционала

Определение 1. Пусть *L* – линейное пространство. L_0 – линейное подпространство *L*. L_0 называется **гиперподпространством**, если существует элемент $x_0 \notin L_0$ такой, что $L = lin(x_0, L_0)$. Говорят, что L_0 имеет коразмерность 1. Например, если *L* – *n*-мерное пространство, то гиперподпространство – это (n-1)-мерное подпространство.

Принимаем без доказательства, что верны следующие **утверждения:**

1. Ядро функционала *ker f* является гиперподпространством. Здесь *f* – линейный функционал.

2. Пусть f_1 и f_2 – линейные функционалы, заданные на *L*. Тогда, если $\ker f_1 = \ker f_2$, то $f_1 = \lambda f_2$ для некоторого числа λ.

Определение 2. Пусть L_0 – гиперподпространство в *L*. Множество $L_1 = L_0 + x$ называется **гиперплоскостью.** Здесь $x \notin L_0$. Иными словами, гиперплоскость – это множество, получающееся из гиперподпространства параллельным переносом (сдвигом) на какой-нибудь вектор $x \in L$. Например, если $L=R^3$, то гиперплоскость – это плоскость, не проходящая через начало координат.

Справедливы следующие **утверждения:**

Утверждение 3.1.

$$M = \{x \in L : f(x) = c, c \neq 0\}$$

является гиперплоскостью.

Утверждение 3.2.

Множество $M \subset L$ является гиперплоскостью тогда и только тогда, когда *M* можно представить в виде $M = \{x \in L : f(x) = 1\}$, где *f* – линейный функционал, причем *f* определяется однозначно.

Таким образом, можно установить взаимно однозначное соответствие между всеми нетривиальными линейными функционалами, определенными на *L*, и всеми гиперплоскотями в *L*.

Можно показать, что линейный функционал в нормированном пространстве непрерывен тогда и только тогда, когда его ядро замкнуто.

Любая гиперплоскость в линейном нормированном пространстве или замкнута, или плотна в нем, а так же является выпуклым множеством.

Пусть *f* – линейный функционал, определенный на линейном нормированном пространстве *L*.

Можно показать, что *f* непрерывен тогда и только тогда, когда множества $\{x \in L : f(x) < c\}$ и $\{x \in L : f(x) > c\}$ являются открытыми в пространстве *L*.

4. Норма функционала

Пусть *f* – линейный непрерывный функционал, заданный на нормированном пространстве *L*. Так как *f* является ограниченным функционалом, то существует постоянная *M* такая, что

$$\left|f(x)\right| \le M\left\|x\right\|. \qquad (4.1)$$

Определение 1. Наименьшая из постоянных *M*, удовлетворяющая неравенству (4.1), называется **нормой** и обозначается $\left\|f\right\|$.

Таким образом,

$$\left|f(x)\right| \le \left\|f\right\|\left\|x\right\|. \qquad (4.2)$$

Отметим некоторые свойства нормы функционала

1. $\left\|f\right\| = \sup\limits_{x\neq 0}\dfrac{\left|f(x)\right|}{\left\|x\right\|}$.

2. $\left\|f\right\| = \sup\limits_{\|x\|\le 1}\left|f(x)\right| = \sup\limits_{\|x\|=1}\left|f(x)\right|$.

Норма функционала обладает всеми свойствами нормы, а именно

1) $\left\|f\right\| \ge 0, \left\|f\right\| = 0 \Leftrightarrow f = 0;$

2) $\left\|\lambda f\right\| = \left|\lambda\right|\left\|f\right\|;$

3) $\left\|f+g\right\| \le \left\|f\right\| + \left\|g\right\|.$

Приведем примеры вычисления нормы функционала.

Пример 4.1. Вычислить норму функционала $f(x) = x(1) - 2x(2)$, заданного в пространстве $C[0,2]$.

Решение. В общем случае имеем пространство C[a,b]. По условию $a = 0, b = 2.$ Норма

$$\left\|x\right\| = \max_{t\in[0,2]}\left|x(t)\right|.$$

Используя неравенство треугольника и определение нормы в пространстве $C[0,2]$, имеем

$$\left|f(x)\right| \le \left|x(1)\right| + 2\left|x(2)\right| \le 3\left\|x\right\|_{C[0,2]}.$$

Так как $\left\|f\right\|$ – это наименьшая константа, то $\left\|f\right\| \le 3$. Чтобы получить противоположное по знаку неравенство, воспользуемся тем, что $\left\|f\right\| \ge \dfrac{\left|f(x)\right|}{\left\|x\right\|_{C[0,2]}}$ для любого $x \neq 0$. Тогда

$$\left\|f\right\| \ge \frac{\left|x(1) - 2x(2)\right|}{\max\limits_{t\in[0,2]}\left|x(t)\right|}.$$

Выберем непрерывную функцию *x* так, чтобы $\max\limits_{t\in[0,2]}\left|x(t)\right| = 1$ и $x(1) = 1$, и $x(2) = -1$. Такие функции существуют. Для этого нужно соединить точки (1,1) и (2,-1) так, чтобы не выйти за пределы полосы, определяемой прямыми *y*=1, *y*=-1. Тогда $\left\|f\right\| \ge 3$ и вместе с ранее полученным неравенством имеем $\left\|f\right\| = 3$.

Пример 4.2. Вычислить норму функционала $f(x) = 2x_1 - 3x_3$, заданного в пространстве l_4.

Решение.

Имеем пространство l_p. По условию *p*=4. Тогда $\left(\sum\limits_{k=1}^{\infty}\left|x_k\right|^4\right)^{\frac{1}{4}} < \infty,$ норма

$$\left\|x\right\| = \left(\sum_{k=1}^{\infty}\left|x_k\right|^4\right)^{\frac{1}{4}}.$$

Имеем отображение:

$$f : \left\{x = (x_1, x_2 \ldots)\ (x_k \in \square)\right\} \to \left\{2x_1 - 3x_3\right\}.$$

Воспользуемся неравенством Гельдера:

$$\left|\sum_{i=1}^{\infty} a_i b_i\right| \le \left(\sum_{i=1}^{\infty}\left|a_i\right|^p\right)^{\frac{1}{p}} \cdot \left(\sum_{i=1}^{\infty}\left|b_i\right|^q\right)^{\frac{1}{q}} \qquad (4.3)$$

где $\dfrac{1}{p} + \dfrac{1}{q} = 1;\ p, q \ge 1$.

Взяв

$$p = 4,\ q = \frac{4}{3},\ a_1 = x_1,\ a_2 = 0, a_3 = x_3,\ b_1 = 2,$$

$$b_2 = 0,\ b_3 = -3,\ a_i = b_i = 0,\ i \ge 4,$$

получим

$$\left|f(x)\right| = \left|2x_1 - 3x_3\right| \le \left(\left|2\right|^{\frac{4}{3}} + \left|-3\right|^{\frac{4}{3}}\right)^{\frac{3}{4}} \cdot \left(\left|x_1\right|^4 + \left|x_3\right|^4\right)^{\frac{1}{4}} \le$$

$$\le c \cdot \left\|x\right\|_{l_4}.$$

Здесь $c = \left(2^{\frac{4}{3}} + 3^{\frac{4}{3}}\right)^{\frac{3}{4}}$.

Из определения нормы функционала следует, что $\left\|f\right\| \le c$. С другой стороны, для любого $x \in l_4\ (x \neq 0)$ выполняется неравенство $\left\|f\right\| \ge \dfrac{\left|f(x)\right|}{\left\|x\right\|_{l_4}}$, т.к. $\left|f(x)\right| \le \left\|f\right\|\left\|x\right\|$. Подберем последовательность $\hat{x}$ так, чтобы $\dfrac{\left|f(\hat{x})\right|}{\left\|\hat{x}\right\|_{l_4}} = c$.

Для этого учтем, что неравенство Гельдера (4.3) обращается в равенство, если $b_i = \left|a_i\right|^{p-1} \cdot \operatorname{sgn} a_i$.

Взяв $p=\frac{4}{3}, q=4, a_1=2, a_3=-3, a_i=0, i\neq 1,3$ и

$$\hat{x}_i=\begin{cases}2^{\frac{1}{3}}, & i=1;\\ -3^{\frac{1}{3}}, & i=3;\\ 0, & \textit{в остальных случаях},\end{cases}$$

получим $f\left(\hat{x}\right)=2^{\frac{4}{3}}+3^{\frac{4}{3}}, \left\|\hat{x}\right\|_{l_4}=\left(2^{\frac{4}{3}}+3^{\frac{4}{3}}\right)^{\frac{1}{4}}$.

Тогда $\frac{\left|f(\hat{x})\right|}{\left\|\hat{x}\right\|_{l_4}}=c$. Окончательно имеем $\|f\|=c$

Пример 4.3. Вычислить норму функционала $f(x)=\int\limits_{[-1,1]} tx(t)dt$, заданного в пространстве $L_3[-1,1]$.

Решение. Имеем пространство $L_p[a,b]$. По условию *p*=3, *a*=-1, *b*=1.

Интеграл Лебега $\int\limits_{[-1,1]}\left|x(t)\right|^3 dt<\infty$, с нормой $\|x\|=\left(\int\limits_{[-1,1]}\left|x(t)\right|^3 dt\right)^{\frac{1}{3}}$.

Имеем отображение:

$$f:\{x=x(t)\}\to\left\{\int\limits_{[-1,1]} tx(t)dt\right\}$$

Используем интегральное неравенство Гельдера:

$$\left|\int\limits_{[a,b]} y(t)x(t)dt\right|\le\left(\int\limits_{[a,b]}\left|y(t)\right|^p dt\right)^{\frac{1}{p}}\left(\int\limits_{[a,b]}\left|x(t)\right|^q dt\right)^{\frac{1}{q}} \quad (4.4)$$

Здесь $\frac{1}{p}+\frac{1}{q}=1;\ p,\ q\ge 1$.

Взяв $p=\frac{3}{2},\ q=3,\ y(t)=t$, получим

$$\left|f(x)\right|\le\left(\int\limits_{[-1,1]}\left|t\right|^{\frac{3}{2}} dt\right)^{\frac{2}{3}}\left(\int\limits_{[-1,1]}\left|x(t)\right|^3 dt\right)^{\frac{1}{3}}.$$

Вычислив первый интеграл, получим $\|f\|\le\left(\frac{4}{5}\right)^{\frac{2}{3}}$. С другой стороны, учтем, что неравенство Гельдера (4.4) обращается в равенство, если $x(t)=\left|y(t)\right|^{p-1}\operatorname{sgn} y(t)$. Выберем $\hat{x}(t)=t^{\frac{1}{2}}\operatorname{sgn} t$. Тогда $f(\hat{x})=\int\limits_{[-1,1]}\left|t\right|^{\frac{3}{2}} dt=\frac{4}{5}$, а $\left\|\hat{x}\right\|_{L_3[-1,]}=\left(\int\limits_{[-1,1]}\left|t\right|^{\frac{3}{2}} dt\right)^{\frac{1}{3}}=\left(\frac{4}{5}\right)^{\frac{2}{3}}$. Так как $\|f\|\ge\frac{\left|f(\hat{x})\right|}{\|x\|}=\left(\frac{4}{5}\right)^{\frac{2}{3}}$, то вместе с доказанным ранее неравенством получим $\|f\|=\left(\frac{4}{5}\right)^{\frac{2}{3}}$.

Пример 4.4. Вычислить норму функционала $f(x)=\int\limits_{-1}^{1} tx(t)dt$, заданного в пространстве *C*[-1,1].

Решение.

Имеем пространство C[a,b] непрерывных на [a,b] функций с нормой

$$\|x\|=\max_{t\in[a,b]}\left|x(t)\right|.$$

Напишем цепочку неравенств:

$$\left|f(x)\right|=\left|\int\limits_{-1}^{1} tx(t)dt\right|\le\int\limits_{-1}^{1}\left|t\right|\left|x(t)\right|dt\le$$

$$\le\max_{t\in[-1,1]}\left|x(t)\right|\int\limits_{-1}^{1}\left|t\right|dt=\|x\|_{C[-1,1]}\int\limits_{-1}^{1}\left|t\right|dt.$$

Вычисляя интеграл $\int\limits_{-1}^{1}\left|t\right|dt=1$, получаем $\|f\|\le\|x\|_{C[-1,1]}$. Тогда $\|f\|\le 1$. Для получения противоположного по знаку неравенства попробуем подобрать непрерывную функцию $\hat{x}$ так, чтобы $\frac{\left|f(\hat{x})\right|}{\left\|\hat{x}\right\|_{C[-1,1]}}=1$. Все попытки найти такую непрерывную функцию оказываются безуспешными. Это объясняется тем, что она не существует. Однако если взять функцию $\hat{x}(t)=\operatorname{sgn}(t)$ и рассмотреть функционал *f* на пространстве M[-1,1] – ограниченных на [-1,1]

функций , то $\frac{\left|f(\hat{x})\right|}{\left\|\hat{x}\right\|_{C[-1,1]}}=1$. Но функция $\hat{x}$ разрывная, а значит, не принадлежит пространству C[-1,1].

Поступим следующим образом. Найдем последовательность непрерывных функций x_n таких, что $x_n(t)\to\hat{x}(t)$ для каждого $t\in[-1,1]$. Для построения последовательности x_n нужно "подправить" функцию $\hat{x}(t)=\operatorname{sgn}(t)$ в месте разрыва. Имеем

$$x_n(t)=\begin{cases}1, & \frac{1}{n}\le t\le 1;\\ nt, & -\frac{1}{n}\le t\le\frac{1}{n};\\ -1, & -1\le t\le-\frac{1}{n}.\end{cases}$$

Вычислим $f(x_n)$. Воспользуемся тем, что функция $tx_n(t)$ является четной.

Тогда

$$f(x_n)=2\left(\int_0^{\frac{1}{n}}nt^2dt+\int_{\frac{1}{n}}^{1}tdt\right)=1-\frac{1}{3n^2}.$$

Так как $\left\|x_n\right\|_{C[-1,1]}=1$, то

$$\left\|f\right\|\ge\frac{\left|f(x_n)\right|}{\left\|x_n\right\|}=1-\frac{1}{3n^2}.$$

Если n устремить к бесконечности, то получим $\left\|f\right\|\ge 1$. Итак, $\left\|f\right\|=1$.

В отличие от предыдущих примеров мы не смогли найти функцию $\hat{x}$ такую, что $\left\|f\right\|=\frac{\left|f(\hat{x})\right|}{\left\|x\right\|}$.

5. Геометрическая интерпретация нормы функционала.

Утверждение 5.1. Пусть f – линейный непрерывный функционал, заданный на нормированном пространстве L. Тогда

$$\left\|f\right\|=\frac{1}{\inf\limits_{x\in L_f}\left\|x\right\|};\qquad(5.1)$$

здесь $L_f=\left\{x\in L: f(x)=1\right\}$.

Таким образом, чтобы найти норму функционала, нужно построить гиперплоскость L_f и найти расстояние d от нуля до этой гиперплоскости. Тогда $\left\|f\right\|=\frac{1}{d}$.

Покажем, как можно найти норму функционала, используя формулу (5.1).

Пример 5.1. Вычислить норму функционала $f(x)=3x_1-4x_2$, заданного в пространстве l_∞^2.

Решение.

В общем случае имеем пространство l_∞^n, элементами которого являются упорядоченные наборы из n действительных чисел. Норма определяется по формуле

$$\left\|x\right\|=\max_{1\le k\le n}\left|x_k\right|.$$

В нашем случае n=2. На плоскости построим прямую $3x_1-4x_2=1$. Чтобы найти расстояние от нуля до этой прямой, нарисуем шар с центром в нуле радиуса r так, чтобы шар не пересекал данную прямую. Теперь увеличиваем размер шара до тех пор, пока он не коснется прямой. Радиус этого шара – это и есть расстояние от нуля до L_f. Напомним, что $\left\|x\right\|_{l_\infty^2}=\max\left\{\left|x_1\right|,\left|x_2\right|\right\}$, поэтому шар $B[0,r]$ – это квадрат с центром в нуле, стороны которого параллельны координатным осям. Длина стороны квадрата равна $2r$. В нашем случае касание произойдет в точке с координатами $(r,-r)$. Так как точка $(r,-r)$ лежит на прямой, то $3r+4r=1$ и $d=\frac{1}{7}$. Окончательно имеем $\left\|f\right\|=7$.

6. Теорема Хана – Банаха

Пусть L – линейное нормированное пространство. L_0 – некоторое его подпространство. Пусть далее на L_0 задан линейный непрерывный функционал f_0. Функционал f называется **продолжением** функционала f_0, если $f(x)=f_0(x)$ для всех $x\in L_0$.

Задача о продолжении линейного функционала часто встречается в анализе. Центральное место в этой теме занимает следующая теорема.

Теорема Хана-Банаха. Пусть f_0 – линейный непрерывный функционал, заданный на подпространстве L_0, $L_0\subset L$. Тогда функционал f_0 может быть продолжен до некоторого линейного непрерывного функционала f на всем пространстве L без увеличения нормы, то есть так, что

$$\left\|f_0\right\|_{L_0}=\left\|f\right\|_L.$$

Можно дать геометрическую интерпретацию теореме Хана-Банаха. Как следует из утверждения 2.1, уравнение $f_0(x)=1$ определяет в L_0 гиперплоскость, лежащую на расстоянии

$\frac{1}{\|f_0\|}$ до нуля. Продолжая f_0 без увеличения нормы до функционала на всем L, мы проводим через эту частичную гиперплоскость "большую" гиперплоскость на всем L, причем расстояние до нуля остается прежним.

Приведем примеры построения продолжения линейного функционала на плоскости и в трехмерном пространстве.

Пример 6.1. Пусть L_0 – подпространство l_1^2, определяемое формулой

$$L_0 = \{x \in l_1^2 : x_1 + x_2 = 0\}.$$

На L_0 задан функционал $f_0(x) = x_1 - 2x_2$. Найти продолжение функционала f_0, чтобы выполнялись условия теоремы Хана-Банаха.

Решение.

Напомним, что в общем случае подпространство $l_p^n, 1 \le p < \infty$ - это пространство, элементами которого являются упорядоченные наборы из n действительных чисел $x = (x_1, \ldots, x_n), n \ge 1$. Норма определяется с помощью равенства

$$\|x\| = \left(\sum_{k=1}^{n} |x_k|^p \right)^{\frac{1}{p}}.$$

По условию p=1, n=2.

Вычислим норму функционала f_0. По определению имеем:

$$\|f_0\|_{L_0} = \sup_{x_1 + x_2 = 0} \frac{|x_1 - 2x_2|}{|x_1| + |x_2|} = \frac{3}{2}.$$

Теорема 6.1. Пусть f – линейный функционал, заданный на l_p^n, $1 \le p < \infty$. Тогда существует единственный элемент $a \in l_q^n$ такой, что

$$f(x) = \sum_{i=1}^{n} a_i x_i \qquad (6.1)$$

и

$$\|f\| = \|a\|_{l_q^n} \qquad (6.2)$$

где $\frac{1}{p} + \frac{1}{q} = 1$.

Из теоремы 6.1 следует, что искомый функционал f имеет вид

$$f(x) = a_1 x_1 + a_2 x_2$$

и

$$\|f\|_{l_1^2} = \max(|a_1|, |a_2|).$$

Так как функционал f является продолжением f_0, то $\begin{cases} a_1 x_1 + a_2 x_2 = x_1 - 2x_2 \\ x_1 + x_2 = 0. \end{cases}$

Отсюда следует, что $(a_1 - a_2)x_1 = 3x_1$. Так как это верно для любого $x_1 \in R$, то $a_1 - a_2 = 3$.

Учитывая, что $\|f\|_L = \|f_0\|_{L_0}$, получаем второе условие на числа a_1, a_2: $\max(|a_1|, |a_2|) = \frac{3}{2}$.

Таким образом, нужно решить уравнение $\max(|a_1|, |a_1 - 3|) = \frac{3}{2}$. Построив график $y = \max(|a_1|, |a_1 - 3|)$, находим, что $a_1 = \frac{3}{2}$.

Итак, искомый функционал имеет вид

$$f(x) = -\frac{3}{2}x_1 + \frac{3}{2}x_2.$$

Приведем пример, показывающий, что функционал f_0 может иметь множество продолжений.

Пример 6.2. Пусть подпространство L_0 пространства l_∞^2 определяется формулой

$$L_0 = \{x \in l_\infty^2 : x_1 = x_2\}.$$

а

$$f_0(x) = x_1 + 3x_2.$$

Найти продолжение f_0.

Решение. Используем геометрический подход. Вычислим норму f_0, используя утверждение 3.1. Сейчас гиперплоскость:

$$A_{f_0} = \{x \in L_0 : x_1 + 3x_2 = 1\}$$

состоит из одной точки $M = \left(\frac{1}{4}, \frac{1}{4}\right)$. Так как расстояние от A_{f_0} до 0 равно $\frac{1}{4}$, то $\|f_0\| = 4$.

Построим теперь гиперплоскость (а сейчас это прямая) $A_f = \{x \in L : f(x) = 1\}$ так, чтобы она прошла через точку *M* и расстояние до нуля было равно $\frac{1}{4}$. Напомним, как находить расстояние от точки до прямой. Берем маленький шар с центром в заданной точке и "раздуваем" его до первого касания с прямой. В нашем случае мы должны нарисовать шар с центром в нуле и радиусом $\frac{1}{4}$.

Выясним, как выглядит шар в пространстве l_∞^2. Вспоминая, как задается норма в l_∞^2, получаем формулу

$$B\left[0,\frac{1}{4}\right]=\left\{x\in l_\infty^2:\ \max\left(|x_1|,|x_2|\right)=\frac{1}{4}\right\}.$$

Таким образом, шар в пространстве l_∞^2 – это квадрат со сторонами, параллельными осям координат. Заметим, что точка *M* попала в "вершину" шара $B\left[0,\frac{1}{4}\right]$. Это означает, что можно проводить любую прямую, которая проходит через *M* и заключена между прямыми $x_1=\frac{1}{4}$ и $x_2=\frac{1}{4}$ (прямая не должна пересекать шар). Уравнение любой такой прямой имеет вид

$$a_1\left(x_1-\frac{1}{4}\right)+a_2\left(x_2-\frac{1}{4}\right)=0,\quad a_1,a_2\ge 0.$$

Это уравнение можно записать по-другому

$$\frac{4}{a_1+a_2}\left(a_1x_1+a_2x_2\right)=1.$$

Итак, функционал

$$f(x)=\frac{4}{a_1+a_2}\left(a_1x_1+a_2x_2\right),$$

где $a_1,a_2\ge 0$ и $a_1^2+a_2^2\ne 0$, является искомым.

Заметим, что если точка попадает на грань шара (как в задаче 6.1), то продолжение единственное, а если в вершину (как в задаче 6.2), то продолжений множество. В частности, отсюда следует, что в евклидовом пространстве l_2^2 продолжение единственно, так как шар в l_2^2 круглый.

Пример 6.3. Пусть подпространство L_0 пространства l_1^3 определяется формулой

$$L_0=\left\{x\in l_\infty^2: x_1=t,\ x_2=2t,\ x_3=3t\right\}.$$

а

$$f_0(x)=2x_1+3x_2-x_3.$$

Найти продолжение f_0.

Решение. Вычислим норму функционала f_0.

Имеем

$$\left\|f_0\right\|=\sup_t\frac{\left|2t+6t-3t\right|}{|t|+|2t|+|3t|}=\frac{5}{6}.$$

Функционал f имеет вид

$$f(x)=a_1x_1+a_2x_2+a_3x_3.$$

Так как f является продолжением f_0 и их нормы совпадают, то получаем следующую систему:

$$\begin{cases}a_1x_1+a_2x_2+a_3x_3=2x_1+3x_2-x_3\\ x_1=t\\ x_2=2t\\ x_3=3t\\ \max\left(|a_1|,|a_2|,|a_3|\right)=\frac{5}{6}.\end{cases}$$

Первые четыре условия дают равенство $a_1+2a_2+3a_3=5$. Итак, остается решить такую задачу:

$$\max\left(|5-2a_2-3a_3|,|a_2|,|a_3|\right)=\frac{5}{6}.$$

Можно просто перебрать варианты, когда одно из чисел равно $\frac{5}{6}$, а два других не превосходят $\frac{5}{6}$. Мы решим задачу по-другому. Так как $|5-2a_2-3a_3|\le\frac{5}{6}$, то $\frac{25}{6}\le 2a_2+3a_3\le\frac{35}{6}$.

С другой стороны, $a_2\le\frac{5}{6}$ и $a_3\le\frac{5}{6}$, а значит, $2a_2+3a_3\le\frac{25}{6}$.

Следовательно, существует единственное решение задачи $a_1=a_2=a_3=\frac{5}{6}$.

Заключение

Новые задачи физики и математики, появившиеся в XX столетии, привели к появлению нового раздела математики – функциональный анализ.

Линейные функционалы и их продолжение на все функциональное пространство с сохранением нормы является мощным математическим аппаратом для развития следующих направлений современной математики:

1. Некоммутативный функциональный анализ.
2. Теория операторных алгебр.
3. Вероятностный и стохастический функциональный анализ.
4. Функциональный анализ на пространствах фрактальных структур.

Литература

1. Лебедев, В. И. Функциональный анализ и вычислительная математика: учеб. пособие. – 4-е изд., перераб. и доп. – М.: Физматлит, 2005. – 296 с.

2. Колмогоров, А. Н. Элементы теории функций и функционального анализа / А. Н. Колмогоров, С. В. Фомин. – 7-е изд. – М.: Физматлит, 2004. – 572 с.

3. Треногин, В. А. Функциональный анализ: учебник. – 4-е изд., испр. – М.: Физматлит, 2007. – 488 с.

4. Фёдоров, В. М. Курс функционального анализа. – СПб.: Лань, 2005. – 352 с.

5. Кудрявцев, Л. Д. Краткий курс математического анализа: учебник: в 3 т. Т. 1-2. – 3-е изд. – М.: Физматлит, 2005. – 424 с.

6. Крылов, В. И. Вычислительные методы высшей математики: в 2 т. / В. И. Крылов, В. В. Бобков, П. И. Монастырный. – Минск: Вышэйшая школа, 1972. – 584 с.

7. Краснов, М. Л. Интегральные уравнения (введение в теорию). – М.: Ком-Книга, 2010. – 304 с.

8. Волков, В. Т. Интегральные уравнения. Вариационное исчисление: курс лекций: учеб. пособие / В. Т. Волков, А. Г. Ягола. – 2-е изд., испр. – М.: Изд-во КДУ, 2009. – 140 с.

9. Суетин, П. К. Классические ортогональные многочлены. – М.: Физматлит, 2007. – 480 с.

10. Романко, В. К. Курс дифференциальных уравнений и вариационного исчисления. – 2-е изд. – М: Лаборатория Базовых Знаний, 2001. – 344 с.

Добровольский Ю.Н. Теорема Хана-Банаха – как одна из основных теорем функционального анализа. *Рассмотрена история возникновения теоремы Хана-Банаха. Введены основные линейные нормированные пространства, линейные непрерывные функционалы и их нормы. Введено понятие гиперподпространство, гиперплоскость. Приведены примеры на вычисление нормы функционала и теорема о продолжении функционала на все пространство с сохранением нормы. Линейные функционалы и их продолжение на все функциональное пространство с сохранением нормы является мощным математическим аппаратом для развития многих направлений современной математики.*

Ключевые слова: *функциональные пространства, линейные функционалы, гиперподпространство, гиперплоскость, норма функционала, теорема Хана-Банаха.*

Dobrovolsky Y.N. Khan-Banach theorem - as one of the main theorems of functional analysis. *The history of the Khan-Banach theorem is considered. Basic linear normalized spaces, linear continuous functionals and their norms are introduced. Introduced the concept of hyperpodpro-space, hyperplane. Examples are given for calculating the functional norm and the theorem on extending the functional to the entire space while maintaining the norm. Linear functionals and their extension to the entire functional space while preserving the norm are a powerful mathematical tool for the development of many areas of modern mathematics.*

Key words: *functional spaces, linear functionals, hyperspace, hyperplane, functional norm, Khan-Banach theorem.*

Статья поступила в редакцию 02.02.2025
Рекомендована к публикации профессором Павлышом В. Н.

Об авторах

Айдин Сергей Анатольевич - студент кафедры программной инженерии им. Л. П. Фельдмана факультета интеллектуальных систем и программирования ФГБОУ ВО «Донецкий национальный технический университет».

Бабич Иван Викторович – студент кафедры прикладной математики и искусственного интеллекта факультета интеллектуальных систем и программирования ФГБОУ ВО «Донецкий национальный технический университет».

Боднар Алина Валериевна - кандидат технических наук, доцент, доцент кафедры программной инженерии им. Л. П. Фельдмана факультета интеллектуальных систем и программирования ФГБОУ ВО «Донецкий национальный технический университет».

Бишаров Дмитрий Андреевич - магистрант кафедры автоматизированных систем управления факультета информационных систем и технологий ФГБОУ ВО «Донецкий национальный технический университет».

Васяева Татьяна Александровна - кандидат технических наук, доцент, доцент кафедры автоматизированных систем управления факультета информационных систем и технологий ФГБОУ ВО «Донецкий национальный технический университет».

Ефименко Константин Николаевич - кандидат технических наук, доцент, доцент кафедры прикладной математики и искусственного интеллекта факультета интеллектуальных систем и программирования ФГБОУ ВО «Донецкий национальный технический университет».

Добровольский Юрий Николаевич - кандидат технических наук, доцент, доцент кафедры прикладной математики и искусственного интеллекта факультета интеллектуальных систем и программирования ФГБОУ ВО «Донецкий национальный технический университет».

Зори Сергей Анатольевич - доктор технических наук, доцент, заведующий кафедрой программной инженерии им. Л. П. Фельдмана факультета интеллектуальных систем и программирования ФГБОУ ВО «Донецкий национальный технический университет».

Кочетуров Виталий Владимирович - магистрант кафедры программной инженерии им. Л. П. Фельдмана факультета интеллектуальных систем и программирования ФГБОУ ВО «Донецкий национальный технический университет».

Лукащук Михаил Олегович - магистрант кафедры программной инженерии им. Л. П. Фельдмана факультета интеллектуальных систем и программирования ФГБОУ ВО «Донецкий национальный технический университет».

Мартыненко Татьяна Владимировна - кандидат технических наук, доцент, доцент кафедры автоматизированных систем управления факультета информационных систем и технологий ФГБОУ ВО «Донецкий национальный технический университет».

Матях Ирина Владимировна - ассистент кафедры автоматизированных систем управления факультета информационных систем и технологий ФГБОУ ВО «Донецкий национальный технический университет».

Павлов Даниил Андреевич – магистрант кафедры программной инженерии им. Л. П. Фельдмана факультета интеллектуальных систем и программирования ФГБОУ ВО «Донецкий национальный технический университет».

Савкова Елена Осиповна - кандидат технических наук, доцент, доцент кафедры автоматизированных систем управления факультета информационных систем и технологий ФГБОУ ВО «Донецкий национальный технический университет».

Федяев Олег Иванович - кандидат технических наук, доцент, доцент кафедры программной инженерии им. Л. П. Фельдмана факультета интеллектуальных систем и программирования ФГБОУ ВО «Донецкий национальный технический университет».

Хомичук Надежда Викторовна - магистрант кафедры программной инженерии им. Л. П. Фельдмана факультета интеллектуальных систем и программирования ФГБОУ ВО «Донецкий национальный технический университет».

Хрюкин Евгений Александрович - аспирант кафедры автоматизированных систем управления факультета информационных систем и технологий ФГБОУ ВО «Донецкий национальный технический университет».

Шворoб Данил Сергеевич - ассистент кафедры патологической анатомии Донецкого государственного медицинского университета им. М. Горького, ORCID 0000-0001-6578-0050.

**Требования к статьям,
направляемым в редакцию научного журнала
«Информатика и кибернетика»**

Редколлегией принимаются к рассмотрению статьи, в которых рассматриваются важные вопросы в области информатики и кибернетики. Научный журнал издаётся с 2015 года, периодичность издания – 4 раза в год.

В журнале предусмотрены следующие рубрики:

- информатика и вычислительная техника;
- компьютерные и информационные науки;
- инженерное образование.

В соответствии с номенклатурой специальностей научных работников МОН ДНР первые две рубрики соответствуют следующим укрупненным группам специальностей научных работников:

05.01 – «Инженерная геометрия и компьютерная графика»,

05.13 – «Информатика, вычислительная техника и управление».

С 01.02.2019 Научный журнал включён в Перечень рецензируемых научных изданий, в которых должны быть опубликованы основные научные результаты диссертаций на соискание учёной степени кандидата наук, на соискание учёной степени доктора наук (приказ МОН ДНР № 135) по группам специальностей 05.01.00 и 05.13.00.

Рубрика «Инженерное образование» предназначена опубликования сотрудниками научно-методических статей.

Журнал также включён в базу данных РИНЦ (Российский индекс научного цитирования) (лицензионный договор № 425-07/2016 от 14.07.2016).

Статьи, представляемые в данный сборник, должны отвечать следующим требованиям. ***Содержание статьи*** должно быть посвящено актуальным научным проблемам и включать следующие необходимые элементы:

- постановку проблемы в общем виде, её связь с важными научными и практическими задачами;
- анализ последних исследований и публикаций, в которых решается данная задача и на которые опирается автор, выделение нерешенных ранее частей общей проблемы, которым посвящается статья;
- формулировка цели статьи и постановка задач, решаемых в ней;
- изложение основного материала с полным обоснованием полученных научных результатов;
- выводы и перспективы последующих исследований в данном направлении.

Каждый элемент должен быть выделен соответствующим названием раздела, например, «введение», «постановка задачи», «цель и задачи работы», «цель статьи», «цель исследования», «цель разработки», «анализ … », «сравнительная оценка … », «разработка … », «проектирование … », «программная реализация», «тестирование … », «полученные результаты», «выводы», «литература». Разделы «введение», «выводы», «литература» являются обязательными. Включать в названия разделов нумерацию не разрешается.

В основном тексте статьи формулируются и обосновываются полученные авторами утверждения и результаты. Выводы должны полностью соответствовать содержанию основного текста. Языки публикаций: русский, английский.

Объём статьи, формат страницы

Для оформления статьи следует использовать листы формата А4 (210х297 мм) с полями по 2,5 см со всех сторон. Нумерацию страниц выполнять не нужно.

Рекомендуемый объём статьи – 6-12 страниц. Рукописи меньшего объёма могут быть рекомендованы к публикации в качестве коротких сообщений.

Последняя страница текста статьи должна быть заполнена не менее чем на две трети, но содержать не менее трёх пустых строк в конце.

Форматирование текста

Подготовка статьи осуществляется в текстовом редакторе Microsoft Office Word.

Весь текст статьи оформляется шрифтом Times New Roman 10 пт с одинарным междустрочным интервалом, если ниже в требованиях не сказано иного. Абзацный интервал «перед» – 0 пт, «после» – 0 пт.

На первой строке с выравниванием по левому краю располагается УДК.

Заголовок (название) статьи оформляется шрифтом Times New Roman 14 пт, полужирное начертание, с выравниванием по центру (без абзацных отступов). Заголовок статьи следует печатать с прописной буквы без точки в конце, переносы слов не допускаются. Абзацный интервал «перед» – 12 пт, «после» – 12 пт.

После названия статьи следует информация об авторах, которая выравнивается по центру (без абзацных отступов). На одной строке указываются инициалы и фамилии всех авторов через запятую. Между двумя инициалами ставится пробел. С новой строки указывается название вуза (организации) и город (для каждого автора, если не совпадают). На следующей строке указываются адреса электронной почты (один адрес либо каждого автора – по желанию). Адрес электронной почты оформляется в виде гиперссылки.

К тексту аннотации применяется курсивное начертание, с выравниванием по ширине, отступы слева и справа по 1 см. Заголовок «Аннотация» выделяется полужирным начертанием. Объём аннотации – 450-550 символов (без пробелов). Абзацный интервал «перед» – 12 пт, «после» – 12 пт.

Основной текст статьи разбивается на две колонки шириной по 7,5 см (промежуток между столбцами – 0,99 см), выравнивается по ширине. Абзацный отступ первой строки – 1 см. Автоматический перенос слов не применяется.

Заголовки разделов выполняются шрифтом Arial 10 пт, полужирное курсивное начертание. Абзацный отступ отсутствует, интервал перед абзацем – 12 пт, после абзаца – 6 пт. Для заголовка «Введение» установить интервал «перед» – 0 пт, «после» – 6 пт.

Таблицы в тексте статьи

Название следует помещать над таблицей с абзацного отступа (1 см) в формате: слово «Таблица», пробел, номер таблицы, пробел, тире, пробел, название таблицы. Название таблицы записывают с прописной буквы без точки в конце строки и выравнивают по ширине. В ячейках таблицы устанавливается выравнивание текста по центру по вертикали. По горизонтали текст выравнивается по центру либо по левому краю. Границы ячеек таблицы должны быть только чёрного цвета, толщина линии – 1 пт. На все таблицы должны быть приведены ссылки в тексте статьи, при ссылке следует писать слово «табл.» с указанием её номера, например, « … данные приведены в табл. 5». Таблицы нумеруются в пределах статьи. Таблица располагается сразу после ссылки на неё, если это возможно (например, после окончания абзаца). Если же таблица не помещается на текущей странице, то она должна быть расположена в начале следующей страницы (или колонки). При необходимости допускается включение в статью таблицы, ширина которой превышает ширину колонки. В этом случае таблица и её название размещаются по центру страницы. Таблица не должна выступать за границы полей страницы. Таблица и её название отделяются от основного текста статьи одной пустой строкой до и после.

Рисунки в статье

Ссылки на иллюстрации по тексту статьи обязательны и оформляются в виде « ... на рис. 2» и т. п. Рисунок и его подпись выравниваются по центру колонки (без абзацных отступов), положение рисунка – «в тексте». Размещается рисунок после его первого упоминания в тексте, если это возможно (например, после окончания абзаца). Если же иллюстрация не помещается на текущей странице, то она должна быть расположена в начале следующей страницы (или колонки). При необходимости допускается включение в статью рисунка, ширина которого превышает ширину колонки. В этом случае рисунок и его подпись выравниваются по центру страницы. Иллюстрация не должна выступать за границы полей страницы. Подпись рисунка оформляется в формате: слово «Рисунок», пробел, номер иллюстрации, пробел, тире, пробел,

название рисунка. Название рисунка записывают с прописной буквы без точки в конце строки. Для подписи иллюстрации применяют курсивное начертание. Иллюстрация и её подпись отделяются от основного текста статьи одной пустой строкой до и после. Не допускается выполнять рисунки с помощью встроенного графического редактора Microsoft Office Word. Если на иллюстрации имеется текст, размер шрифта должен быть не менее чем аналогичный текст, набранный шрифтом Times New Roman 10-го размера. Иллюстрация не должна содержать много незаполненного пространства.

Формулы

Формулы и уравнения рекомендуется набирать с использованием MathType (предпочтительно) или MS Equation. Формулы и математические символы не должны существенно отличаться по размеру от основного текста. Обязательной является нумерация формул, на которые имеется ссылка в тексте статьи. Ссылки в тексте на порядковые номера формул дают в скобках, например, « ... согласно формуле (2)». Формулы размещаются по центру колонки, а их номера – по правому краю. Как для строки с формулой, так и для первой строки пояснений (при наличии), абзацный отступ убирается. Первая строка пояснения начинается со слова «где», после которого следует поставить табуляцию на 1 см, затем само пояснение в формате: символ, подлежащий объяснению, пробел, тире, пробел, поясняющий текст, запятая, обозначение единицы измерения физической величины. Пояснения перечисляются через точку с запятой, выравниваются по ширине. Вторая и последующие строки пояснений начинаются с абзацного отступа (1 см). Весь блок текста, связанный с формулой (только формула, несколько формул подряд или формула с пояснениями), отделяется от основного текста одной пустой строкой до и после. Переносить формулы на следующую строку допускается только на знаках выполняемых операций, причем знак в начале следующей строки повторяют. При переносе формулы на знаке умножения применяют знак «×». Формулы и математические уравнений могут быть записаны в тексте документа, если их высота не превышает высоту строки. При этом следует учитывать, что знаки математических операций отделяются от чисел или символов пробелами с обеих сторон. Например, «Если учесть, что $y < 0$ и $2x + y = 1$, то из формулы (3) можно выразить x…». К символам, которые приведены в формуле, при дальнейшем их употреблении (в том числе в пояснениях к формуле) должно применяться курсивное начертание. При этом к любым числам (верхние и нижние индексы, содержащие цифры и т. п.), а также к математическим знакам курсивное начертание не применяется. Не допускается вставлять формулы, выполненные в виде рисунков.

Перечисления: оформление списков

Основной текст статьи может содержать перечисления, оформленные в виде маркированного списка. В качестве маркера элемента списка разрешается использовать только короткое тире «–». Каждый элемент перечисления записывается с новой строки с абзацного отступа, равного 1 см. После символа короткого тире текст располагается с отступом в 1,5 см от левой границы строки, выравнивается по ширине, при переносе на новые строки располагается без отступов. Нумерованные и многоуровневые списки включать в статью не разрешается.

Литература

В тексте статьи обязательны ссылки на все литературные источники, номер источника указывается в квадратных скобках. Ссылки на неопубликованные работы не допускаются. Рекомендуемое количество источников, на которые ссылается автор, не менее 10. Перечень источников приводится в порядке их упоминания в статье. Библиографическое описание каждого литературного источника оформляется в соответствии с ГОСТ Р 7.0.100–2018. Перечень литературных источников оформляется в виде нумерованного списка. В качестве маркеров элементов списка используют порядковые арабские цифры с точкой. Каждый источник представляет собой отдельный элемент перечисления, записывается с новой строки с абзацного отступа, равного 1 см. После порядкового номера с точкой текст располагается с отступом в 1,5 см от левой границы строки, выравнивается по ширине, при переносе на новые строки располагается без отступов.

В конце статьи обязательно приводятся аннотации на русском и английском языках, каждая заканчивается перечнем 5-6 ключевых слов.

К тексту аннотации применяется курсивное начертание, с выравниванием по ширине, отступы слева и справа по 1 см. Слово «Аннотация» опускается. Текст аннотации начинается с ФИО авторов и названия статьи, выделяемых полужирным начертанием. Аннотация на русском языке совпадает с аннотацией, приведенной в начале статьи. В тексте аннотации на английском языке после фамилии автора указывается только первая буква имени с точкой. Абзацный интервал «перед» – 12 пт, «после» – 12 пт. Ключевые слова оформляются с новой строки аналогично тексту аннотации. Заголовок «Ключевые слова:» (англ. «Keywords:») выделяется полужирным начертанием. Ключевые слова перечисляются через запятую.

Порядок представления статьи и сопроводительные документы

В редакцию необходимо представить:

– файл с текстом статьи;

– файл, содержащий фамилию, имя и отчество авторов полностью; ученую степень, ученое звание; место работы с полным указанием должности, подразделения и наименования организации, города (страны); номера телефонов и e-mail для связи;

– экспертное заключение о возможности публикации статьи, подписанное руководителем и заверенное печатью организации, в которой работает автор статьи;

– выписка из заседания кафедры или письмо организации с просьбой об опубликовании и указанием, что изложенные в статье результаты ранее не публиковались.

Статьи и сопроводительные документы следует высылать на электронный адрес infcyb.donntu@yandex.ru.

К сведению авторов

Если статья оформлена с нарушением указанных выше требований и правил, редакция после предварительного рассмотрения может отклонить статью.

На рецензирование статьи направляются членам редакционной коллегии журнала. Все статьи публикуются при наличии положительной рецензии.

В статью могут быть внесены изменения редакционного характера без согласования с автором. Ответственность за содержание статьи и качество перевода аннотаций несут авторы.

Публикация статей в научном журнале «Информатика и кибернетика» осуществляется на некоммерческой основе.

Все номера Научного журнала размещаются в электронной библиотечной системе ФГБОУ ВО «ДонНТУ» и на сайте http://infcyb.donntu.ru/.

.

CONTENT

Informatics and computer engineering

Компьютерные науки

Электронное периодическое издание

Научный журнал

ИНФОРМАТИКА И КИБЕРНЕТИКА

(на русском, английском языках)

№ 1 (39) - 2025

Ответственный за выпуск Р. В. Мальчева

Технический редактор Р. В. Мальчева

Компьютерная верстка Р. В. Мальчева

Подписано к выпуску 18.03.2025. Усл. печ. лист. 9,0. Уч.-изд. лист. 5,5.
Адрес редакции: ДНР, 283001, г. Донецк, ул. Артема, 58, ФГБОУ ВО «ДонНТУ»,
4-й учебный корпус, к. 36.
Тел.: +7 (856) 301-07-35, +7 (949) 334-89-11
E-mail: infcyb.donntu@yandex.ru, URL: http://infcyb.donntu.ru